UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Proyecto de Fin de Grado en Ingeniería en Tecnologías de la Información

SISTEMA DE MONITORIZACIÓN DE ESTADO Y CONTROL PARA UN HUERTO DOMÉSTICO

MARCOS COLMENERO FERNÁNDEZ

Dirigido por: ALFONSO URQUÍA MORALEDA

Curso: septiembre de 2020

SISTEMA DE MONITORIZACIÓN DE ESTADO Y CONTROL PARA UN HUERTO DOMÉSTICO

Proyecto de Fin de Grado en Ingeniería en Tecnologías de la Información de modalidad específica

MARCOS COLMENERO FERNÁNDEZ

Dirigido por: ALFONSO URQUÍA MORALEDA

Fecha de lectura y defensa: septiembre de 2020

Agradecimientos

A mis padres y a mi hermano, por creer en mí. Estoy orgulloso de vosotros.

A mi mujer Paula, y a su familia; por mostrarme cada día que el esfuerzo y la dedicación dan sus frutos.

A María Antonia, por haberme regalado gran parte del tiempo que aquí he invertido.

Resumen general

El objetivo del proyecto ha sido la creación de una herramienta de apoyo a la toma de decisiones, siendo su ámbito de actuación el mantenimiento de un huerto doméstico.

Se ha desarrollado un sistema configurable, capaz de monitorizar el estado del medio físico, mantener una base de datos de estado de carácter temporal, generar alertas, obtener predicciones de sistemas externos, adaptar su comportamiento y actuar sobre el entorno.

Está formado por un conjunto de sensores ambientales conectados a un microcontrolador programado, servicios de comunicación y almacenamiento de datos, una bomba de riego y una interfaz web multidispositivo que permite supervisar y controlar el sistema.

Permite la monitorización en tiempo real de parámetros ambientales en el contenedor de plantación, el almacenamiento de estados en un sistema externo y el análisis gráfico de series temporales de datos. Incluye un sistema de alerta basado en umbrales preestablecidos y un mecanismo autónomo de riego, capaz de determinar las necesidades diarias de agua del contenedor.

El dispositivo físico está formado por un microcontrolador PHOTON, al que se han añadido mediante una placa de expansión un conjunto de sensores GROVE, una bomba de riego y un sistema de alimentación.

El firmware del dispositivo se ha diseñado sobre la abstracción de una máquina de estados finitos y se ha codificado en C++. Envía las lecturas realizadas por los sensores a un sistema de almacenamiento externo y obtiene la predicción meteorológica de la Agencia Española de Meteorología.

El sistema de almacenamiento se ha implementado en Firestore, una base de datos NoSQL en tiempo real de Firebase. Firebase es la

plataforma de Google para el desarrollo de aplicaciones web y móviles, y sirve también como plataforma de alojamiento de la aplicación web de control.

El intercambio de mensajes realizado por el firmware emplea un mecanismo basado en webhooks. Los webhooks han sido creados y alojados en Particle Cloud.

Finalmente, se ha desarrollado una aplicación web responsive, mediante la cual es posible analizar la información almacenada y configurar el autómata. Para ello, se ha hecho uso de JQuery, del framework Bootstrap4 y algunas librerías JavaScript como Charts.js y Datatables.js.

Particle Cloud y Firebase cuentan con herramientas CLI, de consola y editores web; no obstante, la herramienta más utilizada ha sido Visual Studio Code.

El resultado final del proyecto es una herramienta capaz de monitorizar el medio físico, obtener predicciones meteorológicas y actuar de acuerdo con una configuración. Genera y almacena información de estado y funcionamiento. Además, cuenta con una aplicación web que permite:

- Configurar y supervisar el sistema.
- Analizar la información generada y tomar decisiones.

La herramienta ha sido empleada para controlar varios ciclos de cultivo de pasto de trigo. Ha regado de forma autónoma, ha permitido conocer las condiciones ambientales y ha servido para analizar con posterioridad la información generada durante el cultivo. Ha resultado útil tanto durante como después de los ciclos de plantación. La herramienta cumple los requisitos establecidos.

Palabras Clave:

IoT, microcontroladores, PaaS, FaaS, serverless, autómata finito, riego inteligente, control, monitorización, representación de datos, webhooks, interfaces responsive.

General summary

The main objective of this project has been the creation of a tool to support decision-making. It's focused on the maintenance of a home garden.

A configurable system has been developed, capable of monitoring the state of the environment, keeping a database of temporary states, generating alerts, obtaininig predictions from external systems, adapting its behavior and acting on the environment.

It consists of a set of environmental sensors connected to a microcontroller, communication and data storage services, an irrigation pump and a responsive web interface that allows the system to be controlled.

It allows real-time monitoring of environmental parameters, storing states in an external system and the graphic analysis of historical data. It includes an alert system based on pre-established thresholds and an autonomous irrigation mechanism, capable of determining the daily water needs of the container.

The physical device consists of a PHOTON microcontroller, to which a set of GROVE sensors, an irrigation pump and a power system have been added by means of an expansion board.

The device firmware has been developed on the abstraction of a finite state machine and has been encoded in C ++. It sends the readings made by the sensors to an external storage system and gets the weather forecast from the Spanish Agency of Meteorology.

The storage system has been created in Firestore, wich is a Firebase NoSQL real-time database. Firebase is Google's platform for web and mobile applications development, and it also serves as hosting platform for the web application.

The exchange of messages carried out by the firmware uses a webhooks based mechanism. The webhooks have been created and hosted in Particle Cloud.

Finally, a responsive web application has been developed, through which it is possible to analyze the information stored and configure the finite state machine. JQuery, Bootstrap4 and some JavaScript libraries such as Charts.js and Datatables.js have been used.

Particle Cloud and Firebase have their CLI and console tools, also web editors; however, the most used tool has been Visual Studio Code.

The result of the project is a tool capable of monitoring the physical environment, obtaining weather forecasts and acting according to its configuration. The tool generates and stores status and operating information. In addition, there is a web Application that allows:

- Configuring and monitoring the system.
- Analyze the information generated and take decisions according with it.

The tool has been used to control several cycles of wheat grass growing. It has watered autonomously, it has made it possible to know the environmental conditions and it has served to subsequently analyze the information generated during cultivation. It has been useful both during and after planting cycles. The tool meets the established requirements.

Keywords:

IoT, microcontrollers, PaaS, FaaS, serverless, finite states machine, intelligent irrigation, control, monitoring, graphs, webhooks, responsive interfaces.

Índice

Índice de tablas

Índice de figuras

Capítulo 1

Introducción, objetivos y estructura

1.1 Introducción

La premisa inicial del proyecto ha sido la creación de una herramienta de apoyo a la toma de decisiones. Para ello, se ha fabricado un sistema completo, diseñando e implementando no sólo el software, sino también el hardware desde su inicio. El ámbito de actuación de la herramienta es el cuidado de un huerto doméstico.

La producción de vegetales para el autoconsumo ha ganado importancia en la sociedad del siglo XXI. Los huertos urbanos pueden suponer una medida de estabilización de la economía doméstica y/o comunitaria ante determinadas situaciones. No son sólo una forma de autoabastecimiento, sino una forma de expresión y de relación con la naturaleza.

Aportan beneficios:

- Alimentos de calidad.
- Salud física y mental.

Fomentan valores:

- Respeto por la naturaleza.
- Concienciación ambiental.
- Concienciación nutricional.
- Intercambio comunitario y establecimiento de relaciones.

En el ámbito de los huertos domésticos resulta habitual ubicar las plantas y determinar las necesidades de riego en base a suposiciones que no suelen ser correctas.

En consonancia con los valores expuestos, este proyecto ha supuesto la creación de una herramienta que permite supervisar el estado ambiental de un módulo de plantación, contrastar ese estado frente a los umbrales óptimos de crecimiento de las plantas que contiene y determinar de forma independiente las necesidades de riego del contenedor. Además, incluye un sistema de alerta, gestiona series temporales de estado y riego, administra la configuración del contenedor y lleva el seguimiento de las siembras y recolecciones realizadas.

La herramienta permite:

- Maximizar el aprovechamiento del agua.
- Supervisar el entorno ambiental de las plantas.
- Maximizar el aprovechamiento de los contenedores de plantación.
- Analizar los ciclos de plantación y extraer conclusiones.

Ayuda a elegir los lugares en los que ubicar los contenedores, las plantas que debemos sembrar en cada contenedor y el momento en que debemos hacer estas selecciones.

Modifica la forma de cultivar, que deja de basarse en la intuición y la observación de las plantas, para hacerlo en datos medidos.

1.2 Objetivos

La utilidad de la herramienta se centra en la monitorización de estado, control de riego y configuración de un contenedor de plantación dentro de un huerto doméstico. También en la capacidad de almacenamiento y análisis de la información generada durante los ciclos de plantación.

El objetivo es mantener el huerto en condiciones óptimas de estado y mejorar su rendimiento. Para ello, la herramienta ayuda a responder las siguientes cuestiones:

- ¿Cuál es el estado actual del módulo? ¿Y su evolución?
- ¿Son adecuadas las condiciones del módulo a las plantas que contiene?
- ¿Cuándo estoy regando? ¿Cuánto estoy regando?
- ¿Cómo estoy aprovechando la superficie disponible?
- ¿Qué rentabilidad estoy obteniendo del módulo?
- ¿Cómo está siendo la transición de estados del autómata?

Además, permite administrar:

- Configuración del módulo.
- Listado heurístico de plantas.
- Listado general de tareas.
- Listados de siembras y recolecciones.

El sistema define los siguientes objetivos de funcionamiento:

1. Monitorización de estado. Se entiende como estado el conjunto de valores medidos por los sensores ambientales en un determinado instante. La monitorización será en tiempo real e histórica. Se miden temperatura y humedad ambiente, temperatura y humedad del sustrato de cultivo, e intensidad de la luz ambiente.

2. Evaluación del estado. En caso de que alguno de los valores medidos se encuentre fuera de los umbrales establecidos, activará un sistema de alerta. Se establecen umbrales de temperatura y humedad ambiente, humedad de sustrato y cantidad mínima de horas de luz.

3. Comunicación y almacenamiento de estado. En intervalos regulares, que podrán ser configurados, los valores medidos serán enviados al sistema de almacenamiento externo. Esto es, a una base de datos NoSQL alojada en Firebase.

4. Descarga de información externa. Se obtendrán predicciones meteorológicas y de estimación de riego provenientes de la Agencia Española de Meteorología y del Sistema de Información al Regante del Ministerio de Agricultura. Los valores de las predicciones utilizados serán:
    a. Probabilidad de lluvia diaria para el municipio en el que se ha ubicado el módulo de plantación. La predicción será descargada haciendo uso del API:
        - *https://opendata.aemet.es/dist/index.html*
    b. Evapotranspiración de referencia de la estación SIAR más cercana a la ubicación del módulo de plantación, para lo que se emplea la aplicación SIAR. El valor obtenido se almacena en Firebase, desde donde será descargado por el firmware del autómata.

5. Apoyo a la toma de decisiones. Control de riego. Las mediciones realizadas y la información descargada servirán para planificar y ejecutar el riego diario, actuando así sobre el valor de humedad en el sustrato. El mecanismo de control de riego implementa un autómata finito determinista, basado en las recomendaciones de riego diario enunciadas en el cuaderno 56 de la FAO.

6. Administración del módulo. Todo el sistema será configurable mediante una aplicación diseñada para ello.

7. Mantenimiento de listados: heurísticos y de apoyo. Se mantendrá una base de datos de plantas en la que figurarán los umbrales de crecimiento óptimo de cada planta. Se mantendrán también otros listados:
    a. Tareas de mantenimiento a realizar en el módulo.
    b. Momentos de siembra.
    c. Recolecciones realizadas.

8. Representación de información y apoyo a la toma de decisiones. Toda la información almacenada se podrá consultar, haciendo uso de la aplicación de control, de forma gráfica y en listados, para facilitar su interpretación. Los gráficos mostrarán los niveles de temperatura y humedad, tanto del sustrato como del ambiente; así como horas de luz diarias y cantidad de agua empleada en el riego. Los datos representados en los gráficos podrán ser consultados en forma de tabla, para obtener un mayor nivel de detalle. También serán representados los valores actuales de los distintos sensores, la configuración del módulo y el nivel de alerta asociado a los umbrales controlados.

Para conseguir los objetivos se han dividido las tareas, agrupándolas en subproyectos que pudiesen desarrollarse de forma independiente y paralela unos de otros. Desde un punto de vista metodológico, la ejecución de cada subproyecto ha seguido un ciclo iterativo dividido en etapas:

1. Análisis de requisitos.
2. Estudio y reflexión.
3. Diseño.
4. Implementación.
5. Evaluación.

El resultado obtenido en la fase de implementación ha sido evaluado en cada iteración. Las iteraciones han servido para corregir y/o añadir

características al diseño inicial. Se ha iterado sobre las tres últimas fases hasta conseguir una versión estable y acorde a los objetivos de cada subproyecto.

La división en subproyectos ha sido la siguiente:

1. Selección de plataforma IoT. Ha sido necesario encontrar una plataforma que ofrezca servicios de registro de dispositivos, diagnóstico, administración y actualización remotas; mecanismos de comunicación y herramientas de programación de microcontroladores. Se ha elegido Particle.io.

2. Creación del dispositivo físico. Formado por un microcontrolador PHOTON, al que se ha añadido un sistema de gestión de energía (basado en la placa Photon Power Shield), un conjunto de sensores ambientales (GROVE, de seeedstudio.com) y una bomba peristáltica conectada a un relé. El resultado ha sido un dispositivo programable, que se ha instalado en una caja estanca con grado de protección IP65.

3. Programación del dispositivo. Los microcontroladores de Particle.io tienen dos piezas de código separado: DeviceOS y firmware de aplicación. El DeviceOS abstrae el hardware y permite hacer uso de los servicios Particle Cloud desde el firmware de aplicación. Se ha implementado un autómata finito determinista, encargado de ejecutar los algoritmos de monitorización de estado y control de riego. El autómata se comunica con otros sistemas por medio de webhooks alojados en Particle Cloud. A través del API de Particle.io es posible leer variables y ejecutar funciones del dispositivo desde una aplicación externa.

4. Selección de plataforma de alojamiento. Para el desarrollar el proyecto se ha empleado una solución PaaS de Google centrada en la creación de aplicaciones web y móviles: Google Cloud Firebase.

Firebase ofrece soluciones al desarrollador centradas en tres ámbitos:

a. Desarrollo de la aplicación.
b. Análisis y mejora continua.
c. Atracción y retención de usuarios.

5. Sistema de comunicaciones. Haciendo uso del protocolo HTTP, la arquitectura REST y diversos medios de autenticación y autorización, la aplicación web y nuestro autómata intercambian documentos JSON entre sí y con otros sistemas.
    a. Para cumplir los requisitos de funcionamiento establecidos, el autómata:
        - Obtiene predicciones meteorológicas. AEMET.
        - Descarga la predicción de necesidades hídricas. Desde Firebase.
        - Almacena información de funcionamiento y estado. En Firebase.
    b. Por medio de las API de Particle Cloud y Firebase, la aplicación de control:
        - Obtiene las variables del autómata.
        - Ejecuta funciones en el firmware del autómata.
        - Realiza operaciones CRUD en Firestore.

6. Diseño e implementación del sistema de almacenamiento. Se ha elegido Firestore, la base de datos NoSQL en tiempo real de la plataforma Firebase como sistema de almacenamiento. En Firestore los datos se agrupan en colecciones, las colecciones contienen documentos y los documentos a su vez contienen campos, a los que se asignan valores.

7. Desarrollo de la aplicación de control. Se ha creado una aplicación web responsive, estructurada como un cuadro de mandos desde el

que es posible conocer el estado del sistema, analizar la información almacenada y configurar el dispositivo. En el desarrollo de la aplicación se ha empleado JQuery, Bootstrap4, la plantilla CSS del proyecto Material Design Bootstrap y librerías de representación de datos como Charts.js y Datatables.js. Han sido empleados también los SDK JavaScript de Particle y Firebase.

1.3 Estructura

El proyecto ha sido dividido en diez capítulos, que pueden ser agrupados en tres bloques:

1. Un primer bloque dedicado al planteamiento, la investigación y el desarrollo teórico. Comprende los capítulos 1 y 2.

2. Un segundo bloque dedicado a la implementación de sistemas. Comprende los capítulos del 3 al 7.

3. Un tercer bloque dedicado a la realización de pruebas, análisis de resultados y reflexión en torno a los resultados obtenidos. Comprende los capítulos del 8 al 10.

Estructura final de capítulos:

- Capítulo 1: Introducción, objetivos, tareas y estructura de la memoria. En este capítulo se establecen los objetivos del proyecto, tanto el objetivo final como los objetivos de funcionamiento y los resultados esperados. Se detallan las tareas realizadas, el conocimiento adquirido, las herramientas empleadas y la metodología seguida.

- Capítulo 2: Marco teórico. En el que se tratan conceptos teóricos cuya comprensión resulta necesaria.

- Capítulo 3: Desarrollo del sistema hardware.

- Capítulo 4: Desarrollo del autómata.
- Capítulo 5: Desarrollo del sistema de comunicaciones.
- Capítulo 6: Desarrollo del sistema de almacenamiento.
- Capítulo 7: Desarrollo de la aplicación de control.
- Capítulo 8: Pruebas y análisis.
- Capítulo 9: Planificación y gestión de proyecto.
- Capítulo 10: Conclusiones y trabajos futuros.
- Anexos: Código desarrollado.
    a. Anexo I: Código del autómata.
    b. Anexo I: Funciones suscriptoras.
    c. Anexo I: Aplicación de control.

Capítulo 2

Marco teórico

2.1 Introducción

En el primer capítulo de la memoria se ha expuesto de forma detallada el proyecto. Se han enunciado los objetivos, la metodología empleada y las tareas realizadas. Finalmente, se ha hecho un resumen de la estructura del proyecto. El segundo capítulo es una exposición de conceptos teóricos, cuya comprensión resulta necesaria e imprescindible en el ámbito de este proyecto.

Los párrafos siguientes son una introducción teórica necesaria para la comprensión del proyecto.

2.2 Objetivos de riego

El diseño de sistemas de riego localizado ha resultado ser una disciplina compleja, llena de conceptos teóricos y con múltiples enfoques. La investigación en este campo ha pasado de la consulta online en blogs de jardinería a la lectura de documentos técnicos, densos y complejos.

El conocimiento adquirido en este punto se refleja en el mecanismo de control de riego del autómata, que ha sido simplificado al máximo, asumiendo constantes en los cálculos, teniendo en cuenta la utilidad de la herramienta y el entorno en el que se desea hacerla útil.

La primera fuente de información sólida y el punto de partida en este apartado fue (MiriadaX, 2018), el MOOC de la Universidad Politécnica de

Madrid: "Diseño agronómico del riego localizado", disponible en la plataforma de e-learning MiriadaX, de la Fundación Telefónica. Es una revelación de la complejidad subyacente al diseño de sistemas de riego.

El temario del MOOC referencia la segunda fuente de información empleada: El cuaderno de trabajo FAO-56 (FAO, 2018), en el que se muestra la forma de calcular las necesidades hídricas de un cultivo en base a la evapotranspiración, es decir, la suma de la evaporación de agua desde el sustrato y la transpiración de vapor de agua llevada a cabo por las hojas de la planta.

El funcionamiento del controlador de riego está basado en el concepto de evapotranspiración de referencia, en adelante ETo, siendo esta un valor complejo, que puede ser calculado mediante el uso de diversos métodos científicos, y que en España suministra el SIAR (Servicio de Información al Regante). Lo que lleva a la tercera fuente de información: (SIAR, 2018). La Figura 1 muestra el concepto de evapotranspiración.

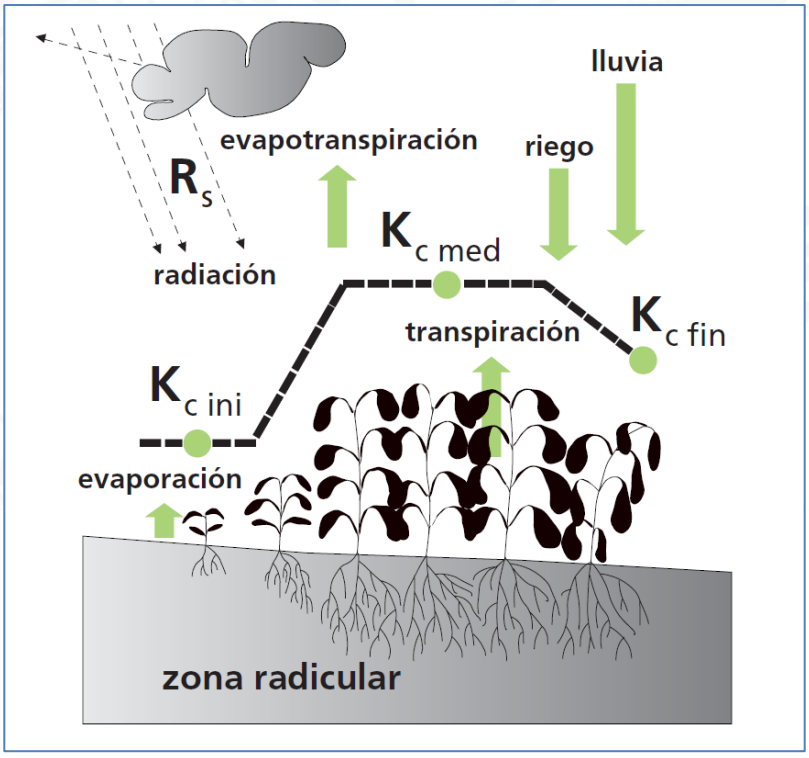

Figura 1- Evapotranspiración. [FAO-56]

Los servicios de información al regante son habituales en España. Se prestan por parte del Ministerio de Agricultura, Pesca y Alimentación, y en ocasiones también a nivel regional.

El sitio web del Ministerio contiene información abundante sobre el riego y sus tecnologías. Expone conceptos teóricos y justifica la forma de obtener la información que suministran las estaciones SIAR. Se trata de aplicaciones prácticas de lo expuesto en (FAO, 2018).

A continuación, se exponen los conceptos y simplificaciones necesarios para comprender el algoritmo de determinación de las necesidades de riego del autómata. Las magnitudes se miden en litros.

- **ETo**: Evapotranspiración de referencia. Se establece en referencia a la definición de Penman, que en 1956 definió la evapotranspiración potencial con respecto a una pradera de pasto bien regada. Obtenemos la **ETo** de los servicios **SIAR**.
- **Kc**: Constante de cultivo. Permite adaptar la ETo a nuestra plantación. Varía en el tiempo junto con las características de nuestra planta.
- **ETc**: Evapotranspiración del cultivo. Se calcula como indica la Ecuación 1.

$$\mathbf{ETc} = \mathbf{ETo} \cdot \mathbf{Kc} \tag{1}$$

- **P**: Precipitación.
- **Δ Hs**: Incremento de la humedad del suelo entre dos riegos.
- **Wc**: Ascenso capilar del agua. Cantidad de agua que sube desde estratos inferiores hasta la zona radicular de la planta. Lo tomaremos como cero.
- **NNr**: Necesidades netas de riego. Que podemos calcular como indica la Ecuación 2.

$$\mathbf{NNr = ETc - P - \Delta Hs - Wc}$$

(2)

Es posible simplificar la ecuación anterior, asumiendo ciclos de riego diarios (**Δ Hs** tenderá a cero), ausencia de lluvias (**P** tenderá a cero) y módulos de plantación alejados de una capa freática desde la que el agua asciende (**Wc** tenderá a cero). La Figura 2 muestra el concepto de constante de cultivo.

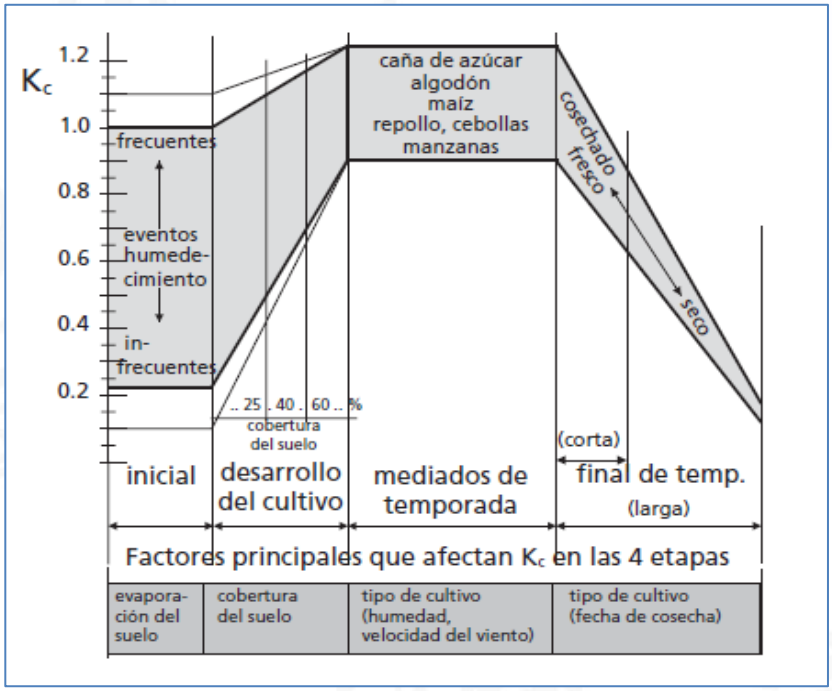

Figura 2 – Cte. De cultivo. [FAO-56]

Si se realiza una monitorización adecuada de los ciclos de riego, las necesidades hídricas pueden aproximarse en base a dos factores:

- **Km** = Constante del módulo.
- **Fc** = Factor de corrección por encharcamiento.

Lo que deja la ecuación de la forma que aparece en la Ecuación 3:

$$\mathbf{NNr = ETo \cdot Km \cdot Fc}$$

(3)

Los factores serán actualizados tras cada ciclo de riego, dando lugar a un ajuste dinámico del autómata.

Finalmente, las necesidades de riego deben ajustarse al tipo de sustrato del contenedor de cultivo. Se ha simplificado la clasificación, aceptando cuatro valores posibles: gravoso, arenoso, franco y arcilloso. Cuanto menor es el tamaño del grano, mayor es la capacidad de retención de agua del sustrato.

La especificación del tipo de suelo permite determinar la eficiencia de riego:

Profundidad Radicular	Gravas	Arenoso	Franco	Arcilloso
< 75 cm	0,85	0,9	0,95	0,95
75 -100 cm	0,9	0,9	0,95	1
> 150 cm	0,95	0,95	1	1

Tabla 1 – Eficiencia de riego. [FAO-56]

Lo que permite determinar el tiempo de riego de la forma mostrada en la Ecuación 4:

$$\mathbf{Tr = \frac{NNr \cdot Intervalo \cdot Sup}{q \cdot Ne \cdot eficiencia}}$$

(4)

- **Ne**: Número de emisores. Emisores por planta.
- **q**: Caudal de riego del emisor. Litros/Hora.
- **Tr**: Tiempo de riego. Horas.
- **NNr**: Necesidades Netas de riego. Litros/m2.
- **Sup**: Superficie en m2.

En un huerto se emplean ciclos diarios de riego. Para evaluar el ciclo de riego, se establecen unos valores de control en base al tipo de suelo del contenedor. Conceptualmente, éstos valores se definen como:

- **Saturación**. En este punto, el sustrato contiene demasiada agua. No llega oxígeno a las raíces y la planta corre peligro por exceso de riego.
- **Capacidad de campo**. Es la capacidad máxima de retención de agua del sustrato.
- **Punto de riego**. Representa el nivel de humedad bajo el que la planta comienza a sufrir estrés hídrico.
- **Punto de marchitez permanente**. Si se llega a este punto, la planta será incapaz de absorber agua, aunque volvamos a regarla.

La Figura 3 muestra los conceptos expuestos.

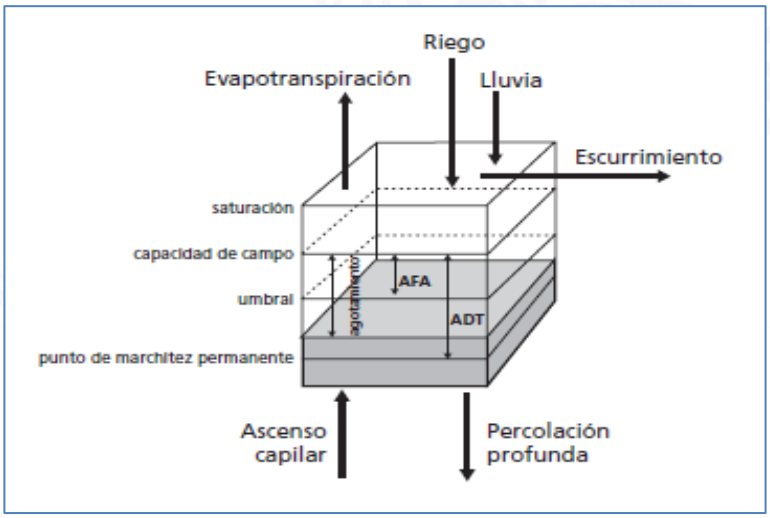

Figura 3 – El agua en el suelo. [FAO-56]

Los valores de los cuatro parámetros anteriores dependen del tipo de suelo y están prefijados en el código del autómata. Después de cada ciclo de riego, se deja reposar el sustrato y se evalúan la humedad inicial y final para actualizar nuestra constante de riego. El factor de corrección por encharcamiento se calcula en base a una medición de humedad efectuada tras la fase de reposo de riego.

En este apartado se han expuesto los conceptos necesarios para comprender la lógica del controlador de riego.

2.3 Componentes y servicios IoT

Scott Fitzgerald y Michael Shiloh (2013), un texto que acompaña al kit de iniciación de Arduino aporta el conocimiento básico necesario para empezar a trabajar con un microcontrolador. Supone una buena iniciación a la programación de estos dispositivos, a la creación de prototipos basados en sensores y actuadores y al diseño de circuitos electrónicos. No obstante, el alcance de este proyecto excede el ámbito del libro.

(Google developers, 2019) alberga una sección completa dedicada al desarrollo de proyectos IoT. La documentación disponible ofrece información muy valiosa para la comprensión de lo que se entiende como IoT. Para empezar, divide los componentes de un servicio IoT en tres grupos:

La Figura 4 muestra los tipos de dispositivo.

Figura 4 – Tipos de dispositivos IoT.[developers.google.com]

Dispositivos: Conjunto hardware y software capaz de procesar e intercambiar información mediante algún mecanismo de comunicación. La información manejada puede a su vez clasificarse en:

- Metadatos asociados al dispositivo.
- Información de estado del dispositivo.
- Telemetría: Información del entorno, captada por los sensores.
- Comandos: Enviados al dispositivo para que realice alguna acción.

Los dispositivos utilizan periféricos. Los periféricos se conectan al procesador de la placa mediante una interfaz de comunicaciones. Las más habituales son:

- **USB**: Bus serie universal.
- **GPIO**: Pins directamente conectados al procesador. Permiten al desarrollador establecer sus propios mecanismos de comunicación.
- **I2C**: Bus serie integrado.
- **SPI**: Bus serie basado en una arquitectura maestro/esclavo.
- **UART**: Permite cambiar datos serie a paralelo. Suele utilizarse para almacenar en memoria (de forma paralela) información transmitida mediante alguna interfaz serial.

Abstracciones hw/sw. Los periféricos necesitan software para ser utilizados. Los fabricantes de dispositivos liberan controladores en forma de librerías que podemos incluir en nuestros proyectos.

Procesamiento de información. Se realiza en el dispositivo, de forma previa a su transmisión y/o almacenamiento. Esto se muestra en la Figura 5.

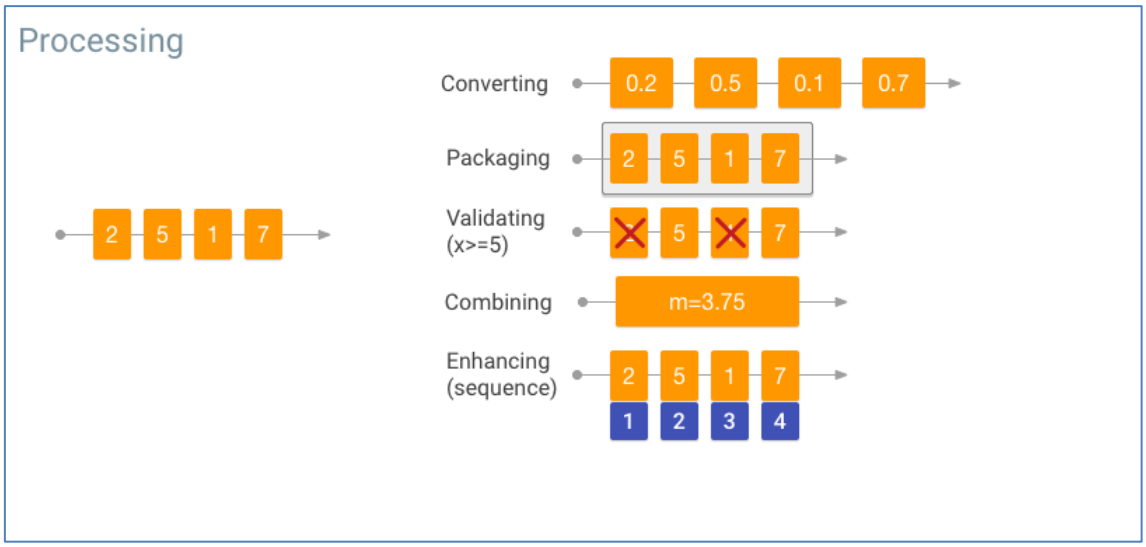

Figura 5 – Proceso de información IoT. [developers.google.com]

Gateways. En numerosas ocasiones, los dispositivos IoT carecen de la implementación de una pila de protocolos de red, así como del hardware necesario para acceder a Internet. Un gateway se encarga de recibir la información del dispositivo y transmitirla a Internet. Puede ofrecer funciones intermedias de proceso y visualización de datos. Puede actuar también como una fuente de tiempo fiable para los sensores, etc.

Plataforma IoT: Ofrece servicios asociados a los dispositivos. Destacan:

1. Administración de dispositivos.
2. Aprovisionamiento: Set Up inicial.
    1. Bootstrapping.
    2. Credenciales para realizar comunicaciones seguras.
    3. Autorización del dispositivo (basada en las credenciales).
    4. Set Up de red.
    5. Registro y monitorización del dispositivo.
3. Operaciones: Podemos llevar un seguimiento mediante logs de la información de estado (interno) del dispositivo.
4. OTA Updates. Actualizaciones “Over the air”.

La Figura 6 es una imagen representativa de lo expuesto hasta ahora:

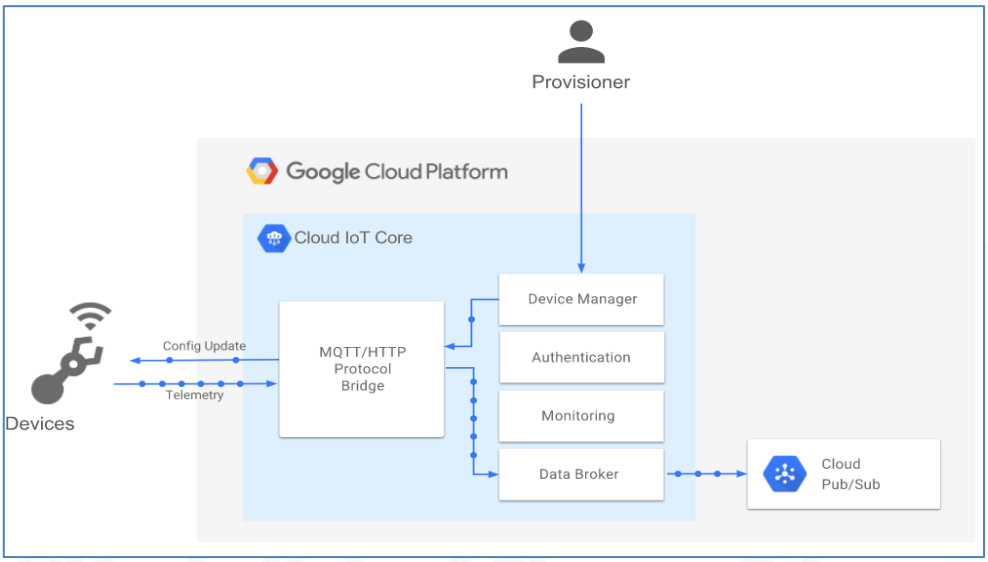

Figura 6 - Plataforma IoT. [developers.google.com]

(Google developers, 2019) es, sin duda, una fuente de información valiosa.

2.4 Programación de microcontroladores

El lenguaje con mayor adopción en la programación de microcontroladores es C, por lo que es recomendable la lectura de: W.Kernighan Brian y M.Ritchie Dennis (1991). Dependiendo de la plataforma, se dispondrá de herramientas de programación específicas y dedicadas. La programación de microcontroladores sigue unas reglas básicas. Muchos son compatibles con Arduino y el firmware desarrollado para una placa es fácilmente portado a la otra.

Hay dos bloques de código destacados: Los métodos setup() y loop(). El primero de ellos se ejecuta al iniciar el dispositivo y es utilizado para cargar librerías, inicializar objetos, controladores y todos aquellos servicios incluidos en la plataforma. El segundo se ejecuta de forma

indefinida mientras el dispositivo permanece encendido. Ahí es donde se define el comportamiento del dispositivo. Una aproximación básica puede extraerse en (Massimo Banzi, 2012).

Los controladores se publican como bibliotecas, que deben ser incluidas en el proyecto, accediendo a su funcionalidad mediante la declaración del correspondiente objeto. Sin embargo, el código del archivo principal tiende a ser lineal, ya que los dispositivos poseen unas capacidades limitadas y prima el ahorro de recursos hardware.

2.5 Comunicación entre sistemas

2.5.1 Protocolo de comunicación: HTTP

La transmisión de datos y peticiones del autómata emplea diferentes enfoques; no obstante, todos utilizan el protocolo de transferencia de hipertexto. Ideado por Tim Berners-Lee en 1991, HTTP es un protocolo de aplicación, inicialmente creado para la transmisión de documentos de hipertexto entre navegadores cliente y servidores web. Queda resumido en (Liu M.L., 2004):

1. Tiene una arquitectura cliente-servidor en la que no se mantiene el estado de la comunicación.
2. Los mensajes se transmiten en texto plano y el conjunto de métodos (verbos) admitidos por el protocolo es bien conocido.
3. Emplea un mecanismo de petición-respuesta.

La Figuras 7 y 8 ilustran la comunicación HTTP.

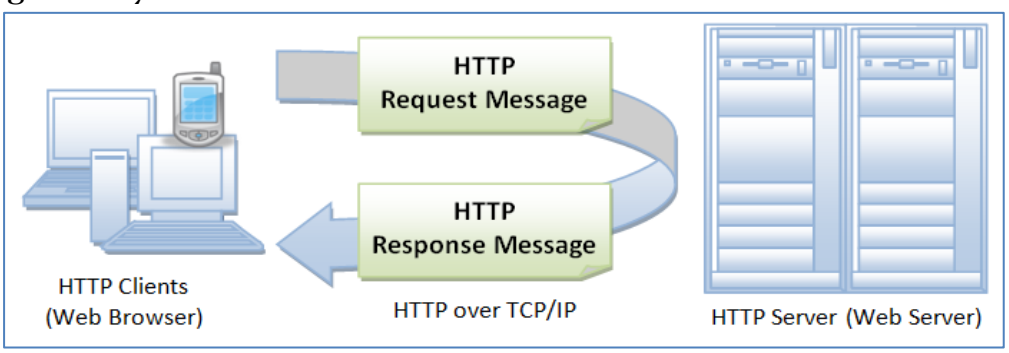

Figura 7 – Comunicación HTTP. [wikipedia.org]

La estructura de una petición HTTP es:

- Línea de petición: Especifica el método, la URL y la versión del protocolo.
- Cabeceras: Pueden interpretarse como parámetros de la llamada.
- Línea en blanco.
- Cuerpo de la petición: Datos transmitidos.

La estructura de una respuesta HTTP es:

- Línea de estado: Especifica el estado del recurso y una descripción.
- Cabeceras: Pueden interpretarse como parámetros de la respuesta.
- Línea en blanco.
- Cuerpo de la respuesta: Datos transmitidos.

2.5.2 Arquitectura del API: REST

La transferencia de estado representacional es una arquitectura de sistemas hipermedia distribuidos construida sobre HTTP y las extensiones MIME para el intercambio de archivos de datos. (Martin Kalin, 2013) ofrece una exposición práctica, basada en una implementación JAVA, de esta arquitectura.

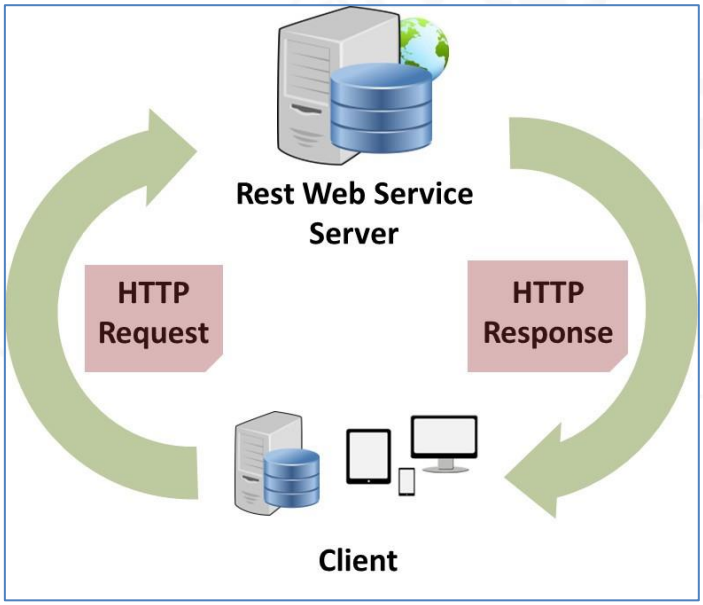

Figura 8 – Interacción REST. [wikipedia.org]

REST permite definir interfaces (APIs) entre sistemas por medio de la exposición de recursos, y la definición de los métodos que se pueden ejecutar sobre ellos. La interacción se lleva a cabo mediante el intercambio de peticiones y respuestas HTTP.

- Accedemos a la representación del recurso especificado en la URI.
- El verbo de la petición determina la acción a realizar sobre el recurso.
- Los nombres son opacos: dos URIs distintas no guardan relación entre ellas.
- Los recursos se codifican mediante la especificación MIME.

Las operaciones CRUD sobre un recurso y sus verbos equivalentes son:

1. Create -→ POST.
2. Read ---→ GET.
3. Update → PUT.
4. Delete → DELETE.

Las operaciones de descarga y obtención de información del autómata son llevadas a cabo mediante la interacción con interfaces de programación de terceros con arquitectura REST.

2.5.3 Codificación de los mensajes: JSON

Los conceptos expuestos en esta parte pueden explorarse en los distintos cursos disponibles en: (Udacity, 2018).

Javascript Object Notation es un formato de archivo estándar, que emplea una estructura legible basada en texto para almacenar objetos de datos, codificados como:

1. Pares “atributo:valor”.
2. Datos de tipo array.

Ligero, ideal para el intercambio de datos, es independiente del lenguaje de programación y se ha convertido en la opción por defecto para el intercambio asíncrono de datos entre sistemas, sustituyendo a XML en numerosos entornos AJAX. Los lenguajes de programación modernos

incluyen mecanismos de codificación y descodificación de objetos JSON, facilitando el tratamiento de los datos en los extremos.

Los datos intercambiados por el sistema de comunicaciones están codificados como objetos JSON.

2.5.4 Autenticación de peticiones

Los medios de autorización y autenticación de peticiones REST son muchos. Para la construcción del sistema es relevante la comprensión de algunos conceptos.

JWT

JSON Web Token es un protocolo de autenticación que define el conjunto de operaciones necesarias para emitir y validar tokens firmados. El estándar define la estructura interna que contiene el token. Los tokens JWT son utilizados en OAuth 2.0, aunque no son los únicos.

API Keys

Son piezas de código (p.ej.: una cadena de caracteres) que se asignan al consumidor de una API y que se emplean en cada petición del usuario. Pueden utilizarse como método de autenticación del usuario, ya que son únicas. Un caso de uso adecuado para este mecanismo sería entre aplicaciones internas y si es posible, sólo para funciones de consulta.

OAuth 2.0

Se trata de un framework de autorización que permite el acceso limitado (por scopes) a los datos de un usuario por parte de aplicaciones de terceros, sin que éstas conozcan las credenciales del usuario.

Podemos establecer una analogía entre OAuth 2.0 y la tarjeta de un hotel:

1. Existe un mecanismo previo de autenticación que tiene como resultado la generación de un token de acceso asociado a la cuenta de usuario (la tarjeta).
2. El token se utiliza para acceder a las partes del hotel (o de la API) a las que se tiene permiso de acceso en base a la autenticación previa.

Sin embargo, OAuth 2.0 es un mecanismo de autorización, no de autenticación. Se trata de obtener un token para garantizar el acceso a un recurso. Los tokens OAuth 2.0 son opacos y no contienen información sobre el usuario o su identidad.

Conceptos OAuth 2.0:

1. **Actores:** Propietario, aplicación cliente, servidor de recursos y servidor de autorización.
2. **Scopes**: Identificadores que permiten determinar los recursos a los que se concede acceso.
3. **Tokens e identificadores**:
    a. **Client ID**: Identifica a la aplicación cliente en el servidor de autenticación.
    b. **Client Secret**: Clave secreta que pertenece a la aplicación.
    c. **Access Token**: Claves proporcionadas al cliente por parte del servidor de autorización. Permiten acceder a las APIs (scope determinado) por un tiempo limitado.
    d. **Refresh Token**: Clave proporcionada por el servidor de autorización que permite solicitar nuevos access tokens.

Los mecanismos para la obtención del token de acceso reciben el nombre de flujos de autorización o Grant Types. Dependiendo del grado de confianza en la aplicación cliente, podemos optar por uno u otro.

Client credentials grant type. El access token se genera autenticando únicamente a la aplicación cliente, no al usuario. La aplicación actúa en nombre propio. La Figura 9 ilustra este mecanismo.

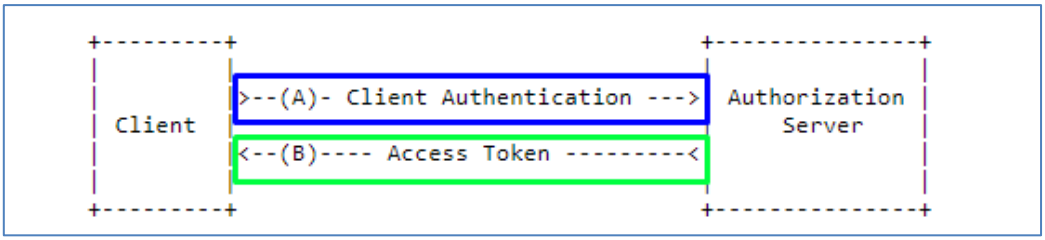

Figura 9 – OAuth2.0: Client credentials grant type. [wikipedia.org]

Se trata de un mecanismo de autorización apto para la comunicación de aplicaciones internas.

Resource owner password grant type. En este flujo de autorización, el usuario proporciona su usuario y contraseña a la aplicación cliente. La aplicación actúa como si fuese el usuario, proporcionando sus credenciales al servidor de autorización, que genera los tokens de acceso y refresco. La Figura 10 ilustra este mecanismo.

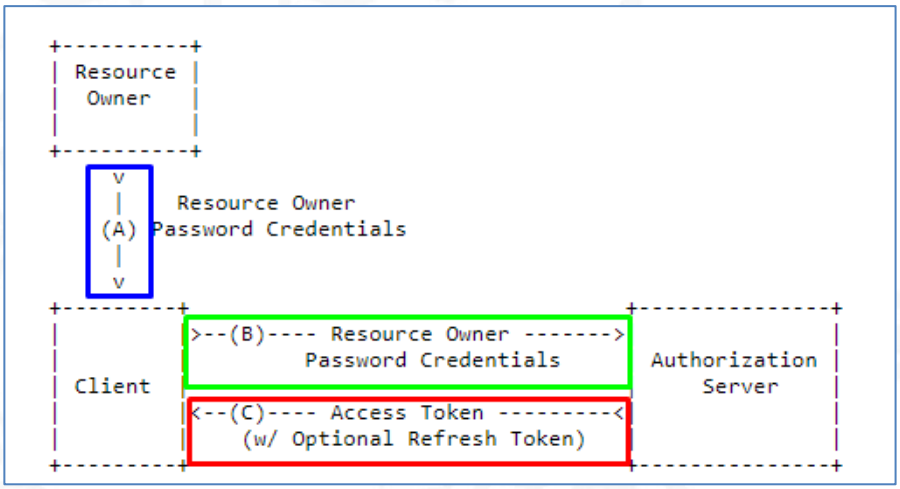

Figura 10 – OAuth2.0: Resource owner password grant type. [Wikipedia.org]

Authorization code grant type. El usuario proporciona sus credenciales al servidor de autenticación, que genera un código para que la aplicación cliente pueda obtener un access token. De esta forma, la aplicación cliente no conoce las credenciales del usuario. La Figura 11 ilustra este mecanismo.

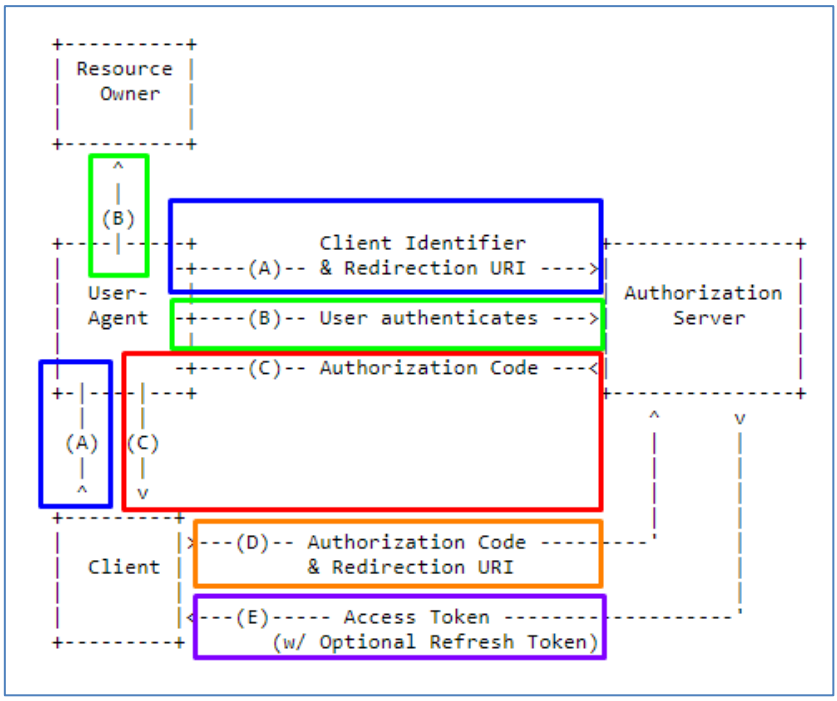

Figura 11 – OAuth2.0: Authorization code grant type. [wikipedia.org]

Implicit grant type. Es una simplificación del flujo anterior: el servidor de autenticación no genera un código de autorización; sino que genera directamente el access token. Recibe el nombre de implícito porque toda la comunicación reside en el navegador, es decir, no hay un servidor de backend. No hay comunicación entre la aplicación cliente y el servidor de autorización. La Figura 12 ilustra este mecanismo.

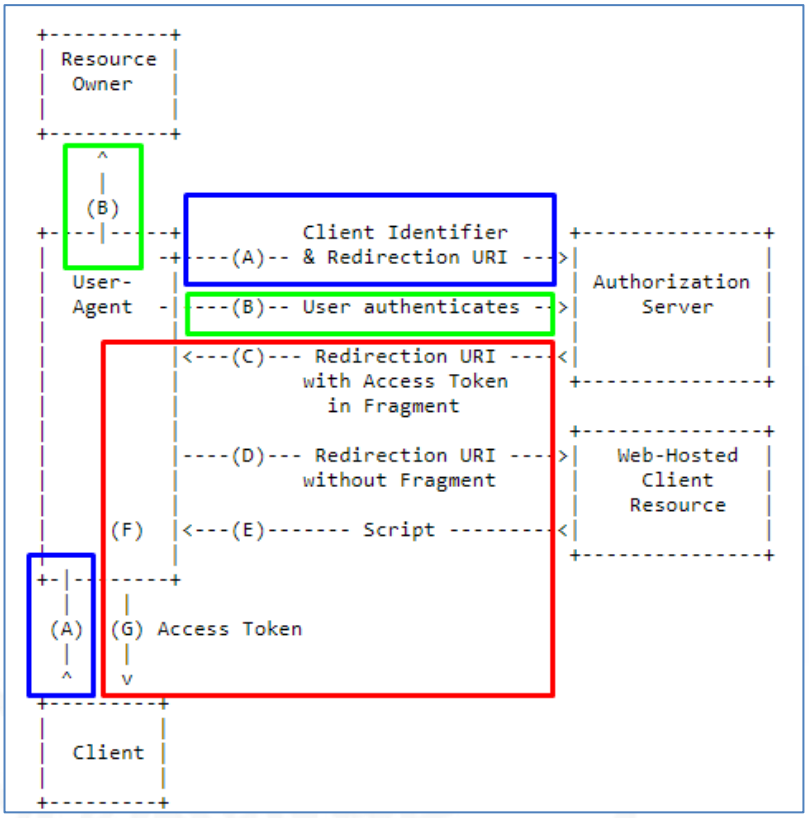

Figura 12 – OAuth2.0: Implicit grant type. [wikipedia.org]

2.5.5 Webhooks

(Rick As, 2018) expone con claridad este mecanismo y su implementación en la plataforma de Particle.io. Webhook es un concepto que está ganando popularidad en la actualidad. Se trata de una forma sencilla de crear interacciones entre aplicaciones basadas en la ocurrencia de eventos. Son también conocidos como callbacks HTTP definidos por el usuario.

Un webhook permite a una aplicación proporcionar información en tiempo real sobre la ocurrencia de un evento a otras aplicaciones o sistemas. En el momento en que un evento ocurre, el webhook hace llegar la información a la URL configurada mediante una llamada HTTP, que habitualmente será POST, pero no obligatoriamente. La llamada podrá tener una respuesta o no.

La orientación al evento hace de los webhooks un mecanismo de comunicación ideal para aplicaciones IoT.

Muchas plataformas modernas ofrecen integración mediante el sistema de webhooks. GitHub, por ejemplo, dispone de un mecanismo de comunicación basado en webhooks: Son lanzados cuando eventos como la publicación de la nueva versión de un archivo ocurren.

La inclusión de servicios de comunicación en una plataforma IoT, como la creación de webhooks, la publicación de eventos (que lanzarán el webhook) y la subscripción a eventos externos (que nos entregarán la respuesta), facilita el desarrollo de proyectos complejos, en los que la comunicación entre dispositivos y sistemas es fundamental.

Se define un webhook mediante su especificación en formato JSON. La plataforma puede ofrecer mecanismos de ayuda para la creación de webhooks. La definición suele llevarse a cabo mediante servicios de consola, llamadas al API o utilizando la línea de comandos CLI de la plataforma que aloja el webhook.

En la creación de un webhook pueden definirse:

- Evento encargado de lanzar el webhook.
- URL destino.
- Método de la llamada.
- Parámetros de la llamada. La codificación de los parámetros puede ser compleja; no obstante, la información a enviar suele ser codificada como parámetros GET, como un formulario, o en formato JSON.
- Información de autenticación.

Es difícil depurar un webhook, dado su carácter asíncrono. Una buena herramienta de depuración es requestbin.

El proyecto realiza un uso intensivo de los webhook como medio de comunicación entre el autómata y servicios externos a la plataforma particle.io.

2.6 Aplicaciones web sin servidor

Técnicamente, el diseño del proyecto impone la ausencia de un servidor en el que alojar el panel de control y el sistema de almacenamiento. La herramienta debe funcionar sin necesidad de llevar a cabo tareas de instalación y mantenimiento de una infraestructura TI subyacente.

Se optó por la utilización de un sistema Cloud en su modalidad PaaS (Plataforma como servicio). PaaS es una categoría de servicios Cloud caracterizada por ofrecer al desarrollador un entorno en el que no tenga que preocuparse de las operaciones relacionadas con la administración y mantenimiento de sistemas de soporte, pudiendo centrarse en el desarrollo y despliegue de su aplicación.

Hay diferentes formas de crear un entorno PaaS, siendo la división más habitual la relacionada con su exposición pública o privada. De este modo, hay servicios PaaS públicos, privados e híbridos.

Los entornos PaaS no solo liberan al desarrollador de las tareas de mantenimiento de infraestructura, sino que manejan de forma flexible la cantidad de recursos que la aplicación necesita. El código es el mismo para un usuario que para un millón, al menos en teoría.

Para el desarrollo del proyecto se optó por hacer uso de una plataforma pública. Las opciones más representativas son Amazon Web Services, Google Cloud Platform y Microsoft Azure. Las tres plataformas presentan un portfolio de soluciones Cloud que exceden el propósito de esta exposición.

Típicamente, un entorno PaaS ofrece servicios de alojamiento de aplicaciones, almacenamiento de datos, autenticación de usuarios,

comunicación entre aplicaciones, monitorización de aplicaciones, herramientas de desarrollo e idealmente, servicios de análisis de datos e integración orientados a aplicaciones IoT. Estas características están presentes en Firebase (Firebase, 2019).

2.7 Interfaces web responsive

Como parte de los requisitos de diseño, el panel de control del sistema debe ejecutarse en un navegador web, sin importar el tamaño de la pantalla del dispositivo.

(Alonso Álvarez García, 2010), (Miguel Ángel Acera García, 2012) y (David Sawyer McFarland 2012) sientan las bases teóricas del diseño de interfaces web interactivas. (W3Schools, 2018) y (Udacity, 2018) son referencias excelentes para adquirir destrezas en la programación JavaScript y sus tecnologías asociadas.

El diseño web adaptable o responsive es una tendencia madura. CSS3 posibilita la adaptación del contenido en pantalla al tamaño de ésta, algo que se consigue mediante correcciones de tamaño y posición de los elementos que forman la interfaz. Estas correcciones se realizan en base a lo que se conoce como media queries: Permiten utilizar puntos de control (o ruptura) en los atributos de la interfaz, que al ser alcanzados modifican el estilo aplicado al elemento. Son el principio básico del diseño web adaptativo.

El desarrollo de un panel de control adaptable puede parecer complejo; pero afortunadamente existen frameworks que aceleran considerablemente el tiempo de entrega y dan homogeneidad al diseño. Uno de los más utilizados y estables es Bootstrap, que va por su cuarta entrega. Bootstrap se centra en el diseño de interfaces web e incluye elementos HTML, CSS, JavaScript y elementos complejos. Presenta un alto grado de compatibilidad con todos los navegadores y se caracteriza por su homogeneidad y vistosidad.

Material Design es un proyecto de Google centrado en establecer una serie de principios de diseño, con el objetivo de mejorar la experiencia de usuario. No se limita al desarrollo de interfaces. Ha sido posible emplear esos principios en la aplicación gracias al framework Material Design for BootStrap, que puede encontrarse en: (MDB, 2018). Combina la vistosidad y usabilidad de Material Design con la sencillez y homogeneidad de Bootstrap en el diseño de interfaces adaptables.

Bootstrap ofrece no sólo elementos gráficos, sino también una serie de animaciones y comportamientos asociados a los componentes. Algo que se consigue gracias el uso de la librería JQuery.

JQuery es una biblioteca JavaScript cuyo objetivo es acelerar el desarrollo de scripts, mejorando la compatibilidad del código entre navegadores web. Simplifica la selección y animación de elementos, ofrece mecanismos de comunicación AJAX y goza de una gran aceptación. Se trata de una biblioteca ampliamente utilizada y forma parte de los dos frameworks anteriormente citados.

Las tecnologías de desarrollo de aplicaciones web gozan de una salud excelente y en la actualidad JQuery, aunque sigue presente en muchos entornos, está cediendo terreno frente al avance de frameworks complejos, que abarcan escenarios más amplios en el desarrollo de aplicaciones, tales como React, Vue o Angular. Más recientemente, frameworks de desarrollo web basados en la utilización de componentes como Polymer parecen marcar la pauta a seguir, más si cabe al desarrollar lo que se conoce como aplicaciones web progresivas (PWA).

La madurez de JavaScript como lenguaje hace posible encontrar librerías para casi cualquier tarea. Algunas de ellas nos permiten incluir gráficos avanzados en nuestras aplicaciones. Destacan Google Charts y Charts.js, pero la lista es larga. Estos dos proyectos se caracterizan por su compatibilidad y adopción. Charts.js se ha utilizado en el desarrollo del panel de control.

2.8 Programación JavaScript en el servidor

En determinadas fases del proyecto ha sido necesario trasladar la carga de trabajo a funciones de backend. En un entorno sin servidor, esto se consigue haciendo uso de servicios PaaS y FaaS *(Functions as a Service)*.

Node.js es código JavaScript que se ejecuta en el lado del servidor. La exposición de su utilidad y funcionalidad excede el contenido de este texto. Firebase Cloud Functions es un producto que permite alojar piezas de código en un entorno sin servidor. Esas piezas de código se ejecutan bajo ciertas circunstancias, realizan su tarea y finalizan.

El sistema hace uso de funciones sin servidor para comunicar información entre plataformas. Son ejemplos prácticos de cómo está cambiando la forma de diseñar, desarrollar y entregar soluciones web, en las que la interacción con sistemas externos se produce por medio de soluciones estandarizadas o mayoritariamente soportadas.

2.9 Conceptos de trabajo

2.9.1 Selección de plataforma IoT

Sin duda la elección más importante. Se han valorado dos alternativas basadas en arquitecturas y características muy distintas.

Arduino surgió en 2005 como resultado de un proyecto de carácter educativo que trataba de crear una placa de bajo coste, mediante la cual los estudiantes pudiesen realizar proyectos de programación e interacción con el mundo real por medio de sensores y actuadores. Su éxito ha generado una lista enorme de proyectos similares: Placas, microcontroladores y sistemas programables que ofrecen características particulares y se adaptan a entornos concretos. Servicios IoT asociados a esas placas facilitan el desarrollo de proyectos complejos, y las

comunidades de desarrolladores ofrecen inspiración y ayuda. Una revolución en toda regla.

2.9.1.1 Primera alternativa: seeedstudio.io

Como puede observarse en su página web, (Seeedstudio, 2018), la empresa comercializa una serie de microcontroladores identificados por el nombre WIO, cuya principal característica es la abstracción del hardware. Podemos observar uno de estos dispositivos en la Figura 13.

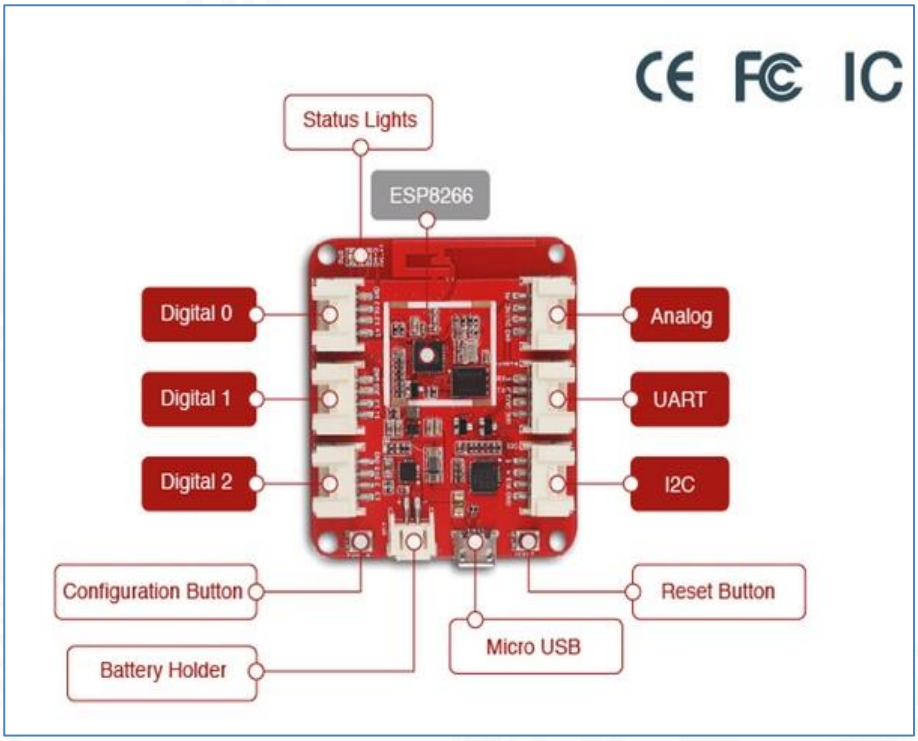

Figura 13 – Microcontrolador IoT. [seeedstudio.com]

Los microcontroladores se conectan a sensores y actuadores GROVE sin necesidad de ser soldados. GROVE es un sistema modular de prototipado rápido, formado por hardware y cables de conexión, basado en zócalos. Cada sensor viene con instrucciones de conexión y librerías de código controlador. Hay cuatro tipos de interfaces GROVE: Digital, Analógica, UART e I2C.

Cumple con los requisitos del proyecto, es compatible con numerosas plataformas y fácil de conseguir.

El conjunto se configura mediante una App móvil que permite asociar gráficamente elementos GROVE a los zócalos conectores de la placa. La misma App sirve para conectar el dispositivo a Internet a través de una red Wifi. El hardware se abstrae y el acceso a las funciones del dispositivo se realiza mediante un API REST. Pasando como parámetros un token de autenticación y el identificador del sensor que deseamos consultar, obtenemos como respuesta un mensaje en formato JSON con los datos del sensor. La interacción se produce a través de servicios Cloud ofrecidos en: **iot.seeed.cc**. La Figura 14 ilustra este concepto.

No es posible programar el dispositivo. La interacción se limita a la lectura de los sensores conectados y a la activación/desactivación de los actuadores. Ofrece, además, integración con la plataforma IFTTT para la integración con otros sistemas.

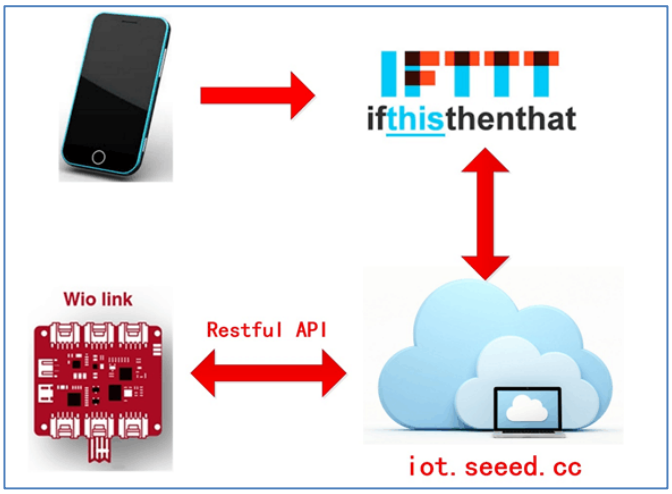

Figura 14 – Plataforma IoT. [seeedstudio.com]

Esta solución parece completa: microcontrolador + sensores/actuadores + servicios de comunicación Cloud IoT + integración con otros sistemas.

No obstante, la simpleza excesiva de la plataforma podría condicionar el desarrollo de un proyecto. Al no poder hacer nada con el microcontrolador, todo el comportamiento del sistema debe ser programado y ejecutado en un servidor o en un entorno PaaS. Sirve como ejemplo; pero no es una plataforma completa de servicios IoT.

2.9.1.2 Segunda alternativa: particle.io

Sistemas hardware bien pensados, mecanismos de conexión a red integrados en los dispositivos, su propio DeviceOS empotrado y una cuidada selección de servicios Cloud son su seña de identidad.

Como puede verse en (Particle, 2019), la plataforma ofrece al desarrollador las herramientas necesarias para ejecutar cualquier tipo de proyecto:

- Servicios de consola: Registro de dispositivos, logs, OTA Updates, diagnóstico, etc.
- Herramientas: SDKs y CLI.
- Bibliotecas de código certificadas por particle.io.
- RESTful API.
- Entornos de desarrollo: Web IDE, Workbench (extensiones de VS-Code) y más recientemente un editor gráfico basado en Node-Red, llamado IoT Rules Engine.

Si esto no fuese suficiente, la documentación disponible, la comunidad y el servicio de apoyo al usuario son excelentes.

Particle.io fabrica hardware conectado. Sus dispositivos incluyen un firmware de sistema, denominado DeviceOS**,** que se encarga de abstraer las particularidades hardware y garantizar el acceso a los servicios Cloud de particle.io. Hay dispositivos GSM, Wifi, Mesh y algunos de ellos combinan varias tecnologías de red en una sola placa.

2.9.2 Creación del dispositivo físico

El proyecto se ha desarrollado sobre el hardware de particle.io denominado PHOTON, un dispositivo con conectividad Wifi y un microcontrolador STM32 ARM Cortex M3 integrado. Era la única opción Wifi disponible en el momento en que se definió la lista de componentes hardware. Hoy en día, particle.io ha mejorado los dispositivos Wifi, mediante la inclusión de características mesh y gestión propia de las fuentes de alimentación en una única placa llamada Argon.

La Figura 15 muestra el dispositivo.

Figura 15 – Microcontrolador PHOTON. [particle.io]

El objetivo ha sido crear un sistema portable de monitorización de estado, por lo que al PHOTON se ha añadido un sistema de gestión de energía y una batería de litio. Particle comercializa una placa de expansión llamada PHOTON Power Shield. Integra un controlador MCP73871, lo que permite la alimentación de una unidad PHOTON mediante una batería y/o una fuente de energía alternativa. Además, permite la carga simultánea de la batería.

La Figura 16 muestra el dispositivo.

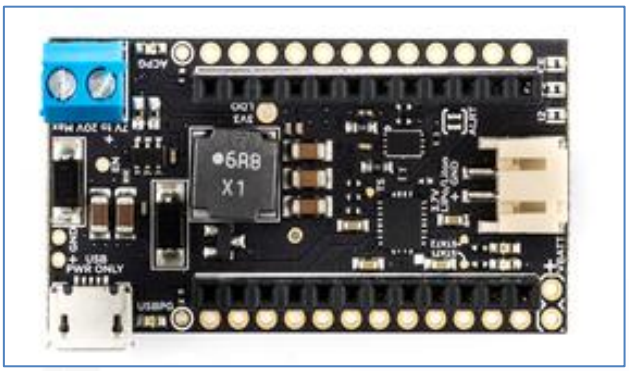

Figura 16 – PHOTON Power Shield. [particle.io]

El bloque central del sistema ha sido completado con productos GROVE: Sensores de temperatura, humedad, luz; un relé para la bomba de riego y una placa de expansión compatible con PHOTON: PHOTON Base Shield. La Figura 17 muestra el dispositivo.

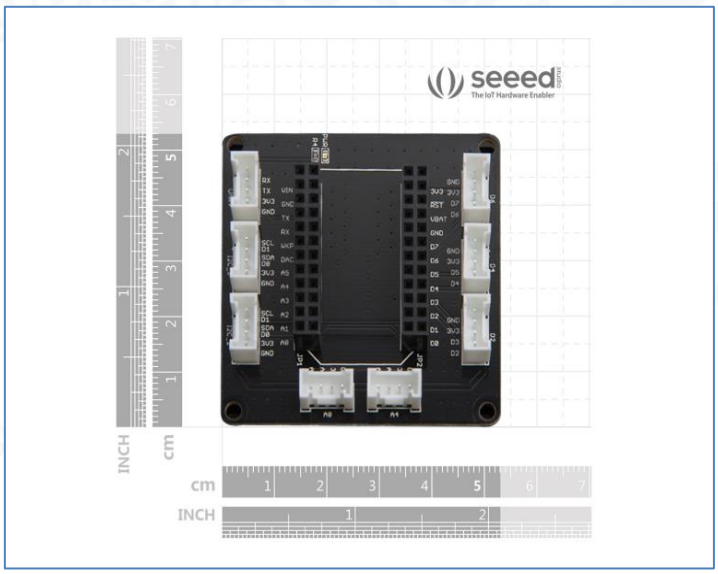

Figura 17 – PHOTON Base Shield. [seeedstudio.com]

El proyecto cuenta con cuatro elementos básicos:

- Placa de desarrollo.
- Breakout board.
- Módulos Grove.
- Cables y conectores.

Hay cuatro tipos de interfaces en GROVE: Digital, Analógica, UART e I2C. La función de cada uno de los cuatro pines del conector según el tipo de interfaz es:

1. Pin 1 [Amarillo] Signal: Digital, Analog, RX on UART, or SCL on I2C
2. Pin 2 [Blanco] Signal: Digital, Analog, TX on UART, or SDA on I2C
3. Pin 3 [Rojo] VCC
4. Pin 4 [Negro] GND

La Figura 18 muestra un kit GROVE.

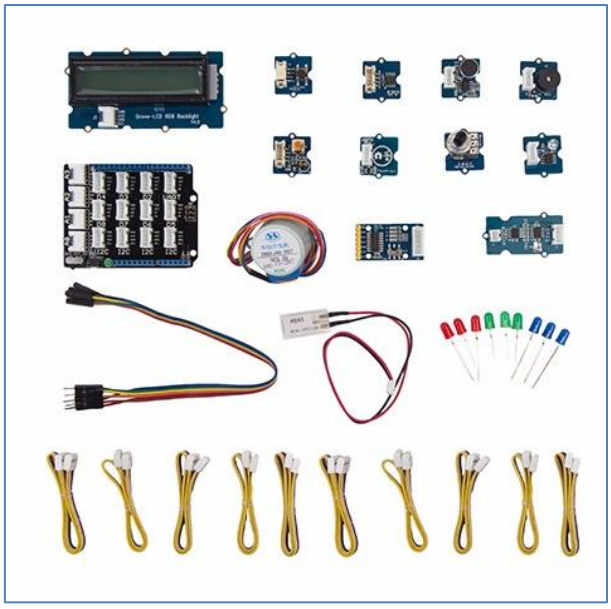

Figura 18 – Sensores IoT GROVE. [seeedstudio.com]

El resto de los elementos necesarios para crear el producto físico, como interruptores, conexiones, carcasa, fuente de alimentación y otros componentes electrónicos han sido adquiridos a través de proveedores de hardware IoT.

2.9.3 Programación del dispositivo

La elección del hardware IoT y su plataforma determinan en gran medida la forma en que se puede programar dicho hardware. Se ha elegido el PHOTON no sólo por su compatibilidad con Arduino y su SDK, sino por el increíble conjunto de herramientas y servicios que Particle pone a disposición del desarrollador.

El PHOTON no se entrega vacío; cuenta con dos secciones de código diferenciadas, que se ejecutan en hilos concurrentes y que alojan sendas aplicaciones:

2.9.3.1 Tinker

Es el firmware de aplicación precargado en todos los dispositivos Particle. No es relevante en nuestro proyecto, por lo que no se analiza en detalle. No obstante, posee una serie de características remarcables:

- Asiste en la configuración inicial del PHOTON y su conexión a una red Wifi.
- Cuando grabamos nuestra primera aplicación lo eliminamos del dispositivo.
- Puede volver a escribirse en el dispositivo en cualquier momento y ser utilizado como base de nuestra aplicación.
- Particle ha diseñado Apps móviles que permiten interactuar con el dispositivo si éste ejecuta Tinker como firmware. Permiten validar el funcionamiento y características del dispositivo antes de empezar a programarlo.

La Figura 19 muestra la interfaz de la aplicación Tinker.

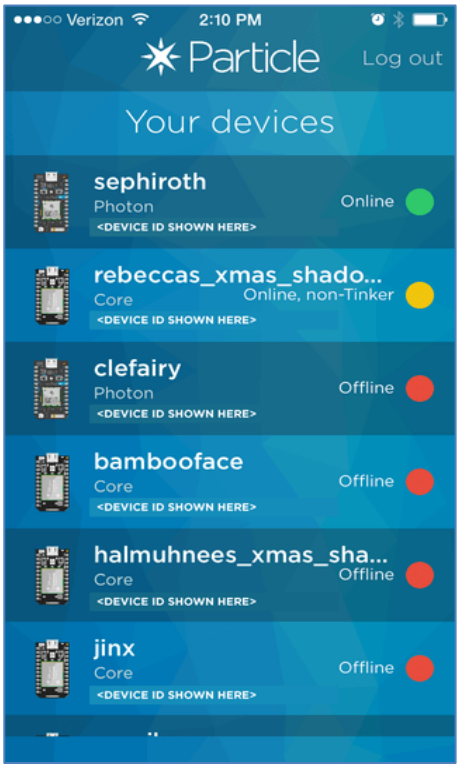

Figura 19 – App de control Particle. [particle.io]

2.9.3.2 DeviceOS

El PHOTON cuenta, de forma natural, con una sección de código o system firmware conocido como DeviceOS, que se ejecuta en un hilo independiente del firmware de aplicación y que nos permite, por medio de un API de sistema, hacer uso de los servicios Cloud de Particle.

El DeviceOS es común a todos los dispositivos Particle, creando una capa de abstracción sobre el hardware subyacente que facilita el desarrollo de programas portables entre dispositivos con características físicas distintas. Ayuda al firmware de aplicación ofreciendo:

1. Comunicaciones seguras con Particle Cloud.
2. Abstracción hardware.
3. Integración con las aplicaciones: mediante una API.
4. OTA updates.

El DeviceOS es mantenido por el equipo de Particle. Periódicamente se liberan nuevas versiones del firmware, algo que es relevante, ya que, al desarrollar una aplicación, se hace para una versión del DeviceOS y ésta será compatible con esa versión del system firmware o una posterior.

La Figura 20 muestra la versión del DeviceOS de un dispositivo en Particle Cloud.

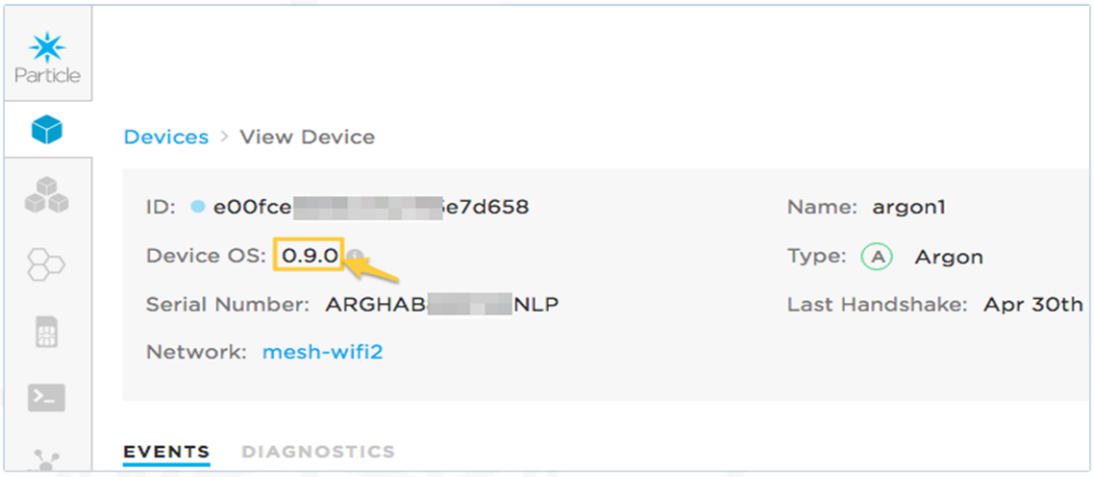

Figura 20 – Consola web Particle. [Particle.io]

En el caso del PHOTON, podemos encontrar todas las características expuestas por el DeviceOS en este enlace:

- *https://docs.particle.io/reference/device-os/firmware/PHOTON/*

No obstante, hay una descripción detallada de las primitivas básicas de comunicación y de los elementos más importantes en el Capítulo dedicado al desarrollo del autómata.

2.9.3.3 Device Cloud

Todos los dispositivos Particle hacen uso de lo que se conoce como Device Cloud: un conjunto de servicios de administración, comunicación, control y actualización automática ofrecidos por la plataforma IoT de Particle. Esta integración nativa permite al desarrollador centrarse desde el primer momento en su aplicación. Se

puede hacer uso de los servicios de la plataforma mediante el API REST, la herramienta CLI, o desde la consola web de Particle.

Los servicios que ofrece la plataforma son:

1. Seguridad de acceso al dispositivo.
2. Registro de dispositivos.
3. Administración de dispositivos.
4. Monitorización de dispositivos.
5. Actualizaciones automáticas del DeviceOS.
6. Actualización remota del firmware de aplicación.
7. Gestión completa de eventos: monitorización, lectura y publicación.
8. Gestión de integraciones: Webhooks Particle e integraciones de terceros.

Cada dispositivo está debidamente identificado mediante un identificador único, y el acceso se controla a través de Tokens de seguridad generados por la plataforma. Si estamos creando un producto, la plataforma implementa OAuth 2.0 como mecanismo de autorización de peticiones al API.

La Figura 21 muestra la pantalla de diagnóstico de un dispositivo conectado a Particle Cloud.

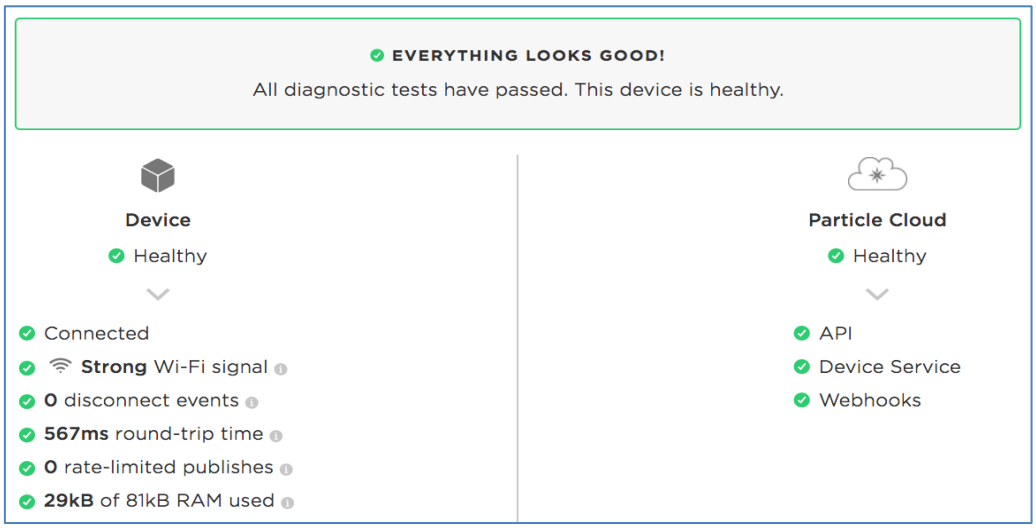

Figura 21 – Control de estado Particle. [particle.io]

Acceso mediante CLI

La forma más sencilla de hacer uso de los servicios Cloud es mediante la línea de comandos. Particle CLI es una herramienta desarrollada haciendo uso de Node.js, que debemos instalar en nuestro sistema y que nos permite principalmente:

- Interactuar con nuestros dispositivos.
- Actualizar remotamente nuestros dispositivos.
- Compilar y grabar nuestro firmware de aplicación en el dispositivo de forma local.
- Trabajar con proyectos locales.

Acceso mediante el API REST

Por medio del API REST, cada dispositivo puede ser utilizado para:

1. GET variables.
2. PUT nuevo firmware.
3. POST una llamada a función.

Protocolo host del API:

- ***https://api.particle.io***

Acepta peticiones en dos formatos:

1. content type Application/json → JSON.
2. content type Application/x-www-form-urlencoded → Form encoded format.

La respuesta siempre tendrá el mismo formato:

1. content type Application/json → JSON.

Acceso al dispositivo:

El acceso al dispositivo es controlado mediante OAuth2. El token podemos enviarlo de tres formas:

1. En una cabecera de autorización HTTP (siempre funciona).

    a. *curl -H "Authorization: Bearer 38bb7b318cc6898c80317decb34525844bc9db55" \ https://...*

2. En la query string de la petición (para peticiones GET)

    a. *curl https://api.particle.io/v1/devices?access_token=38bb7b318cc6898c80317decb34525844bc9db55*

3. En el cuerpo de la petición (funciona con POST, PUT y con DELETE cuando el body es form encoded)

    a. *curl -d access_token=38bb7b318cc6898c80317decb34525844bc9db55 \ https://…*

Acceso mediante la consola

Particle cuenta con una consola web accesible en la URL:

- *https://console.particle.io/*

Desde la consola podemos gestionar nuestros dispositivos, tal y como lo haríamos desde el API:

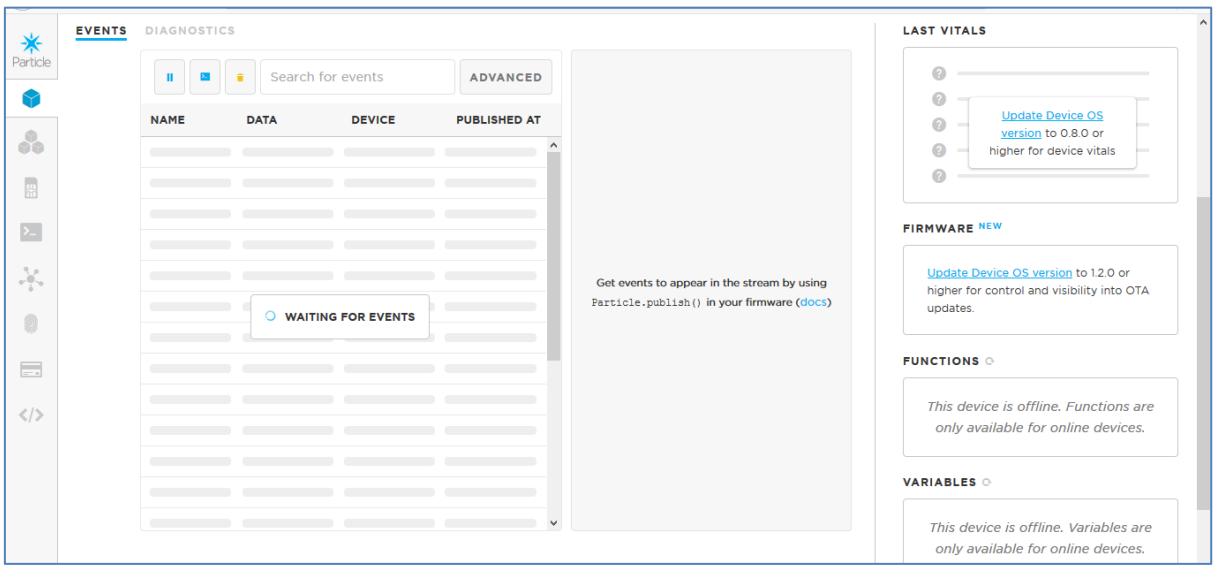

Figura 22 – Eventos en Particle. [Particle.io]

En la Figura 22 es posible observar el estado de conexión del dispositivo, el listado de eventos, las funciones y variables expuestas por el firmware, y un menú lateral que nos da acceso a funciones adicionales.

La consola nos permite controlar, de forma gráfica, aspectos avanzados de la plataforma; como la creación de productos, actualizaciones del firmware, integraciones, diagnóstico remoto, mecanismos de autenticación, etc.

2.9.3.4 Entornos de desarrollo.

Particle cuenta con un set de herramientas inmejorable: cubren de forma excelente cada etapa de desarrollo, abarcando distintos escenarios en cuanto a la tipología del equipo de trabajo y adaptándose a las características del desarrollador.

Como complemento a las herramientas de desarrollo, Particle ha liberado diversos SDK, que podemos emplear en el desarrollo de nuestras aplicaciones web y móviles.

Web IDE

Entorno de desarrollo en la nube, accesible en la URL:

- *https://build.particle.io*

Completamente integrado en Particle Cloud, es la herramienta que se ha empleado para compilar y grabar el código en la unidad PHOTON. Ofrece un grado menor de personalización del entorno. A cambio, garantiza la corrección del firmware y facilita su generación.

Al utilizar el Web IDE:

El preprocesador hará lo necesario para que nuestro código se ajuste al estándar C++ antes de generar el binario que se grabará en el dispositivo.

El preprocesador incluye automáticamente la línea #include “Particle.h”. Incluye también los prototipos de las funciones que definimos en nuestra aplicación. La Figura 23 muestra la interfaz del Web IDE.

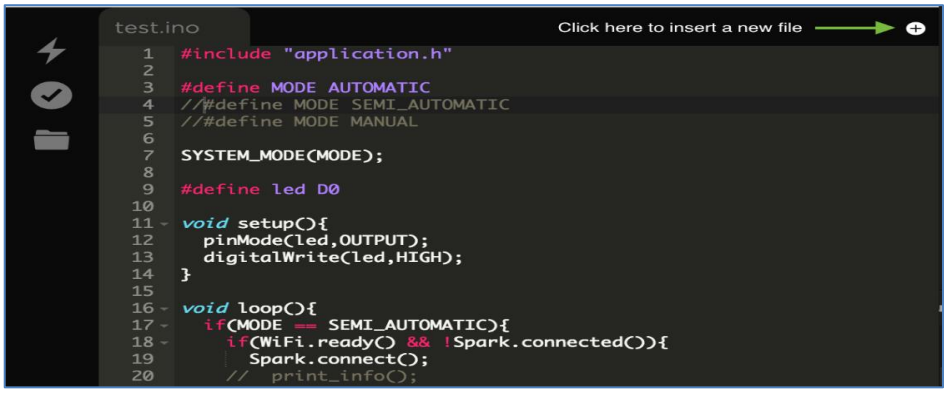

Figura 23 – Particle Web IDE. [Particle.io]

Workbench IDE

Se trata de una extensión de Visual Studio Code, mediante la que podemos desarrollar, compilar y grabar el firmware de aplicación de nuestros dispositivos. Puede trabajar de forma local, o haciendo uso de los servicios de Particle Cloud.

Trabajando en Visual Studio Code, nos beneficiamos de un IDE altamente personalizable y de sus extensiones; además:

- Se instala un módulo de comandos Particle.
- Tenemos acceso a la herramienta CLI desde la consola del IDE.
- Se instalan controladores y diversas versiones del DeviceOS de forma local.
- Podemos depurar el código si disponemos de un debugger físico.
- Tiene integración GIT.

Esta herramienta se ha utilizado para el desarrollo local de código. La Figura 24 muestra la interfaz de la extensión Workbench.

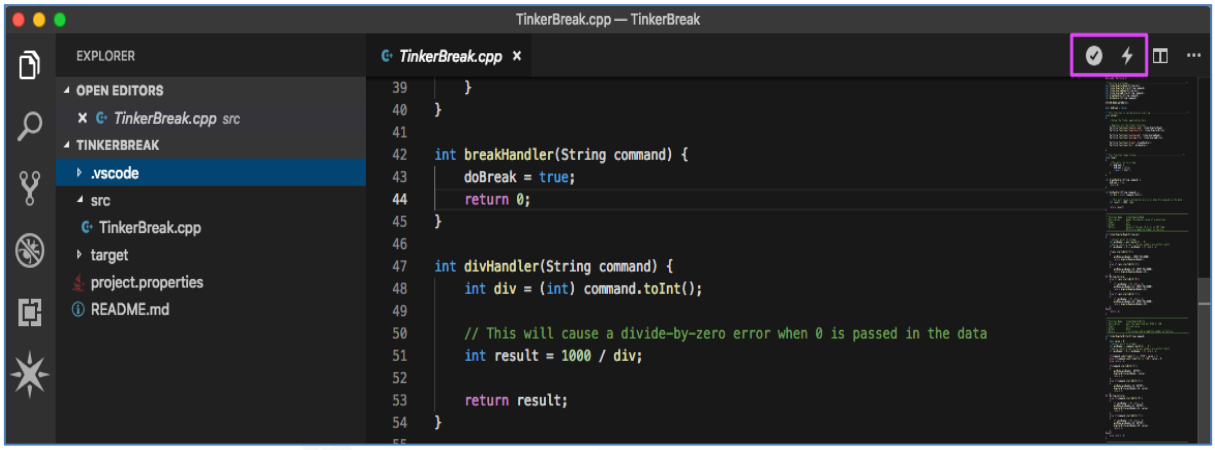

Figura 24 – Particle Workbench extensión. [Particle.io]

2.9.3.5 El bucle de ejecución

Las aplicaciones en Particle pueden ser escritas en C/C++; no obstante, para preservar la compatibilidad con Arduino, es necesario respetar una serie de convenciones:

1. **setup()** → Se ejecuta una vez, al arrancar el sistema.
2. **loop()** → Se ejecuta constantemente.
3. Variables predefinidas:

    - HIGH|LOW
    - INPUT, OUTPUT, INPUT_PULLUP, INPUT_PULLDOWN.
    - true|false.

Es posible hacer uso de todas las funciones y bibliotecas incluidas en el estándar C.

- *https://sourceware.org/newlib/libc.html*

2.9.3.6 Máquinas de estados finitos

Autómata finito o máquina de estados finitos es un modelo de computación ampliamente utilizado en el diseño de programas y circuitos lógicos. Adecuado para el análisis de gramáticas y expresiones regulares, tiene especial relevancia en el ámbito de los procesadores de lenguajes formales.

La abstracción subyacente es la de un autómata que puede encontrarse en un único estado, dentro de un conjunto finito de estados, llamado estado actual. La transición entre estados es provocada por un evento o por el cumplimiento de una condición.

Una máquina de estados finitos particular es definida por:

1. El conjunto de sus estados.
2. El estado inicial.
3. El conjunto de posibles entradas.
4. Las transiciones entre estados.

Como puede observarse en las siguientes figuras, representamos una máquina de estados finitos por medio de su tabla de transición de estados, o mediante el grafo correspondiente:

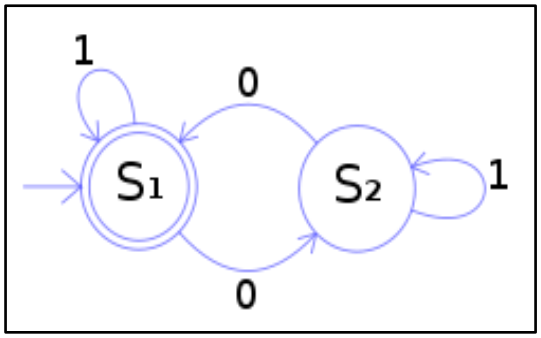

Figura 25 – Diagrama MEF. [wikipedia.org]

salida $q \in Q$	símbolo $\sigma \in \Sigma$	llegada $\delta(q,\sigma) \in Q$
s_1	0	s_2
s_1	1	s_1
s_2	0	s_1
s_2	1	s_2

	0	1
$\rightarrow^* s_1$	s_2	s_1
s_2	s_1	s_2

Figura 26 – Tabla de representación MEF. [wikipedia.org]

Se ha creado un autómata finito determinista, en el que a partir de un estado origen y para una entrada concreta, existe un único estado destino. (Gustavo Gonnet, 2018) Muestra ejemplos de utilización de este tipo de autómatas en el firmware de microcontroladores programables.

El Capítulo 4 muestra el grafo del autómata creado.

2.9.3.7 Funciones de configuración y control

Por medio de los servicios Cloud de Particle, el firmware de aplicación expone una serie de variables y funciones, haciéndolas visibles a través del API REST de la plataforma.

La aplicación web de control emplea esta característica para conocer el estado de las variables y realizar llamadas a las funciones de configuración y control en tiempo real, al margen del ciclo de estados del autómata. La ejecución en hilos separados del firmware de aplicación y el DeviceOS hace posible esta solución.

El listado de variables expuestas para su consulta es:

1. *Particle.variable("Temp_Tierra", tempTierra);*
2. *Particle.variable("Temp_Aire", tempAire);*
3. *Particle.variable("Humedad_Aire", humedadAire);*
4. *Particle.variable("Luz_Visible", luzVisible);*
5. *Particle.variable("Luz_IR", luzIR);*
6. *Particle.variable("Luz_UV", UVIndex);*
7. *Particle.variable("Moisture", moisture);*
8. *Particle.variable("On_Off", mode); // ON - OFF*
9. *Particle.variable("In_Out", modo); // IN - OUT*
10. *Particle.variable("Riego_Fijo", riegoFijo);*
11. *Particle.variable("Tiempo_Riego", tiempoDeRiego);*
12. *Particle.variable("Hora_Riego", horaDeRiego);*
13. *Particle.variable("Moisture_Ini", moistureInicial);*
14. *Particle.variable("Moisture_Fin", moistureFinal);*
15. *Particle.variable("Horas_Luz", horasDeLuz);*
16. *Particle.variable("Estado_AF", estadoSistema);*
17. *Particle.variable("Km", Km); // CTE DEL MODULO*
18. *Particle.variable("Fc", Fc); // FACTOR DE CORRECCIÓN*
19. *Particle.variable("Voltaje", cellVoltage); // ESTADO DE CONEXIÓN*
20. *Particle.variable("Carga", stateOfCharge); // ESTADO DE CARGA*

El listado de funciones expuestas para su invocación remota es:

- *Particle.function("Turn_On_Off", turnOnOff);*
- *Particle.function("Set_Modo", setModo);*

- *Particle.function("Set_Modulo", setModulo);*
- *Particle.function("Get_Umbrales", getUmbrales);*
- *Particle.function("Reset_Riego", resetRiego);*
- *Particle.function("Reset_Estado", resetEstado);*
- *Particle.function("Set_Riego", setRiego);*
- *Particle.function("Set_Rest_T", setRestTime);*
- *Particle.function("Set_Publ_T", setPublishTime);*

2.9.3.8 Memoria EEPROM

El sistema cuenta con un mecanismo de alimentación redundante: externa y batería; no obstante, el dispositivo puede apagarse, de forma accidental o intencionada.

La aplicación web de control permite configurar el dispositivo; pero no sería lógico tener que volver a hacerlo cada vez que se apaga. Además, ciertos valores de la configuración son actualizados constantemente por el autómata, como resultado de la evaluación continua de los ciclos de riego.

El PHOTON posee un módulo de memoria FLASH, en el que emula una EEPROM, que se ha utilizado como mecanismo de almacenamiento de la última configuración buena del autómata. Esta configuración será cargada tras un reinicio, evitando así la necesidad de volver a grabarla desde la aplicación web.

El número de escrituras posibles en la FLASH es limitado, por lo que es necesario cuidar la cantidad de información guardada en cada escritura. Se graban alertas y umbrales de trabajo, la configuración del módulo, y variables de control de funcionamiento: la mínima cantidad de información que permite al autómata continuar con su planificación de riegos.

La estructura de datos grabada en EEPROM es:

```
struct EepromMemoryStructure
{
        uint8_t version = EEPROM_VERSION;      // 1 byte.
        int modo;                              // 4 byte.
        int mode;                              // 4 byte.
        int riegoFijoEEprom;                   // 4 byte.
        double KmEEprom;                       // 8 bytes.
        double FcEEprom;                       // 8 bytes.
        int elSueloEeprom;                     // 4 bytes.
        int qEeprom;                           // 4 bytes.
        int NeEeprom;                          // 4 bytes.
        float m2Eeprom;                        // 4 bytes.
}; EepromMemoryStructure eepromMemory;
```

Junto a esta estructura se almacenan los umbrales de trabajo y el estado de alerta del dispositivo.

```
// Un vector de alertas. 16 bytes.
struct alerta
{
        tipoAlerta nivel;
        elapsedMillis momento;
};
alerta alertas[NUM_SENS];

// Un vector con los umbrales de funcionamiento. 8 bytes.
struct umbral
{
        float max;
```

```
        float min;
};
umbral umbrales[NUM_SENS];
```

2.9.3.9 Librerías de proyecto

Los controladores de dispositivo, la abstracción de la máquina de estados finitos y los contadores de tiempo han sido importados en el proyecto en forma de librerías C++. La plataforma Particle cuenta con un repositorio de librerías organizadas y catalogadas, listas para ser utilizadas. La forma de utilizar las librerías difiere según ha herramienta que se utilice: Web IDE, Desktop IDE o CLI.

Las hay de tres tipos:

1. Públicas.
2. Oficiales.
3. Verificadas.
4. Privadas.

Estructura de una librería en Particle:

- examples // Una carpeta por ejemplo.
    - usage
        - usage.ino
- src
    - mylibrary.cpp
    - mylibrary.h
- library.properties // Información descriptiva y dependencias de la librería.
- README.md
- LICENSE.txt

Estructura de un proyecto en Particle:

Legacy: No soporta el uso de librerías.

- myApp
    - Application.ino

Simple: Las librerías son añadidas al proyecto mediante CLI (library add) o el desktop IDE library manager.

- myApp
  - Application.ino
  - project.properties

Extended: Las librerías son añadidas como en la versión simple y mediante la copia de archivos.

- myApp
  - project.properties
  - src
    - Application.ino

Una de las ventajas de utilizar el Web IDE es que se encarga de crear la estructura de proyecto, importando las librerías y organizando el código. Lo hace de forma transparente, generando el firmware en base a la versión del DeviceOS elegida.

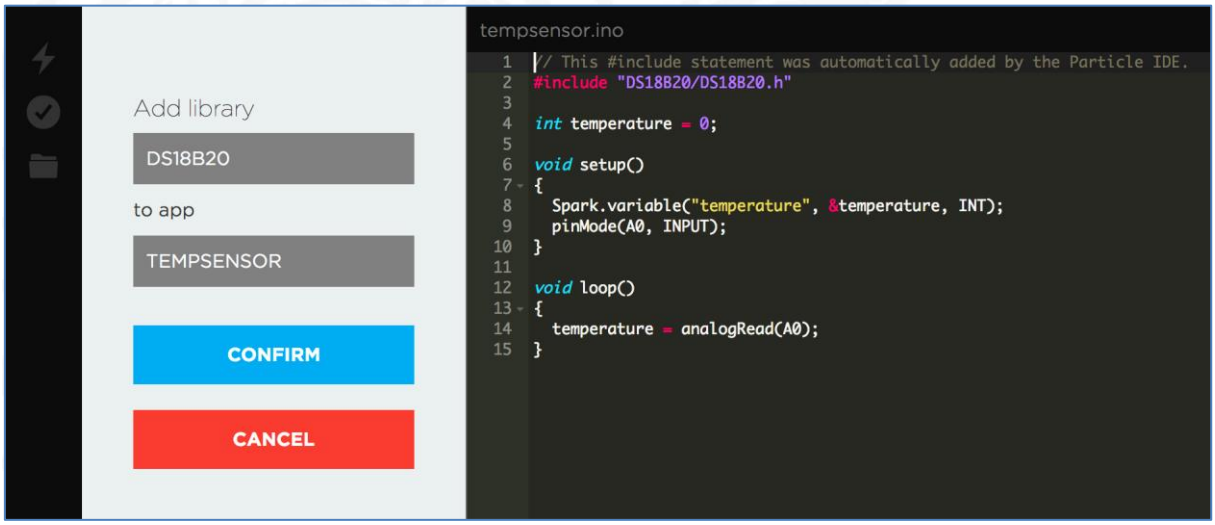

Figura 27 – Añadiendo librerías con el web IDE. [particle.io]

2.9.3.10 Trabajando con el hardware

El acceso al hardware se realiza mediante los controladores de dispositivo importados en el proyecto.

```
#include <Adafruit_DHT_Particle.h>
#include <Adafruit_SI1145.h>
#include <PowerShield.h>

// Inicializamos y localizamos los sensores I2C.

DHT dht(tempHumSensor, DHTTYPE);

Adafruit_SI1145 uv = Adafruit_SI1145();

PowerShield batteryMonitor;
```

2.9.3.11 Acceso a los sensores

Las lecturas se toman mediante la invocación del correspondiente método expuesto por el objeto controlador:

```
// Temperatura del aire
tempAireSamples = dht.getTempCelcius();

// Humedad del aire

humedadAireSamples = dht.getHumidity();

// Luz visible

luzVisibleSamples = (double) uv.readVisible();

// Luz IR

luzIRSamples = (double) uv.readIR();

// Indice UV

UVIndexSamples = ((double) uv.readUV() / 100.0);

// Datos de la bateria

cellVoltage = batteryMonitor.getVCell();

stateOfCharge = batteryMonitor.getSoC();
```

En ocasiones, la electrónica del módulo sensor proporciona un valor concreto que puede ser leído de forma directa en el pin de entrada al que está conectado. En estos casos, no es necesario trabajar con un controlador de dispositivo. Esto sucede con los sensores de humedad en suelo (moisture) y temperatura en suelo. Las lecturas se hacen aquí de forma directa:

```
// moisture
moistureSamples = analogRead(moistureSensor);
// temperatura del suelo
analogValue = analogRead(tempSensor);
```

2.9.3.12 Control del relé

El actuador relé es accionado mediante la escritura en el pin de datos al que está conectado:

```
// Encendido.
digitalWrite(releBomba, HIGH);
// Apagado.
digitalWrite(releBomba, LOW);
```

2.9.4 Selección de plataforma de alojamiento

Para el desarrollar el proyecto se ha empleado una solución PaaS de Google centrada en la creación de aplicaciones web y móviles: Google Cloud Firebase. cuyas particularidades pueden verse en: (Firebase, 2019). Firebase ofrece soluciones al desarrollador centradas en tres ámbitos:

1. Desarrollo de la aplicación.
2. Análisis y mejora continua.
3. Atracción y retención de usuarios.

Para el proyecto, sólo el primer grupo resulta relevante. La plataforma ofrece al desarrollador:

Mecanismos de almacenamiento:

1. **Cloud-Firestore**: Almacenamiento NoSQL con sincronización y soporte al funcionamiento off-line.
2. **Cloud-Storage**: Almacenamiento de objetos. Adecuado para aplicaciones que trabajan con imágenes, vídeo y otro tipo de contenido generado por los usuarios.
3. **Realtime-Database**: Almacenamiento NoSQL con sincronización en tiempo real. Adecuado para el desarrollo de aplicaciones de mensajería entre usuarios, juegos, etc.

Mecanismos de identificación:

1. **Firebase-Auth**: Ofrece diversos métodos para autenticar usuarios de forma segura.

Supresión del servidor:

2. **Hosting**: Alojamiento web estático con funciones avanzadas. Incluye una red de distribución de contenido y certificados SSL. El alojamiento de la aplicación en Cloud Hosting simplifica el proceso de configuración del SDK de Firebase.
3. **Cloud-Functions**: Funciones de backend escritas por nosotros que se activan mediante eventos, por medio de ***webhooks***.

Soluciones avanzadas:

1. **MLearning-Kit**: Permite añadir características machine learning a nuestra aplicación. A través de los modelos predefinidos por la plataforma o mediante la carga de nuestros propios modelos.

2.9.5 Sistema de comunicaciones

Para cumplir los requisitos de funcionamiento establecidos, el autómata debe:

1. Obtener predicciones meteorológicas. AEMET.
2. Obtener la predicción de necesidades hídricas. SIAR.
3. Generar y almacenar información de estado y funcionamiento. FIREBASE.

Haciendo uso del protocolo HTTP, la arquitectura REST y diversos medios de autenticación y autorización, la aplicación web y nuestro autómata intercambian documentos JSON entre sí y con sistemas externos.

2.9.5.1 Llamadas HTTP API REST

Como puede leerse en (Particle, 2019), la comunicación en Particle Cloud se basa en webhooks e integraciones. Los webhooks se construyen sobre la base de una petición HTTP al endpoint correspondiente del API REST. Como ejemplo, este modelo de petición GET a una colección de documentos Firestore, mediante la que es posible leer la información correspondiente al documento:

```
curl -X GET \
https://Firestore.googleapis.com/v1beta1/projects/YOUR_PROJECT_ID/databases/(default)/documents/colection/document \
 -H 'Authorization: Bearer [ACCESS TOKEN]' \
 -H 'cache-control: no-cache'
```

2.9.5.2 Particle WebHooks

Un webhook en Particle Cloud espera a que uno o varios dispositivos publiquen un evento. Cuando el evento es publicado, se lanza una petición HTTP a una URL en la web. La petición incluye información y el método empleado puede ser: POST, GET, PUT o DELETE.

La Figura 28 muestra este concepto.

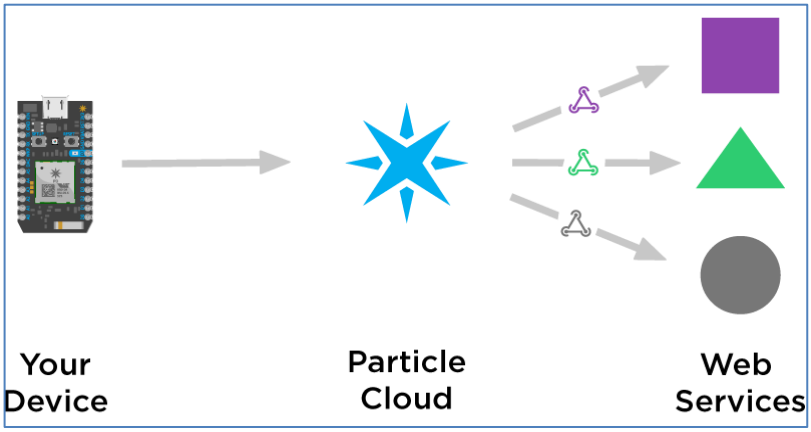

Figura 28 – Webhooks Particle. [Particle.io]

El webhook es activado por todos los eventos cuyo nombre comienza por el prefijo definido en el campo event. A la hora de lanzar un webhook hay que tener claro:

1. Qué eventos vigilar.
2. Desde qué dispositivos.
3. Qué URL debemos alcanzar.
4. Qué datos y de qué forma incluirlos en la URL.

Es posible crear y administrar webhooks mediante la consola web y la herramienta de línea de comandos: CLI. El webhook se crea mediante su especificación en formato JSON, en la que hay una serie de propiedades que pueden definirse.

De forma general, se define:

1. El prefijo de los eventos a vigilar por el webhook.
2. la URL destino.
3. El tipo de petición: GET, POST, PUT o DELETE.
4. Los dispositivos cuyos eventos debemos vigilar.

Es posible añadir información en la URL, en forma de pares clave=valor. Igual que en cualquier petición HTTP.

Para los tipos de petición POST y PUT, es posible además enviar información en el cuerpo de la petición. Esta información podrá ser codificada:

1. En formato JSON. Tipo: "*text/json*".
2. Como datos de formulario. Tipo: "*Application/x-www-form-urlencoded*"
3. De forma libre, haciendo uso del bloque "*body*" en nuestro documento JSON de definición del webhook.

La especificación del webhook permite añadir un campo de autenticación HTTP básica al documento JSON.

Es posible añadir cabeceras HTTP personalizadas en la sección headers del JSON de especificación del webhook.

Finalmente, es posible modificar los eventos activados cuando Particle Cloud obtiene una respuesta a la petición HTTP lanzada a la URL destino del webhook.

Características avanzadas:

La definición de webhooks en Particle permite hacer uso de plantillas mustache, lo que deja hacer transformaciones y procesar el JSON antes y después de ser enviado. Es posible aprovechar esta característica para:

1. Modificar los datos enviados a la URL. De esta forma, se puede publicar un único evento que provoque la ejecución de varios webhooks, cada uno de los cuales alcanzará una URL distinta y enviará su petición adaptada a la URL destino: método, parámetros de petición y cuerpo de petición. Opcionalmente, también se pueden cambiar los nombres de los eventos de respuesta y error para cada petición.
2. Modificar los datos recibidos. De esta forma, se filtra y se adapta la respuesta para adecuarla al manejador. Se asocia cada manejador al nombre de evento de respuesta (o error) que más interese.

La flexibilidad del método expuesto es enorme, pudiendo adaptarse a un elevado número de escenarios de comunicación de forma nativa.

Generalidades:

Si se crea un webhook especificando sólo un nombre de evento, un método y una URL, la información enviada en la petición será:

- ***event***: El nombre del evento que lanza el webhook.
- ***data***: Datos asociados al evento.
- ***published_at***: Momento de la publicación del evento.
- ***coreid***: El ID del dispositivo que lanza el evento.

El envío de estos datos está controlado por la opción *noDefaults*, que por defecto es false.

En algunas ocasiones no será conveniente enviarlos, ya que pueden causar problemas en el servidor objeto de la URL.

Desde la consola es posible vigilar de forma gráfica la evolución de los webhooks. La secuencia de eventos asociada a la ejecución de un webhook es:

1. *"name":"name_of_my_event"*
2. *"hook-sent/name":"name_of_my_event"*
3. *"name":"hook-response/name_of_my_event/0"*

Esta secuencia puede ser personalizada en la especificación JSON del webhook, como se ha indicado antes.

Los posibles errores vendrán en la respuesta. Si hay demasiados, la plataforma detendrá el envío de solicitudes al servidor objeto. No es posible aprovechar eventos de error para lanzar otros webhooks.

Los límites serán los impuestos por el servidor destino de las peticiones URL. En caso de que el servidor destino falle de forma continua, Particle dejará de enviar peticiones durante algún tiempo.

2.9.5.3 Integraciones: Particle & Google Cloud Platform

Los webhooks son un mecanismo flexible y poderoso de comunicación entre sistemas: salvan una gran cantidad de situaciones y son suficientes en muchas de ellas; no obstante, limitaciones como el tamaño máximo de los datos intercambiados entre el firmware y el webhook (hasta 255 caracteres (en versiones hasta la 0.8.0 del DeviceOS), 622 caracteres (a partir de la versión 0.8.0)) dificultan la comunicación con algunos servicios de terceros.

El tamaño del token de autenticación de peticiones al API REST de Firestore excede los límites mencionados. Se podría haber desarrollado un API, implementando funciones personalizadas y desplegándolas en el entorno FaaS de Firebase; pero se ha utilizado una de las integraciones disponibles en el conjunto de herramientas para desarrolladores de Particle.

La integración entre Particle y Google ofrece un mecanismo simple y testado de envío de datos desde los dispositivos Particle hacia los servicios y productos de Google Cloud Platform. Esto se muestra en la Figura 29.

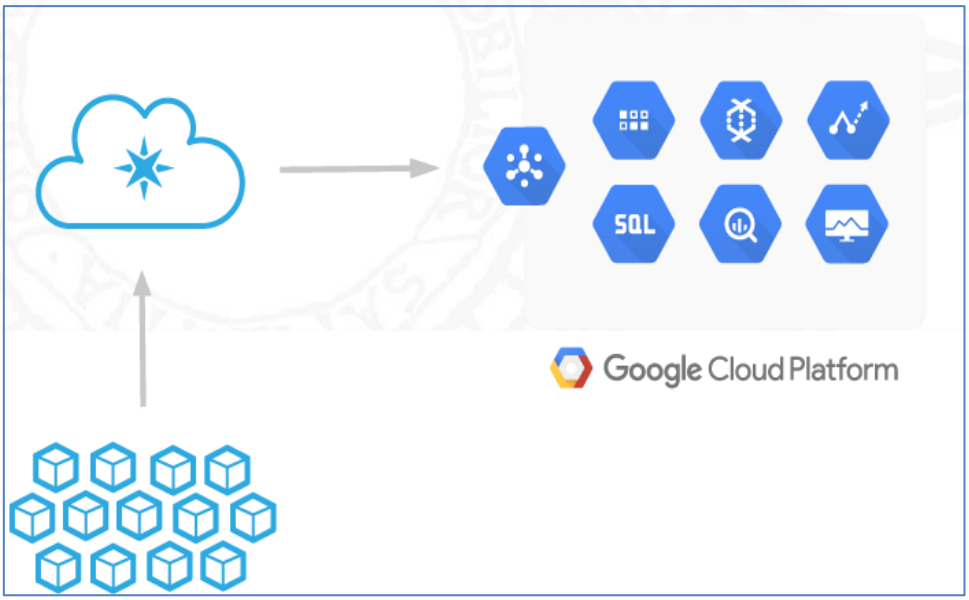

Figura 29 – Integraciones Particle. [Particle.io]

El servicio que recibe los datos de estado es *pub/sub*, un middleware para el intercambio de mensajes basado en eventos. Permite a aplicaciones de distintos orígenes el intercambio asíncrono de mensajes, asegurando los extremos y garantizando la disponibilidad. Conceptos clave:

- **Topic**: Nombre del recurso al que serán enviados los mensajes.
- **Subscription**: Flujo de mensajes de un topic que serán enviados a un suscriptor.
- **Message**: Conjunto de datos y atributos enviados a un topic.
- **Message-attribute**: Par clave-valor que se define para un message.

El flujo de publicación:

De forma previa, se habrá creado un topic y (opcionalmente) una suscripción.

1. La aplicación envía messages al topic.
2. Los messages son retenidos hasta que un suscriptor los reclama.
3. Se crea una función suscriptora.
4. La función suscriptora almacena el payload del message en Firestore.
5. Se confirma la recepción del message para su borrado de la cola.

La Figura 30 muestra el flujo de publicación.

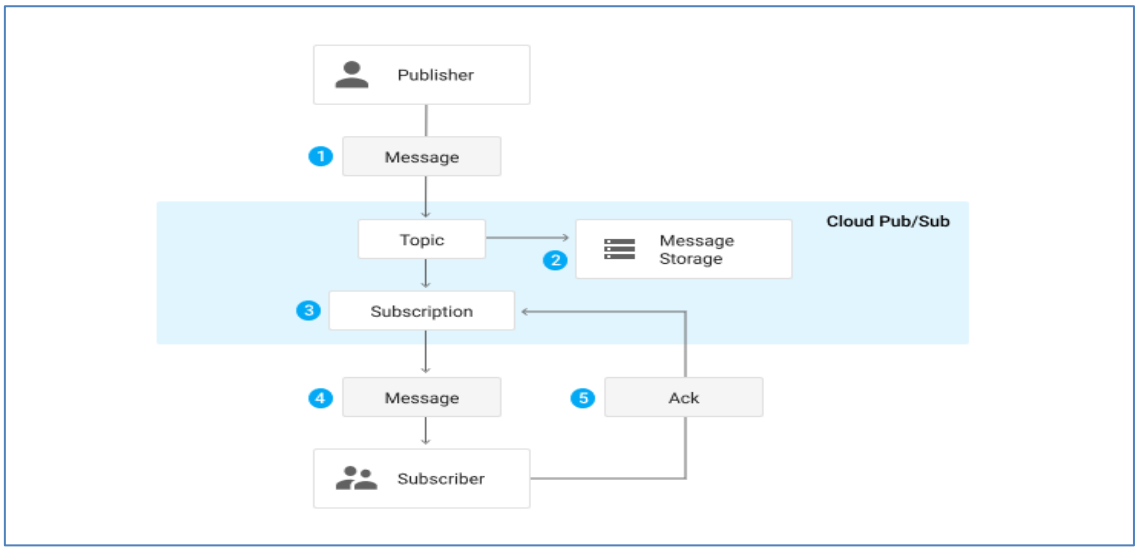

Figura 30 – Flujo de publicacióno Pub/Sub. [developers.google.com]

La integración:

Es necesario crear un proyecto en Google Cloud Platform, al que se añade un *topic pub/sub* sobre el que tendrá permisos de publicación la cuenta de servicio de Particle integrada en Google Cloud: "*particle-public@particle-public.iam.gserviceaccount.com*". Hecho esto, se activa la integración en la consola web de Particle y se le asocia un evento, igual que con los webhooks. Al publicar el evento en el firmware de los dispositivos, se envían datos en un message al topic *pub/sub* creado.

El servicio pub/sub no almacena los datos, sólo los recoge. Para almacenar los datos ha sido necesario desarrollar una función javascript de servidor, alojarla en Firebase y asociarla al evento de publicación de nuestro topic como suscriptora. Se ha repetido el proceso para: hitos, alertas, estados y riegos. Cuatro topics, cuatro funciones suscriptoras.

2.9.6 Diseño e implementación del sistema de almacenamiento

2.9.6.1 Bases de datos NoSQL en tiempo real

Se ha elegido Firebase como plataforma debido a las características de los sistemas almacenamiento que incluye: bases de datos NoSQL en tiempo real, ya que la sincronización entre el autómata y la aplicación de control es esencial en nuestro proyecto.

Las funciones suscriptoras almacenan cada mensaje del autómata en la base de datos y la información está disponible para la aplicación de control de forma instantánea. La sincronización es inmediata y la información mostrada por la aplicación web está siempre actualizada.

El almacenamiento NoSQL es flexible y dinámico. No es imprescindible en el desarrollo del proyecto, pero ha resultado útil.

2.9.6.2 Firestore y Realtime Database

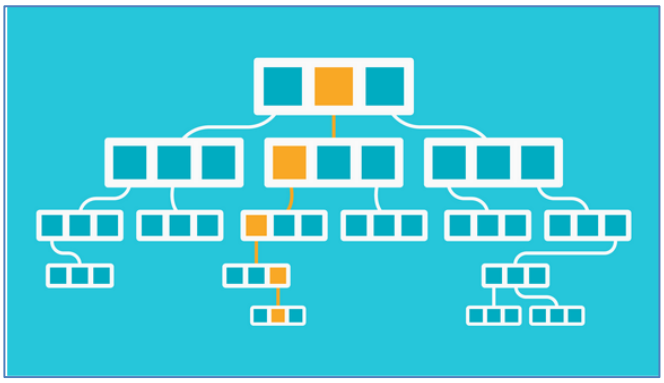

Figura 31 – Almacenamiento NoSQL. [firebase.com]

Son sistemas de base de datos NoSQL con sincronización en tiempo real y soporte off-line (alojadas en Cloud), a las que es posible acceder mediante la consola, el API REST y los distintos SDK disponibles en Firebase.

Realtime Database almacena la información como un gran documento JSON: distribuida en pares atributo-valor y arrays de elementos. Es adecuada para almacenar datos con una estructura simple.

En Firestore los datos se agrupan en colecciones, las colecciones contienen documentos y los documentos a su vez contienen campos, a los que se asignan valores.

La unidad básica de almacenamiento es el documento. Un documento es parecido a un registro JSON: contiene campos con valores asignados. Los campos pueden ser tipos de datos simples o estructuras complejas, llamados mapas.

Los documentos se agrupan en colecciones, los documentos dentro de la misma colección pueden contener distintos campos o almacenar

distintos tipos de datos en esos campos; sin embargo, se recomienda usar los mismos campos y tipos de datos para evitar errores.

Los nombres de documento dentro de una colección son únicos. El desarrollador puede proporcionar sus propias claves, o puede dejar que Firestore cree identificadores aleatorios de forma automática. Firestore crea, también de forma automática, índices de campo único en los documentos de una colección.

No es necesario "crear" ni "borrar" las colecciones: cuando se crea el primer documento de una colección, esta pasa a existir; si son borrados todos los documentos de una colección, esta deja de existir. Si se borra un documento, no se borrarán las subcolecciones que contiene. Esto es importante: debemos borrar de forma explícita todo el contenido anidado.

Hay que respetar el patrón alternado de colecciones y documentos, no pudiendo hacer referencia a una colección dentro de otra colección ni a un documento dentro de otro documento.

Firestore permite conservar datos sin conexión. Esta función almacena en caché una copia de los datos de Cloud Firestore que usa la App de forma activa, de modo que la App pueda acceder a los datos sin conexión en el dispositivo. Es posible escribir, leer, escuchar y consultar los datos en caché. Cuando el dispositivo vuelve a estar en línea, Firestore sincroniza los cambios locales que hizo la App con los datos almacenados de forma remota en Cloud Firestore.

2.9.7 Desarrollo de la aplicación de control

Se ha desarrollado un panel de control en forma de cuadro de mandos que muestra información precisa al usuario. La interfaz cuenta con los elementos necesarios para configurar el sistema.

La aplicación se ejecuta sin servidor en un entorno Cloud, en la modalidad PaaS. Se visualiza correctamente en cualquier dispositivo con acceso a Internet, capaz de ejecutar un navegador web moderno.

2.9.7.1 Elementos de diseño

Se ha utilizado el framework Bootstrap4. La personalización gráfica de los elementos en pantalla emplea la plantilla CSS del proyecto Material Design Bootstrap, una aproximación de los principios de diseño Material Design al framework Bootstrap.

2.9.7.2 Librerías JavaScript

JQuery acelera el desarrollo de scripts, mejora la compatibilidad del código y tanto las librerías, como algunos plugins de los frameworks seleccionados basan su funcionamiento en JQuery.

Los gráficos de representación de datos se han creado con Charts.js, una biblioteca simple; con una interfaz limpia y agradable. Basada en HTML5, es responsive y compatible con cualquier navegador. Los gráficos son programables y pueden actualizarse de forma dinámica.

Las tablas de datos utilizan *datatables.js*, un plugin basado en JQuery que nos permite realizar operaciones basadas en datos sobre la representación gráfica de una tabla HTML.

2.9.7.3 SDK's empleados

Se han utilizado los SDK de las plataformas elegidas:

1. **Particle JavaScript SDK**. Biblioteca para clientes web que permite interactuar con la plataforma Particle Cloud, facilitando el acceso a las variables y funciones de los dispositivos.
2. **Firebase JavaScript SDK**. Implementa las librerías cliente empleadas por las aplicaciones web para el acceso a los servicios de Firebase

2.10 Conclusión

Este segundo capítulo ha servido para introducir los conceptos teóricos necesarios para abordar un proyecto IoT. Comenzando por el hardware y su programación, siguiendo con los mecanismos de intercambio y almacenamiento de información, y finalizando con la programación de una aplicación web que nos permita monitorizar y controlar el sistema en tiempo real.

Capítulo 3

Desarrollo del sistema hardware

3.1 Introducción

En el segundo capítulo se han sentado las bases para entender el proyecto y acometer su consecución con garantías de éxito. Se ha hecho una introducción teórica, presentado las tecnologías empleadas y expuesto la metodología seguida en su ejecución.

Este es el primer capítulo de desarrollo. Centrado en el diseño del sistema hardware, ofrece una exposición detallada de los componentes del sistema y su relación. Termina mostrando el diseño final, razonando la forma de conectar cada elemento y exponiendo su rol en el sistema.

3.2 Requisitos hardware

Los requisitos del proyecto establecen la necesidad de crear un sistema autónomo, portátil, programable y configurable. Un sistema capaz de obtener información de estado en tiempo real y actuar sobre el medio.

Se trabaja sobre la abstracción del huerto como un conjunto de uno o varios contenedores de plantación. Cada contenedor de plantación es monitorizado y gestionado de forma independiente mediante un nodo IoT. El huerto, por lo tanto, estará formado por un conjunto de asociaciones nodo-módulo.

El nodo se ha diseñado para ser vinculado a un contenedor de plantación, monitorizar su estado y gestionar los ciclos de riego. Debidamente programado, recibirá su configuración y adaptará su comportamiento a ésta.

Para cumplir la especificación se ha creado un nodo formado por un microcontrolador PHOTON (Wifi y servicios Cloud de Particle integrados), al que se ha añadido un sistema de control de alimentación, además de los sensores y actuadores necesarios. El ensamblado final ha sido empaquetado en una caja estanca con grado de protección IP65.

Se optó por utilizar el PHOTON debido a su capacidad de transmisión de datos vía Wifi. Además, la plataforma IoT de Particle sustenta los mecanismos de monitorización y comunicación necesarios para el proyecto.

El mecanismo de control de suministro eléctrico está basado en la placa de expansión PHOTON Power Shield, capaz de alimentar el sistema mediante interfaz USB, una fuente de alimentación externa, una batería e incluso un pequeño panel solar. La placa actúa, además, como sistema de carga de la batería conectada.

Los sensores y actuadores, de la familia GROVE de seeedstudio, se conectan al núcleo mediante una tarjeta de expansión, denominada PHOTON Base Shield.

Los sensores son el medio de obtención de datos del nodo. Hay sensores con distintos tipos de interfaz: digital, analógica e I2C. Los elementos GROVE incluyen, además del sensor, toda la electrónica adicional necesaria para su conexión al sistema. Se miden parámetros de aire (temperatura, humedad e iluminación) y tierra (temperatura y humedad).

Los actuadores permiten modificar las condiciones del entorno. El nodo incluye un único actuador: una bomba peristáltica conectada a un módulo relé, que es el que controlamos desde el PHOTON. Tal y como sucede con los sensores, el relé incluye en el empaquetado la electrónica necesaria para su conexión y uso desde nuestro sistema.

Todos los componentes disponen de controladores en forma de librerías.

La Tabla 2 contiene el listado de elementos empleados:

Nombre	Descripción	Interfaz
PHOTON	Microcontrolador	
Placa de alimentación	Placa de alimentación	
Li Po Batería	Alimentación del circuito	
PHOTON Placa base	Placa Grove-PHOTON	
GROVE - Temp & Hum. Sensor. Pro.	Sensor de aire.	Digital
GROVE – Luz solar. Sensor.	Sensor de aire.	I2C
GROVE - Humedad. Sensor.	Sensor de tierra.	Analógico
GROVE - Temperatura. 1.2 Sensor.	Sensor de tierra.	Analógico
GROVE - Relé. Actuador.	Accionador de la bomba	Digital
GROVE – Adaptador de montaje	Montaje	
GROVE - Terminal	Conexión de sensores	
GROVE - Switch(P)	Desactivación del riego	
Fuente de alimentación - 12VDC 1ª	Alimentación del circuito	
Bomba peristáltica de líquido.	Bomba de agua	
Empaquetado y montaje	Caja estanca IP65	

Tabla 2 – Listado de componentes Hw.

3.3 Componentes principales

3.3.1 PHOTON PCB

El dispositivo se muestra en la Figura 32.

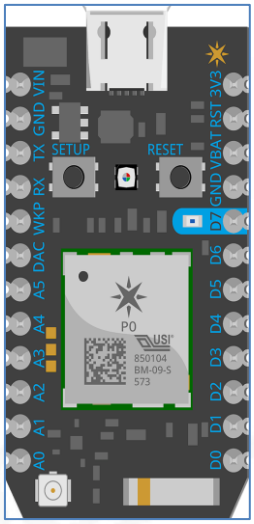

Figura 32 – PHOTON. [particle.io]

Este es el núcleo del nodo. Su diseño es de código abierto y posee las siguientes **características**:

1. Procesador STM32F205RGY6 ARM 120Mhz Cortex M3
2. Chip Wi-fi Broadcom BCM43362.
3. 1MB flash, 128KB RAM
4. On-board RGB status LED.
5. Sistema operativo en tiempo real (FreeRTOS)
6. Soft AP setup.
7. Certificación FCC, CE e IC.

Se trata de una PCB programable, capaz de conectarse a cualquier red 802.11b/g/n, e incluye un DeviceOS pregrabado que expone un API mediante el cual podemos acceder a los servicios IoT de particle.

3.3.2 PHOTON Power Shield

El dispositivo se muestra en la Figura 33.

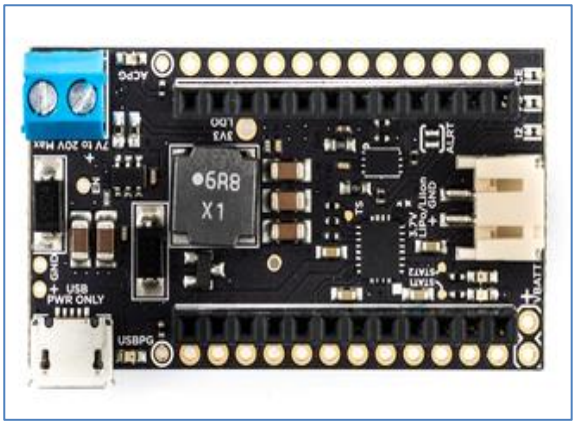

Figura 33 – Phower Shield. [particle.io]

Específicamente diseñada para ser compatible con el PHOTON, la Power Shield convierte el nodo en un dispositivo portátil. Basada en el sistema de administración y control de batería MCP73871, permite alimentar el PHOTON mediante: cable USB, alimentación externa (fuente de alimentación DC o panel solar) o una batería (Li-Po o Li-Ion). La placa permite administrar y monitorizar los niveles de carga de la batería desde el PHOTON.

Características:

1. Alimentación externa: USB o fuente DC (De 7V a 20V).
2. Consumo de corriente: 500mA max (USB) y 1.2A max (fuentes DC)

3.3.3 PHOTON Base Shield

El dispositivo se muestra en la Figura 34.

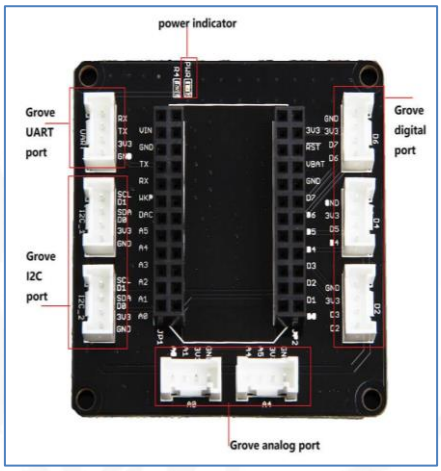

Figura 34 – Base Shield. [seeedstudio.com]

GROVE es un sistema modular de prototipado rápido: un conjunto de sensores, actuadores y cables de conexión basados en zócalos.

Cada zócalo permite la conexión de un elemento GROVE de prototipado rápido, garantizando el desarrollo acelerado de dispositivos de prueba.

Cada elemento dispone de un controlador empaquetado en forma de librería, que podemos obtener junto con instrucciones de uso y código de ejemplo.

Hay cuatro tipos de interfaces en GROVE: digital, analógica, I2C y UART. Los conectores y los zócalos tienen cuatro pines (amarillo, blanco, rojo y negro) cuya función varía en base al tipo de interfaz.

La tarjeta de expansión Grove Base Shield para el PHOTON es una placa que incluye un zócalo en el que conectar el PHOTON, además de ocho puertos GROVE: un puerto UART, dos I2C, dos analógicos y tres digitales.

3.3.4 GROVE - Temp & Hum. Sensor. Pro

El dispositivo se muestra en la Figura 35.

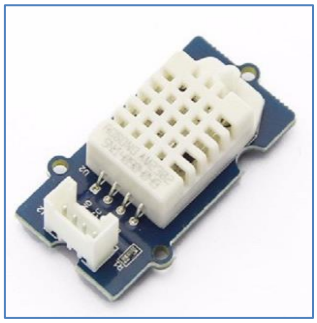

Figura 35 – Sensor de Temp. Y humedad. [seeedstudio.com]

Basado en el sensor de temperatura y humedad DHT22, permite detectar variaciones de humedad entre el 5% y el 99%, así como rangos de temperatura entre -40ºC y 80ºC.

Su precisión es de un 2% en el cálculo de humedad relativa y de 0.5ºC en la medición de temperatura ambiente.

Especificaciones

Item	Min	Norm	Max	Unit
Input voltage (VCC)	3.3	-	6	V
I/O Logic Level	-	based on VCC	-	V
Measuring Current Supply	1	-	1.5	mA

Standby Current Supply	40	-	50	uA
Measuring range **(Humidity)**	5%	-	99%	RH
Measuring range **(Temperature)**	-40	-	80	˚C
Accuracy **(Humidity)**	-	-	±2%	RH
Accuracy **(Temperature)**	-	-	±0.5	˚C
Resolution **(Humidity)**	-	-	0.1%	RH
Resolution **(Temperature)**	-	-	0.1	˚C
Repeatability **(Humidity)**	-	-	±0.3%	RH
Repeatability **(Temperature)**	-	-	±0.2	˚C
Long-term Stability	-	-	±0.5%	RH/year
Signal Collecting Period	-	2	-	S
Respond Time 1/e(63%)	6	-	20	S
Signal pin mode	-	Digital	-	-

Tabla 3 – Especificaciones sensor.

3.3.5 GROVE – Luz solar. Sensor

El dispositivo se muestra en la Figura 36.

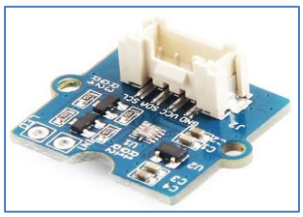

Figura 36 – Sensor de luz. [seeedstudio.com]

Se trata de un elemento multicanal, con la capacidad de medir luz UV, luz visible e infrarroja. El dispositivo se basa en el sensor SI1145 y se comunica mediante una interfaz digital I2C. Ofrece un rendimiento excelente en condiciones dinámicas de funcionamiento y puede ser programado.

Las mediciones se especifican en lúmenes para luz visible e infrarroja, mientras que la radiación UV se mide en base al índice UV.

Especificaciones:

Operating Voltage	3.0-5.5V
Working current	3.5mA
Wave length	280-950nm
Default I2C Address	0x60
Operating Temperature	-45-85℃

Tabla 4 – Especificaciones sensor.

3.3.6 GROVE - Humedad. Sensor

El dispositivo se muestra en la Figura 37.

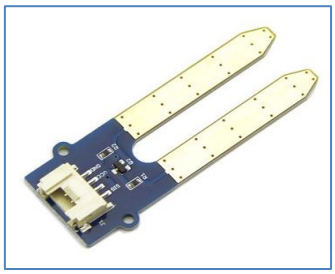

Figura 37 – Sensor de humedad. [seeedstudio.com]

Sensor de humedad para el suelo basado en resistividad. Su forma permite insertarlo varios centímetros en el suelo, detectando la presencia de agua alrededor de sus patas.

Especificaciones.

Item	Condition	Min	Typical	Max	Unit
Voltage	-	3.3	-	5	V
Current	-	0	-	35	mA
Output Value	Sensor in dry soil Sensor in humid soil Sensor in water	0 300 700	-	300 700 950	-

Tabla 5 – Especificaciones del sensor.

3.3.7 GROVE - Temperatura. 1.2 Sensor

El dispositivo se muestra en la Figura 38.

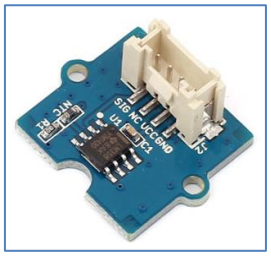

Figura 38 – Sensor de temperatura. [seeedstudio.com]

El sensor de temperatura GROVE 1.2 utiliza un termistor para medir la temperatura. La resistencia del termistor aumenta a medida que la temperatura desciende, permitiéndonos determinar la temperatura en torno al sensor.

El rango de detección de temperatura es de -40 a 125ºC, con una precisión de ±1.5ºC. Se alimenta con un voltaje de 3.3V a 5V.

3.3.8 GROVE - Relé. Actuador

El dispositivo se muestra en la Figura 39.

Figura 39 – Relé. [seeedstudio.com]

El módulo relé es un interruptor digital normalmente abierto. Haciendo uso del relé, es posible controlar cargas elevadas desde un circuito eléctrico de 5V. Incluye un indicador LED, que se enciende al cerrar el interruptor.

Especificaciones.

Parameter	V1.2
Operating Voltage	3.3V - 5V
Operating Current	100mA
Relay Life	100,000 Cycle
Max Switching Voltage	250VAC/30VDC
Max Switching Current	5ª

Tabla 6 – Especificaciones del relé.

3.3.9 Batería LiPo

El dispositivo se muestra en la Figura 40.

Figura 40 – Batería de litio. [seeedstudio.com]

Las baterías Li-Po son el estándar de la industria para la creación de dispositivos recargables. Son seguras y eficientes.

Se ha utilizado una batería de 3.7v directamente conectada a la PHOTON Power Shield, que actúa como mecanismo de conexión al PHOTON y sistema de carga de la batería en presencia de una fuente de alimentación externa.

3.3.10 Fuente de alimentación - 12VDC 1A

Se emplea una fuente de alimentación de 12v y 1A como medio de suministro de energía y carga de la batería Li-Po. Conectada a la PHOTON Power Shield, alimenta el circuito y recarga la batería.

La fuente de alimentación externa es necesaria para operar la bomba, ya que el suministro eléctrico de la batería resulta insuficiente para hacerla funcionar.

3.3.11 Bomba peristáltica

El dispositivo se muestra en la Figura 41.

Figura 41 – Bomba peristáltica. [adafruit.com]

A diferencia de otras bombas de líquido, las peristálticas realizan su función mediante el aplastamiento del tubo, creando un efecto de succión capaz de mover el líquido de su interior sin que las partes de la bomba estén en contacto directo con el líquido, así el líquido no se contamina.

La bomba puede ser controlada directamente desde un relé o a través de un chip de control para motores, lo que da la posibilidad de modificar la velocidad de giro, así como el sentido de giro del motor.

Detalles técnicos.

- Temperatura de trabajo:0℃ - 40 ℃
- Voltaje del motor: 12V DC
- Intensidad del motor: 200-300mA
- Caudal: hasta 100 ml/min
- Peso: 200 gramos

3.4 Diseño final del sistema hardware

El esquema final de conexión revela un dispositivo empotrado, en el que el PHOTON se conecta a la Power Shield y ésta a la Base Shield para formar un bloque electrónico que acepta alimentación USB, externa y mediante una batería.

Cuando la fuente de alimentación externa está conectada, suministra energía a la bomba peristáltica y al bloque electrónico. El relé controla el encendido de la bomba.

Cuando la fuente de alimentación está desconectada, el bloque electrónico se alimenta a través de la batería y la bomba queda inutilizada.

El diseño final del hardware se muestra en la Figura 42.

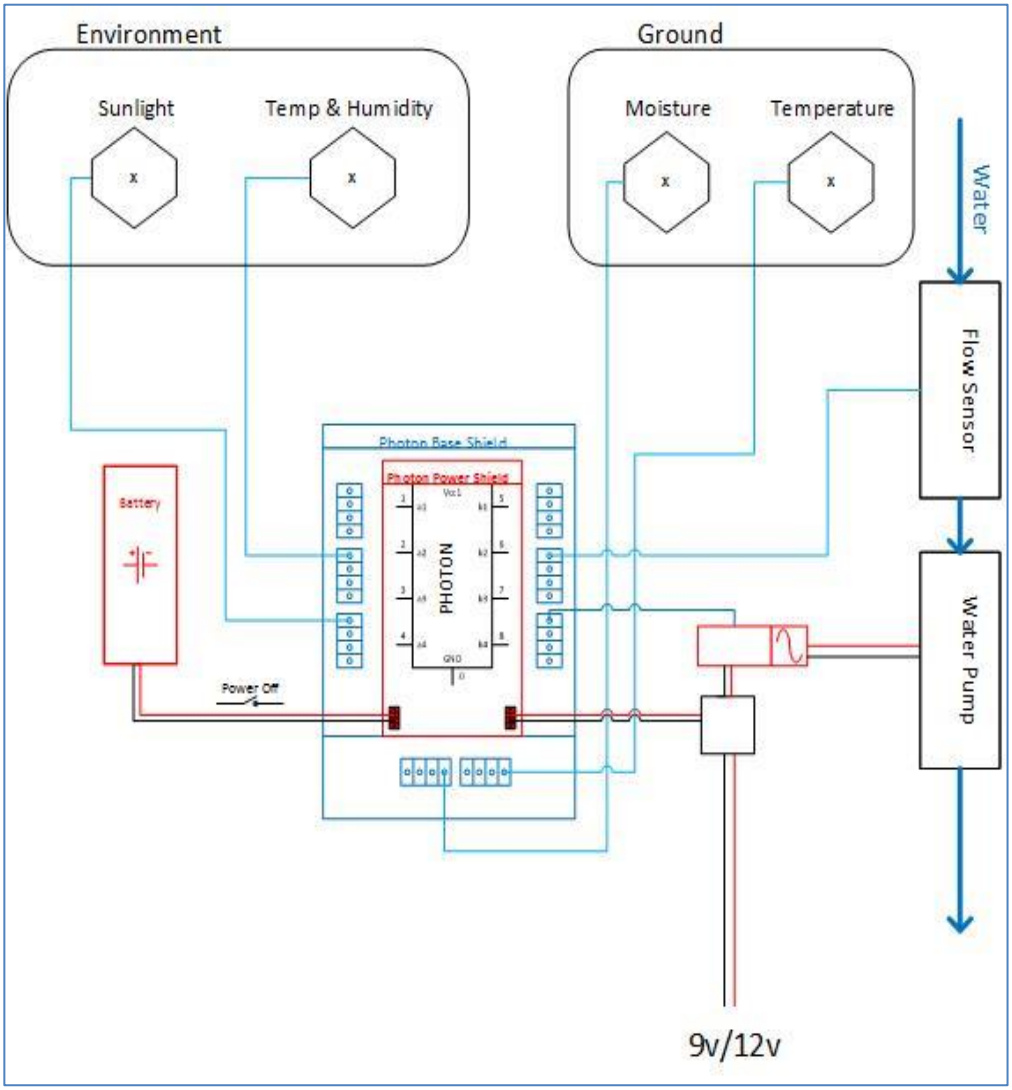

Figura 42 – Diseño final del hardware.

Si queremos apagar el dispositivo, debemos accionar el switch situado entre la batería y la Power Shield.

La Base Shield tiene conectados el relé y los sensores: moisture (humedad del suelo), temperatura de tierra, humedad y temperatura en aire y el sensor de luz.

3.5 Conclusión

En este punto, se dispone de un dispositivo capaz de cumplir con la especificación de requisitos inicial:

- La Power Shield ha permitido crear un dispositivo portátil en el que la electrónica es capaz de mantenerse en funcionamiento, aunque no exista fuente de alimentación externa. El sistema de riego, sin embargo, requiere la conexión de una fuente de alimentación externa debido a los requisitos eléctricos de trabajo de la bomba.
- La Base Shield convierte el núcleo en un sistema de prototipado, listo para conectar los elementos GROVE necesarios sin necesidad de soldar componentes.
- Los sensores permiten capturar la información necesaria y el relé conectado a la bomba activar el mecanismo de riego a voluntad.
- El dispositivo es programable y está integrado con la plataforma de servicios IoT de Particle.

Hasta aquí el diseño y desarrollo físico del dispositivo. En adelante comienza el trabajo de programación del sistema.

Capítulo 4

Desarrollo del autómata

4.1 Introducción

En este cuarto capítulo se exponen, de forma detallada, el conocimiento necesario y el trabajo realizado hasta crear un autómata conectado de monitorización y control ambiente.

Se detallan los elementos implicados en el desarrollo de software y dispositivos IoT, se enumeran las herramientas empleadas en el proceso, se describen las particularidades de la plataforma IoT de Particle, y finalmente, se presenta la lógica subyacente en la máquina de estados.

Al final del capítulo se exponen los métodos que definen el comportamiento del autómata, así como los estados y las reglas de transición entre ellos.

4.2 Conceptos de trabajo

La comunicación del dispositivo se basa en una serie de primitivas de que el DeviceOS provee a la aplicación cliente. Estas primitivas son:

- **Call a function**. La plataforma permite registrar funciones para que sean accesibles a través de Internet mediante llamadas al API REST. Es posible registrar hasta 15 funciones y el nombre está limitado a 12 caracteres. La función debe devolver un entero y -1 se toma como error en la llamada. El parámetro de llamada a la función es un string, limitado a 63 caracteres.

Puede exponerse también un método de un objeto C++.

- ***Particle.function("led",ledToggle);***
- ***POST /v1/devices/{DEVICE_ID}/{FUNCTION}***

- **Retrieve a variable**. Se pueden registrar variables, de forma que su valor sea accesible a través de Internet mediante llamadas al API REST. Es posible registrar hasta 20 variables y el nombre de la variable está limitado a 12 caracteres. Las variables serán INT, DOUBLE o STRING.

  - ***Particle.variable("analogvalue", &analogvalue, INT);***
  - ***GET /v1/devices/{DEVICE_ID}/{VARIABLE}***

- **Publish an event**. *Particle.publish* envía un mensaje al Cloud diciendo: un envento acaba de ocurrir. Se puede fijar el nombre del evento, su privacidad y aportar algo de información al evento.

  - ***Particle.publish("beamStatus", "broken", 60, PRIVATE | [WITH_ACK]);***
  - ***GET /v1/events/{EVENT_NAME}***

  1 evento por segundo. Se pueden enviar lotes de 4 por segundo; pero eso hará que tarde 4 segundos en recuperarse.

- **Subscribe to events**. Es posible registrar una función como manejador o un método de un objeto C++. El handler devuelve void y tiene como argumentos dos cadenas (const char *nombreCadena). El primer argumento es el nombre del evento y el segundo son los datos que vienen con el evento (puede ser null).

  - ***Particle.subscribe("buddy_unique_event_name", myHandler, MY_DEVICES);***

  Particle.subscribe() devuelve bool. Un dispositivo puede registrar un máximo de 4 manejadores de evento.

- ***Unsubscribe***: Elimina todos los handlers activos.

  - ***particle.unsubscribe();***

Particularidades relevantes para el proyecto:

Mientras que las variables y funciones es recomendable registrarlas en el setup(), los eventos deben ser publicados y los handlers suscritos a ellos dentro del loop(), llegando a cambiar la lista de eventos publicados/suscritos de forma dinámica.

Las llamadas a particle.publish() y particle.subscribe() utilizan el mismo buffer, por lo que es recomendable almacenar los datos recibidos en cuanto lleguen.

4.3 Descripción del autómata

El autómata está formado por siete estados, cuya definición y condiciones de transición se exponen aquí de forma detallada:

1. ***InitCheck***: Aquí se inicia el sistema.

   El sistema arranca y comienza en este estado.

   Inicia los sensores y el resto del Hardware.

   .- Si no es capaz, habria que reiniciarlo.

   Carga los datos EEPROM. [Modulo-Umbrales-Alerta].

   .- Si no es capaz, habria que configurarlo desde la App.

   Reset de los flags de estado.

   Si la inicializacion-carga es correcta:

   mode = ON

   Siguiente estado: ReadSens.

   Si la inicializacion-carga no es correcta:

   Siguiente estado: ErrorReport.

   Los actuadores permanecen inactivos.

2. ***ReadSens***: Aquí se mantiene el vector de estado.

Llegamos a este estado desde InitCheck, SetPlanning, RestEval, o SystemOff.

Ajusta la ventana de riego:

Primavera-verano = 20 - 23 horas.

Otoño-invierno = 13 - 16 horas.

Si mode = OFF

Siguiente estado: EstadoOff.

Si mode = ON

Actualiza el vector de estado.

Actualiza el vector de alertas.

Si ha variado alguna alerta:

Publica las alertas en Firebase.

Si estamos en la ventana de riego y debemos regar:

Siguiente estado: SetPlanning.

Los actuadores permanecen inactivos.

3. ***SetPlanning***: Aquí se planifica el riego.

Llegamos a este estado desde ReadSens.

Si mode = OFF

Siguiente estado: EstadoOff.

Si mode = ON

Esta dentro de la ventana de riego:

Si el modulo es de exterior:

Descarga informacion externa: AEMET.

Si va a llover:

Siguiente estado: ReadSens.

Si no va a llover:

Descarga informacion externa: SIAR.

Si el modulo es de interior:

Descarga informacion externa: SIAR.

moistureInicial = moisture.

Si el flag de moisture no esta activo:

Km -= 0.1 (Valores entre 0.5 y 1.5)

Si moistureInicial < marchitez:

Km += 0.1 (Valores entre 0.5 y 1.5)

marchitando = true.

publica hito en firebase.

Siguiente estado: doWatering.

Si no lo esta:

Siguiente estado: ReadSens.

Si hay errores en la descarga de informacion externa:

Siguiente estado: ErrorReport.

Los actuadores permanecen inactivos.

4. ***DoWatering***: Aquí se ejecuta el riego.

Llegamos a este estado desde setPlanning.

Si mode = OFF

Siguiente estado: EstadoOff.

Si mode = ON

horaDeRiego = Time().hour();

Ejecuta el plan de accion en base a:

Datos dinamicos: ETo - Km - Fc.

Datos del modulo: q - Ne - eficiencia - m2.

Siguiente estado: RestEval.

Si hay errores con la bomba:

Siguiente estado: ErrorReport.

Los actuadores permanecen activos.

5. ***RestEval***: Aquí dejamos reposar el riego y lo evaluamos.

Llegamos a este estado desde doWatering.

Si mode = OFF

Siguiente estado: EstadoOff.

Si mode = ON

Permanece inactivo durante restTime minutos.

moistureFinal = moisture.

Si moistureFinal < umbrales[moisture].max:

Fc += 0.1 (Valores entre 0.5 y 1.5)

Si moistureFinal > capacidadCampo:

Fc -= 0.1 (Valores entre 0.5 y 1.5)

Si moistureFinal > saturacion:

Fc -= 0.1 (Valores entre 0.5 y 1.5)

encharcado = true.

publica hito en firebase.

Enviamos los datos del riego a Firebase.

Actualiza los valores EEPROM.

Siguiente estado: ReadSens.

Si hay errores con la medicion, al publicar o al guardar los datos:

Siguiente estado: ErrorReport.

Los actuadores permanecen inactivos.

6. ***ErrorReport***: Aquí se tratan los errores.

Llegamos a este estado desde cualquier otro estado.

Publica en firebase la informacion del error como hito.

Siguiente estado: SystemOff.

Los actuadores permanecen inactivos.

7. ***SystemOff***: Aquí el sistema permanece inactivo.

Llegamos a este estado si el usuario desactiva el sistema o se ha producido un error.

Apaga el automata.

Si mode = ON

Siguiente estado: ReadSens.

Los actuadores permanecen inactivos.

En la Figura 43 puede verse la representación gráfica del autómata.

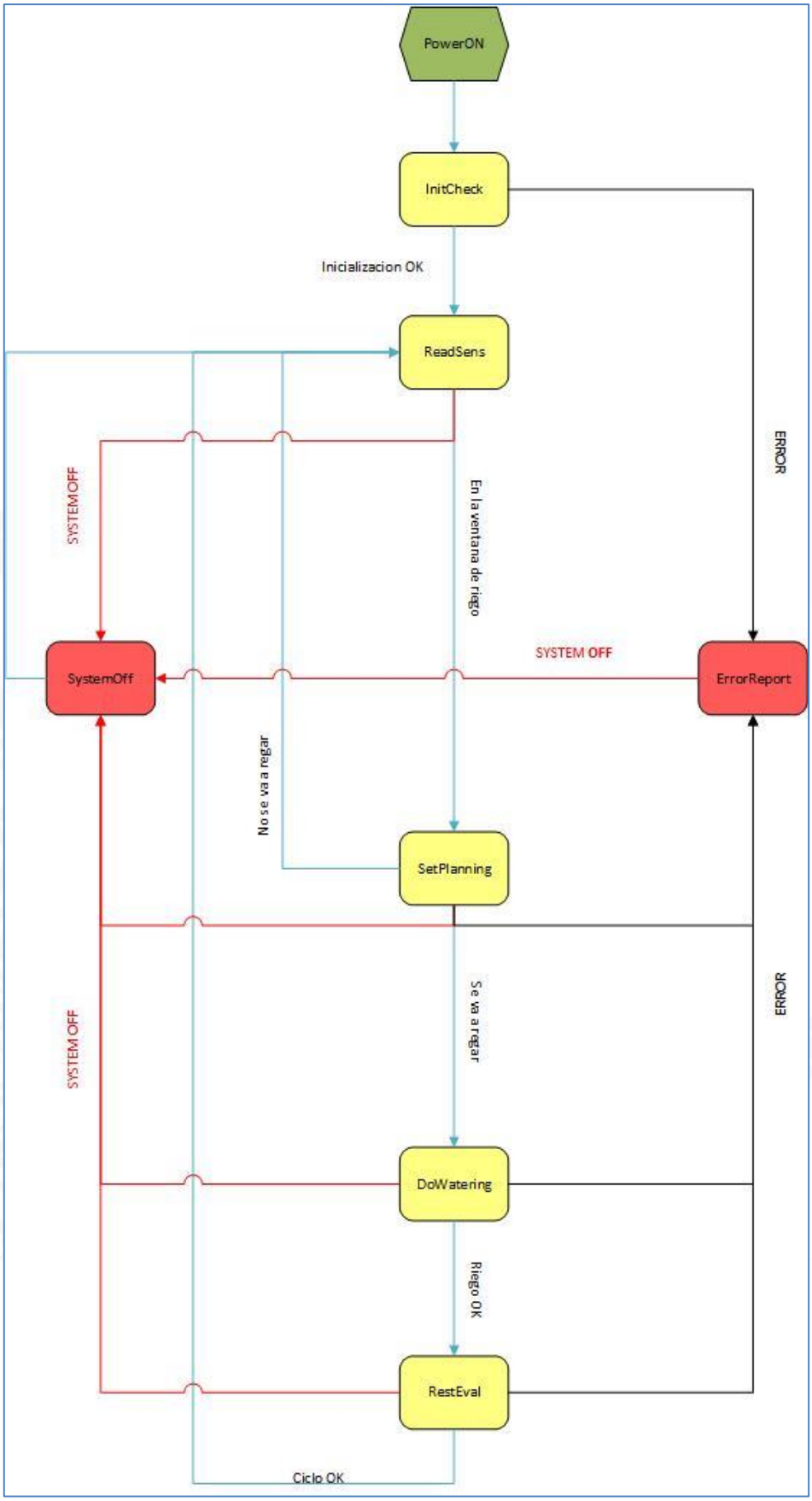

Figura 43 – Diagrama de la MEF.

4.5 Conclusión

Este ha sido un capítulo denso e interesante, el núcleo del proyecto. Cuando se establecieron los requisitos de funcionamiento de la herramienta fueron ambiciosos.

En este punto, el autómata está listo para ser instalado, monitorizar un contenedor de plantación y ser controlado de forma remota.

La plataforma y el hardware de Particle han hecho posible el diseño e implementación, sin conocimientos previos, de un dispositivo completamente funcional, portable y listo para ser conectado.

Capítulo 5

Sistema de comunicaciones

5.1 Introducción

En este momento, el dispositivo es funcional; pero la especificación de requisitos exige ir un poco más lejos: es necesario construir un dispositivo conectado. De forma nativa, el PHOTON incluye capacidad de conexión Wifi. Las primitivas de comunicación del DeviceOS facilitan al firmware de aplicación la transmisión de datos mediante publicación de eventos, y los mecanismos de integración de Particle Cloud conectan la plataforma a servicios externos. Es necesario, no obstante, comprender los fundamentos y las tecnologías de base empleadas en la autenticación, autorización y el envío de peticiones a otros sistemas.

En este capítulo se detalla el diseño e implementación del sistema de comunicación del autómata.

5.2 Implementación de la comunicación

Se han diseñado tres webhooks:

- Dos para la descarga de información meteorológica:
    - El primero obtiene la URL en la que se publica la predicción diaria para nuestro municipio de referencia.
    - El segundo descarga el JSON con la predicción semanal desde la URL obtenida en el primero.
- Uno para la descarga de la información de riego.

Además, se ha creado un modelo de interacción con Firestore basado en funciones de servidor, utilizadas para actualizar los datos de: eventos, hitos, alertas y estados. La Figura 44 muestra las integraciones en Particle Cloud.

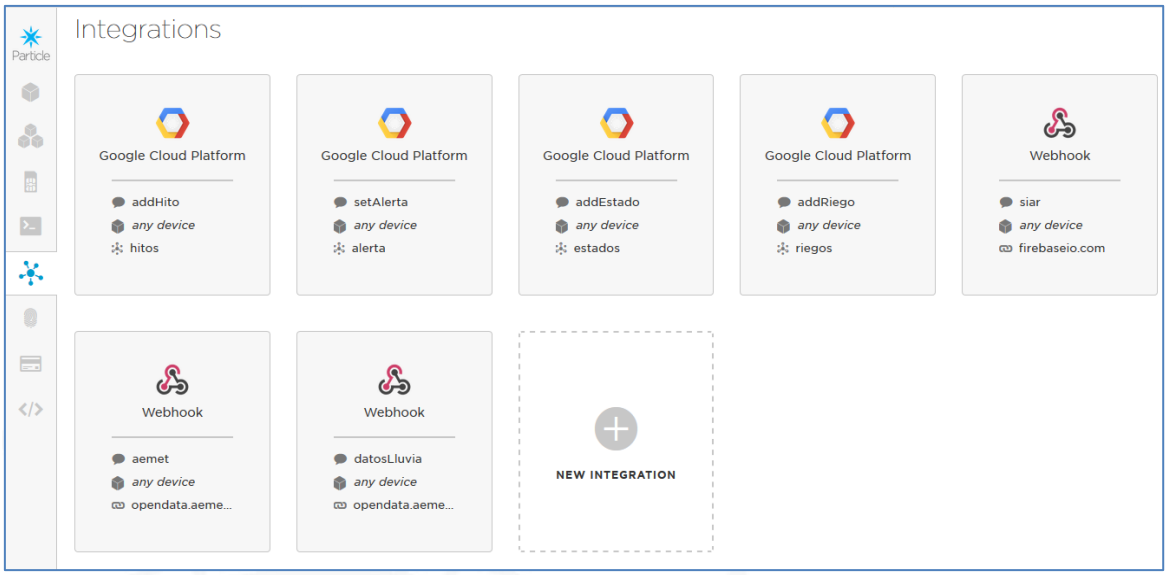

Figura 44 –Integraciones en Particle. [particle.io]

En el firmware de aplicación, los métodos responsables de la comunicación son:

- *void sendFbEstado()*: Envía el estado físico del módulo.
- *void sendFbAlertas()*: Mantiene actualizado el sistema de alerta.
- *void sendFbHito()*: Envía los eventos del sistema.
- *void sendFbRiego()*: Envía los datos de evaluación de riego.
- *bool getAemetInfo()*: Obtiene la predicción de lluvia.
- *bool getSiarInfo()*: Descarga el valor de la evapotranspiración de referencia.

Son cuatro métodos para el envío de datos y dos para la obtención de predicciones; no obstante, la complejidad no está en el firmware, sino en el sistema de webhooks y funciones de servidor que ha sido necesario crear.

5.3 Descarga de información meteorológica

El planificador de riego tiene en cuenta la probabilidad de lluvia. Se obtiene la predicción de la Agencia Española de Meteorología a través de AEMET Open Data, un API REST que permite la difusión y la reutilización de información meteorológica y climatológica.

Para el acceso al servicio es necesario solicitar un API KEY, que servirá para autenticar las peticiones HTTP. La documentación del API está disponible en la URL:

- *https://opendata.aemet.es/dist/index.html*

Para obtener la predicción meteorológica de un municipio es necesario hacer dos peticiones:

1. Una primera petición para averiguar la URL de la predicción diaria:
    a. *https://opendata.aemet.es/opendata/api/prediccion/especifica/municipio/diaria/idMunicipio*

2. Una segunda petición para descargar la petición de la URL obtenida:
    a. *https://opendata.aemet.es/opendata/sh/endPointDinamico*

Esta segunda petición obtiene como respuesta un documento JSON con los datos de la predicción semanal para el municipio de referencia.

5.3.1 Definición de webhooks

Se han definido sendos webhooks para la descarga del documento JSON con la predicción meteorológica.

- **Primer webhook**: autenticación basada en API KEY y filtrado de la respuesta mediante la plantilla mustache *{{datos}}*, lo que hace llegar al firmware únicamente la URL generada dinámicamente por el sistema de la AEMET.

Definición del webhook:

```
{
    "event": "aemet",
 "url":
"https://opendata.aemet.es/opendata/api/prediccion/especifica/municipio/diaria/id",
    "requestType": "GET",
    "noDefaults": true,
    "rejectUnauthorized": false,
    "responseTemplate": "{{datos}}",
    "headers": {
        "Accept": "Application/json",
        "api_key": [YOUR_API_KEY]
    }
}
```

- **Segundo webhook**: toma como parámetro en la URL el endpoint obtenido anteriormente y descarga el JSON con la predicción meteorológica semanal. Interesa únicamente la del día, así que la respuesta es filtrada mediante una plantilla mustache:

  Definición del webhook:

```
{
    "event": "datosLluvia",
    "url":
"https://opendata.aemet.es/opendata/sh/{{PARTICLE_EVENT_VALUE}}",
    "requestType": "GET",
    "noDefaults": true,
    "rejectUnauthorized": false,
    "responseTemplate":
        "{[PLANTILLA_MUSTACHE]}"
}
```

5.3.2 Publicación de eventos

La ejecución del webhook se provoca en Particle mediante la publicación de eventos. La secuencia se inicia en *getAemetInfo()*:

```
// Lanzamos el webhook para obtener los datos
Particle.publish("aemet", sistema, PRIVATE);
delay(10000);
```

Con lo que se obtiene la URL de la predicción diaria. Después, en *myHandlerAemet(const char *event, const char *data)*:

```
// Lanzamos el segundo webhook pasando el endpoint actual
Particle.publish("datosLluvia", endPoint, PRIVATE);
delay(1000);
```

Se descarga el JSON con la predicción dinámica.

5.3.3 Recogida y tratamiento de datos

Cada webhook genera un evento con la respuesta (o el error) de la petición HTTP. En el firmware es necesario asignar un manejador a ese evento de respuesta (o error).

Hay un manejador para cada webhook:

1. *Particle.subscribe("hook-response/aemet", **myHandlerAemet**, MY_DEVICES);*
    a. *Obtiene el endPoint de la URL para lanzar el segundo webhook.*
2. *Particle.subscribe("hook-response/datosLluvia", **myHandlerDatosLluvia**, MY_DEVICES);*
    a. *Almacena la información del JSON en variables del firmware.*

El manejador de errores es el mismo para todos los webhooks:

1. *Particle.subscribe("hook-error", **myHandlerError**, MY_DEVICES);*

5.4 Descarga de información de riego

El planificador de riego estima la cantidad de agua necesaria en base a la evapotranspiración de referencia, que es calculada por los servicios de información al regante. El SIAR ofrece una aplicación móvil de consulta; pero no un API REST, por lo que se almacena diariamente en Realtime Database (un producto de Firebase) el valor diario de la evapotranspiración de referencia y desde ahí lo descarga el firmware del autómata.

Es posible acceder a Realtime Database mediante peticiones HTTP de la forma:

- *https://idProyecto.Firebaseio.com/database/archivo.json*
  - *La petición irá autenticada en la URL.*

5.4.1 Definición de webhooks

El webhook hace una petición al API de Firebase. Se hace a la URL de la base de datos y se autentica mediante el API_KEY que suministra la AEMET.

Definición del webhook:

```
{
    "event": "siar",
    "url": "https://huertotool.Firebaseio.com/SIAR/ETo.json",
    "requestType": "GET",
    "noDefaults": true,
```

```
    "rejectUnauthorized": false,
    "query": {
       "auth": [YOUR_API_KEY]
    }
}
```

5.4.2 Publicación de eventos

Se lanza el webhook mediante la publicación del evento correspondiente. La secuencia se inicia en *getSiarInfo()*:

```
// Lanzamos el webhook y esperamos la respuesta
Particle.publish("siar", sistema, PRIVATE);
delay(5000);
```

5.5.3 Recogida y tratamiento de datos

Como ya se ha indicado, el webhook genera un evento con la respuesta (o el error) de la petición HTTP. Se definen los manejadores de eventos:

Un manejador para la respuesta del webhook:

1. *Particle.subscribe("hook-response/siar", **myHandlerSiar**, MY_DEVICES);*
    a. Asigna el valor de la respuesta a la variable del firmware.

El manejador de errores es el mismo para todos los webhooks

1. *Particle.subscribe("hook-error", **myHandlerError**, MY_DEVICES);*

5.5 Envío de datos a Firestore

El autómata genera de forma constante información de estado, alertas, hitos de sistema y seguimiento de riego. Se almacena en Firestore (base de datos NoSQL en tiempo real de Firebase) mediante la integración de Particle Cloud con Google Cloud Platform.

5.5.1 Implementación de las integraciones

Pasos necesarios para hacer funcionar la integración de Particle con Google:

1. Se crean los temas *pub/sub* a los que enviar mensajes.
2. Se añade como editor de cada tema la cuenta de servicio de Particle.
3. Se crea una integración por tema.

Esto se ilustra en las Figuras 45 y 46.

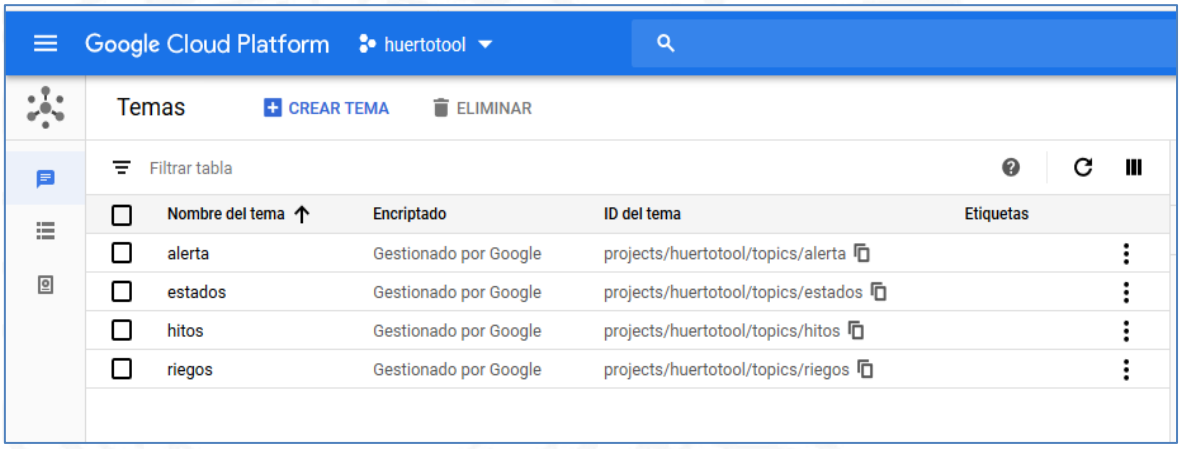

Figura 45 - Nuestros topics Pub/Sub. [cloud.google.com]

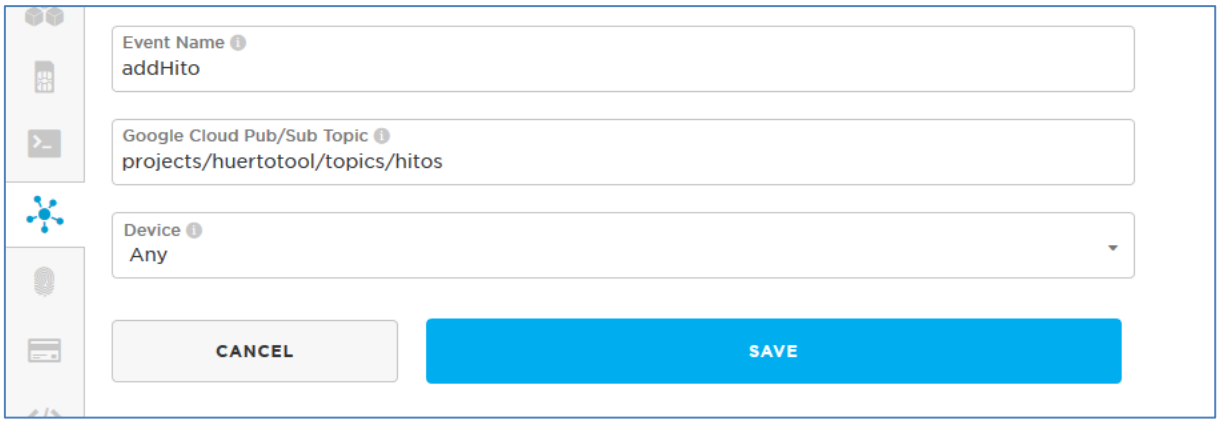

Figura 46 –Integración Google Cloud. [particle.io]

En este punto, la integración de Particle hará llegar los eventos publicados a los temas *pub/sub* correspondientes.

5.5.2 Implementación de las funciones de servidor

No hay una conexión directa entre Particle y Firestore, por lo que se han desarrollado un conjunto de funciones de servidor que se ejecutan en un entorno FaaS, monitorizan la publicación de mensajes en los temas y almacenan los mensajes en Firestore. Son llamadas funciones suscriptoras. La Figura 47 ilustra este punto.

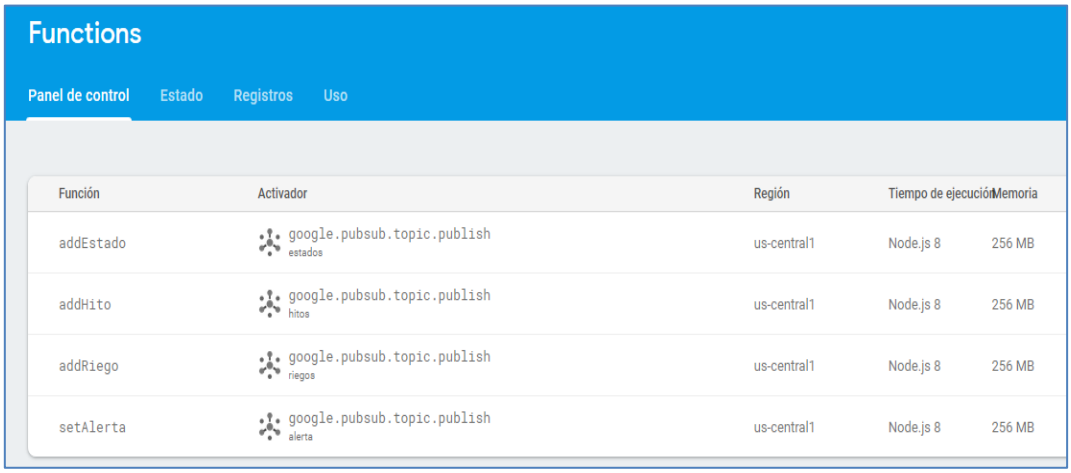

Figura 47 –Funciones suscriptoras. [firebase.com]

Cada función monitoriza un tema, analiza el mensaje, lo procesa y almacena los valores en Firestore. Se ejecutan cuando hay una publicación en el tema al que están suscritas:

```
exports.addHito = functions.pubsub.topic('hitos').onPublish((message) => {
```

5.5.3 Publicación de eventos y envío de datos

Los datos son enviados al tema *pub/sub* mediante la publicación del evento que lanza la integración. El cuerpo del mensaje es un JSON, adecuado para ser procesado por la función suscriptora. En el caso de los hitos:

```
void sendFbHito()
{
 // Variables de Hito
 descHito = INIT_CHECK;
 tipoHito = "Cambio de estado";
 snprintf(datosHito, sizeof(datosHito), "{\"descripcion\":\"%s\",\"tipo\":\"%s\"}",
 descHito.c_str(), tipoHito.c_str());
 Particle.publish("addHito", String(datosHito), PRIVATE);
 delay(1000);
}
```

El formato del JSON en datosHito es:

```
{
"descripcion":"Descripcion de Hito",
"tipo":"Tipo de Hito"
}
```

5.6 Medios de autenticación y autorización empleados

1. La App web de control utiliza:
    a. OAuth 2.0 para autenticar peticiones a las API de Firebase.
    b. Tokens de acceso (API key) para autenticar peticiones al API de Particle.
2. La integración de Particle con Google Cloud Platform emplea un mecanismo de autenticación basado en OAuth 2.0 y cuentas de servicio.
3. El webhook de acceso a datos AEMET emplea un API key.
4. El webhook de acceso a Realtime Database emplea un API key.

5.7 Conclusión

Haciendo uso del protocolo de transporte HTTP, arquitecturas REST y diversos medios de autenticación y autorización de solicitudes, Particle Cloud ofrece al firmware de aplicación mecanismos de comunicación avanzados, en forma de webhooks e integraciones con sistemas externos. El autómata hace uso de todo ello para obtener predicciones meteorológicas, estimaciones de riego y almacenar los datos de estado y funcionamiento.

Este capítulo ha servido para finalizar el autómata. En este momento no sólo funciona, sino que es capaz interactuar con servicios externos y adaptar su funcionamiento en base a predicciones.

Capítulo 6

Desarrollo del sistema de almacenamiento

6.1 Introducción

El resultado obtenido en el capítulo cinco es la versión final del autómata. En este punto se dispone de una herramienta de control completa:

- Expone las variables de estado a través del API de Particle.
- Expone sus métodos de control mediante el API de Particle.
- Implementa un autómata de estados finitos con la lógica del sistema.
- Descarga predicciones de servicios externos.
- Adapta su comportamiento.
- Envía la información de estado y funcionamiento a un sistema externo.

No obstante, el objetivo del proyecto es crear una herramienta de apoyo a la toma de decisiones, por lo que es necesario almacenar la información generada por el autómata. El análisis e interpretación de la información almacenada será la base de la toma de decisiones.

En este sexto capítulo se enuncian los requisitos de almacenamiento de información necesarios para alcanzar los objetivos, se exponen las particularidades de los mecanismos de almacenamiento de información en Firebase y se detalla la implementación del sistema de almacenamiento.

6.2 Requisitos de almacenamiento

Las especificaciones del proyecto establecen la necesidad de controlar una lista de tareas y mantener una base de datos heurística de plantas, además de los datos propios del módulo de plantación.

La base de datos de plantas debe contener información sobre los umbrales óptimos de crecimiento de cada planta, e información relativa a las fechas óptimas de siembra de cada planta.

Respecto al módulo de plantación, se controla:

1. Tipo de suelo.
2. Número de emisores de agua y caudal de riego de cada emisor.
3. Superficie del módulo.
4. Nivel de alerta del módulo.
5. Estados. Log de estados. Almacena el valor de los sensores en intervalos. Permite hacer un seguimiento de las condiciones del módulo.
6. Recolecciones. Histórico de recolecciones. Permite analizar la rentabilidad del módulo.
7. Siembras. Histórico de siembras. Permite analizar el uso del suelo del módulo.
8. Riegos. Histórico de riegos. Permite analizar y optimizar los riegos.
9. Hitos. Listado de eventos de funcionamiento: transiciones de estado, errores, etc.

6.3 Diseño e implementación del sistema de almacenamiento

Se ha implementado el sistema de almacenamiento sobre Firestore; no obstante, la posibilidad de acceder a los datos en Realtime Database mediante peticiones HTTP autenticadas con un API_KEY ha hecho que se almacene la información relativa a las predicciones del SIAR en esta base de datos. Así resulta más sencillo recuperarla desde el autómata.

6.3.1 Almacenamiento de la evapotranspiración de referencia

La información del SIAR no está disponible para su consulta a través de un API REST, como la información de la AEMET. Se salva este obstáculo utilizando la App móvil SIAR Info y guardando el valor diario de ETo en Realtime Database.

La estructura es simple:

1. Se crea la base de datos del proyecto, accesible en la URL:
    a. https://huertotool.firebaseio.com/

2. Se añade un espacio para la información relativa al SIAR:
    a. https://huertotool.firebaseio.com/SIAR

3. Finalmente, se añaden los pares clave-valor de los parámetros:
    a. https://huertotool.firebaseio.com/SIAR/ETo:valor

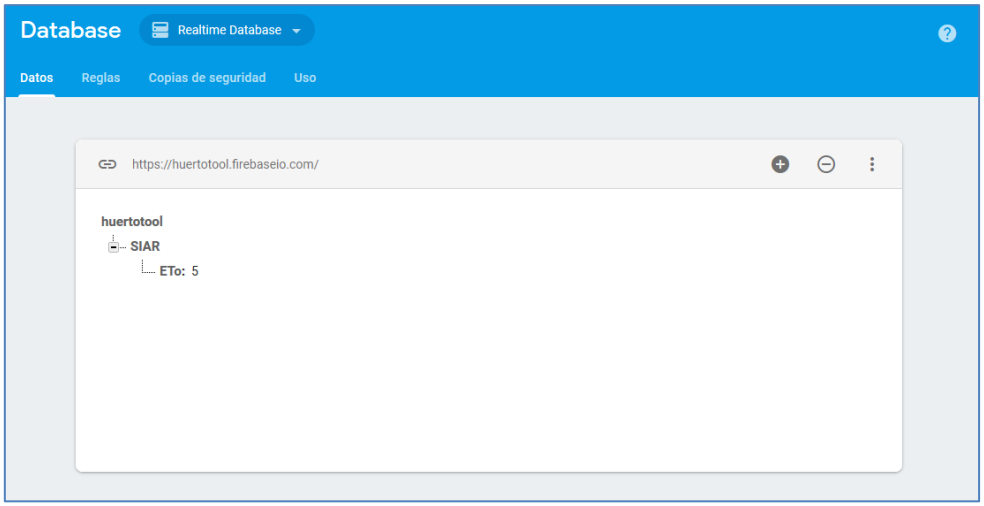

Figura 48 – Real Time Database. [firebase.com]

6.3.2 Almacenamiento de la información del autómata

La información del cuadro de mando y el autómata se almacena en Firestore. La estructura de documentos y colecciones se ha adaptado muy bien a los requerimientos del proyecto, quedando el diseño de la siguiente forma:

1. En el nivel superior, la base de datos: *huertotool*.
2. Un segundo nivel compuesto por tres colecciones:

    a. ***Plantas***. El listado heurístico de plantas es común a todos los módulos del huerto. Los campos de cada documento en esta colección son:
        i. *descripcion*. Cadena de caracteres.
        ii. *fechaInicioSiembra*. Marca de tiempo.
        iii. *fechaFinSiembra*. Marca de tiempo.
        iv. *horasDeLuz*. Número.
        v. *humMax*. Número.
        vi. *humMin*. Número.
        vii. *moistMax*. Número.
        viii. *moistMin*. Número.
        ix. *tempMax*. Número.
        x. *tempMin*. Número.

    b. ***Tareas***. El listado de tareas es común a todos los usuarios de la aplicación, así que es único. Los campos de cada documento en esta colección son:
        i. *criticidad*. Número.
        ii. *descripcion*. Cadena de caracteres.

    c. ***Módulos***. Idealmente, el huerto estará compuesto por una serie de módulos de plantación. Cada módulo contendrá la información relativa a su estado y funcionamiento. Los campos que describen el módulo son:
        i. *Identificador*. Este es el único tipo de documentos de la base de datos que no tendrá identificadores aleatorios.
        ii. *Ne*. Número de emisores. Número.
        iii. *q*. Caudal. Número.
        iv. *m2*. Superficie. Número.
        v. *tipoSuelo*. Cadena de caracteres.

3. En el tercer nivel, Las colecciones de cada módulo son:

    1. ***Alertas***. Niveles actuales de alerta del módulo.
        a. *horasDeLuz*. Número.
        b. *humedad*. Número.
        c. *moisture*. Número.
        d. *temperatura*. Número.

    2. ***Estados***. Log de estados. Serie temporal. Cada documento de esta colección tiene los siguientes campos:
        a. *UVIndex*. Número.
        b. *horasDeLuz*. Número.
        c. *humedadAire*. Número.
        d. *luzIR*. Número.
        e. *luzVisible*. Número.
        f. *moisture*. Número.
        g. *momento*. Marca de tiempo.
        h. *tempAire*. Número.
        i. *tempTierra*. Número.

    3. ***Hitos***. Listado de eventos de sistema. Cada documento de esta colección tiene los siguientes campos:
        a. *descripcion*. Cadena de caracteres.
        b. *momento*. Marca de tiempo.
        c. *tipo*. Cadena de caracteres.

    4. ***Plantas***. Listado de plantas presentes en el módulo. Cada documento de esta colección tiene los siguientes campos:
        a. *IdPlanta*. Cadena de caracteres. Referencia la tabla heurística de plantas.

    5. ***Recolecciones***. Listado de recolecciones del módulo. Cada documento de esta colección tiene los siguientes campos:
        a. *Kg*. Número.
        b. *IdPlanta*. Cadena de caracteres. Referencia la tabla heurística de plantas.
        c. *momento*. Marca de tiempo.

6. ***Riegos***. Listado de riegos del módulo. Cada documento de esta colección tiene los siguientes campos:
    a. *encharcado*. Boolean.
    b. *horaDeRiego*. Número.
    c. *moistureInicial*. Número.
    d. *moistureFinal*. Número.
    e. *momento*. Marca de tiempo.
    f. *tiempoDeRiego*. Número.

7. ***Siembras***. Listado de siembras del módulo. Cada documento de esta colección tiene los siguientes campos:
    a. *idPlanta*. Cadena de caracteres. Referencia la tabla heurística de plantas.
    b. *m2*. Número.
    c. *momento*. Marca de tiempo.

La Figura 49 muestra el sistema de almacenamiento en la consola Firebase.

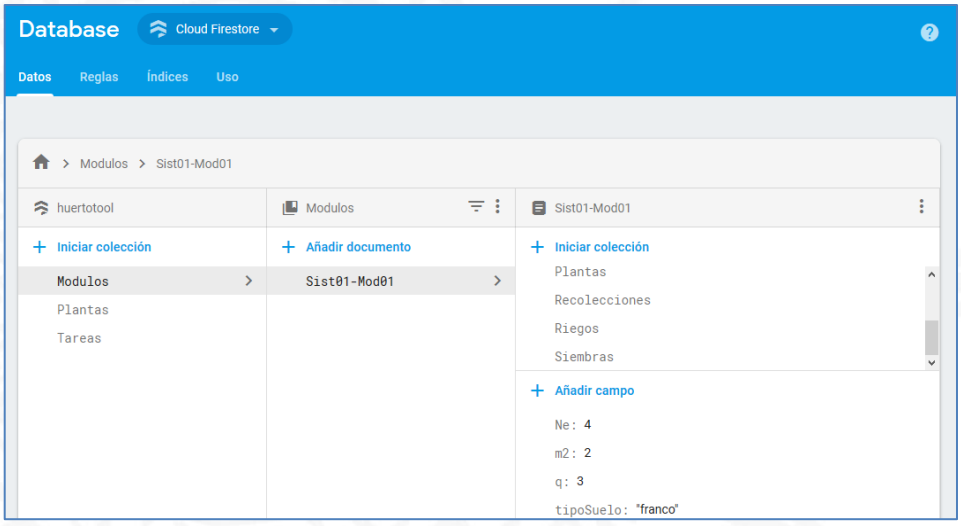

Figura 49 – Firestore. [firebase.com]

6.3.3 Asegurando el acceso a los datos

Se han establecido unos requisitos mínimos de seguridad en el acceso a los datos por parte de la aplicación. Al tratarse de una aplicación web, se controla el acceso a la información mediante Firebase authentication y las reglas de seguridad de Firestore.

Las reglas de seguridad de Firestore consisten en declaraciones match, que identifican documentos de la base de datos, y expresiones allow, que controlan el acceso a esos documentos. Una regla simple, que evite el acceso a usuarios que no tengan sesión iniciada en la aplicación sería:

- En el caso de Firestore:

```
service Cloud.Firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if request.auth.uid != null;
    }
  }
}
```

- Para Realtime Database:

```
{
  "rules": {
    ".read": "auth != null",
    ".write": "auth != null"
  }
}
```

6.4 Conclusión

En este sexto capítulo del proyecto se ha diseñado e implementado el sistema de almacenamiento de información del autómata. Se ha comenzado sentando las bases teóricas de los sistemas de almacenamiento empleados y se ha terminado describiendo el diseño final de la estructura de almacenamiento.

A falta de la aplicación web de control, el sistema está completo: es funcional, configurable a través de Particle Cloud, y guarda toda la información relativa a su funcionamiento en Firestore.

Capítulo 7

Desarrollo de la aplicación de control

7. 1 Introducción

El sexto capítulo termina con el sistema de almacenamiento listo para recibir información de forma continua. En este punto sólo queda diseñar y programar la aplicación de control.

En este capítulo se exponen los requisitos de la aplicación, se enumeran los componentes elegidos para desarrollarla, se muestra el diseño gráfico de la interfaz y se describen los mecanismos de interacción de la aplicación con el usuario, el autómata y el sistema de almacenamiento.

7.2 Diseño de la interfaz

La aplicación tiene una interfaz responsive, formada por una barra de navegación superior, un panel de controles y un pie de página simple. Se muestra en una única columna en tamaños reducidos de pantalla y expande su contenido a medida que el tamaño de pantalla aumenta.

7.2.1 La barra de navegación

La barra de navegación, que podemos ver en la Figura 50, se encuentra anclada a la parte superior. Se expande en tamaños grandes de pantalla y muestra un menú tipo hamburguesa cuando el espacio disponible disminuye.

Figura 50 – La barra de navegación.

Los enlaces de la barra son:

1. Módulo. Muestra información del módulo y su estado.
2. Tareas. Pantalla de gestión del listado de tareas.
3. Plantas. Pantalla de gestión del listado de plantas.
4. Sistema. Permite configurar el sistema de forma remota.
5. Alertas. Muestra el estado de alerta.
6. LogIn/LogOut. Gestiona el acceso a la aplicación.

7.2.2 Elementos de representación de datos

La aplicación tiene su punto de entrada en la sección dedicada al módulo. Por medio de elementos gráficos muestra información relativa a la configuración, el estado, los ciclos de riego, las recolecciones, el aprovechamiento del espacio y los hitos de funcionamiento del autómata. La Figura 51 muestra una de las secciones con gráficos.

Figura 51 – Sección de gráficos.

La información se estructura en secciones y cada sección muestra un botón con acceso a la pantalla de detalles. Las pantallas detalle muestran los listados de la sección correspondiente en forma de tabla.

Hay elementos, como las siembras y las recolecciones, cuya pantalla detalle permite la edición del listado; y otros, como los riegos y los hitos, en los que el detalle es simplemente una vista ampliada de la información en pantalla. Dos páginas amplían el detalle de estados: una el del contenedor y otra el de su entorno. La Figura 52 muestra el detalle del listado de riegos.

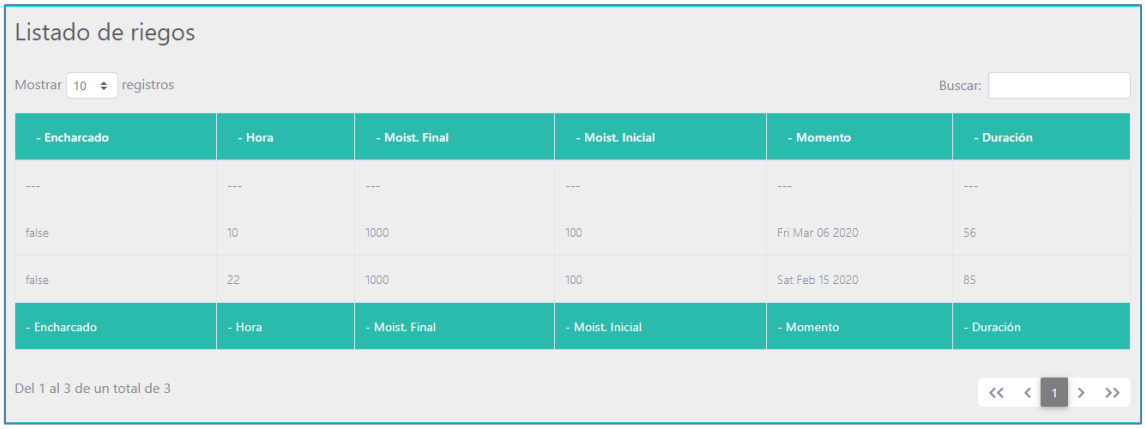

Figura 52 – Listado de riegos.

El botón de alertas muestra un formulario modal con información sobre el estado de alerta del módulo. Puede observarse en la Figura 53.

Figura 53 – Formulario de alertas.

7.4.3 Elementos de control

En la página dedicada al módulo, si pulsamos en el botón detalle de la sección de configuración, un formulario modal nos permite modificar las propiedades del módulo de plantación: tipo de suelo, emisores, caudal y superficie. Los parámetros del módulo son los mostrados en la Figura 54.

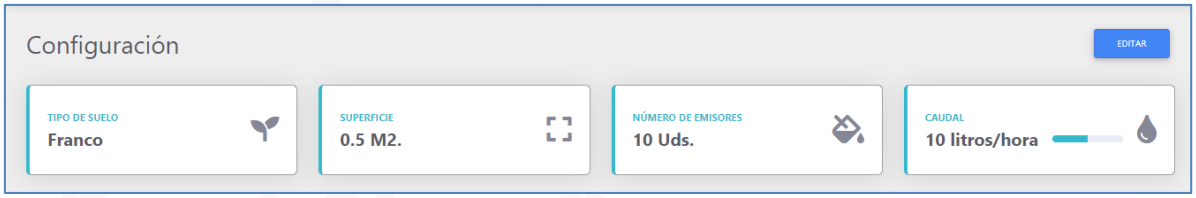

Figura 54 – Sección de configuración.

La página dedicada al sistema ofrece interacción con el autómata. Muestra el valor de las variables de estado y permite ejecutar las funciones expuestas por el firmware. De esta forma podemos: encender/apagar el autómata, modificar los ciclos de reposo y publicación de estados, etc.

El funcionamiento del autómata puede ser configurado mediante los controles mostrados en la Figura 55.

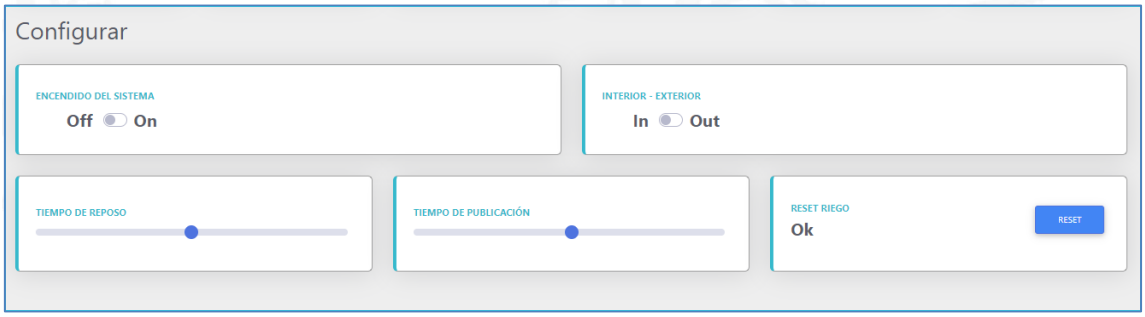

Figura 55 – Configuración del módulo.

7.4.4 Pie de página

El pie de aplicación contiene elementos puramente informativos. La Figura 56 muestra el pie de página.

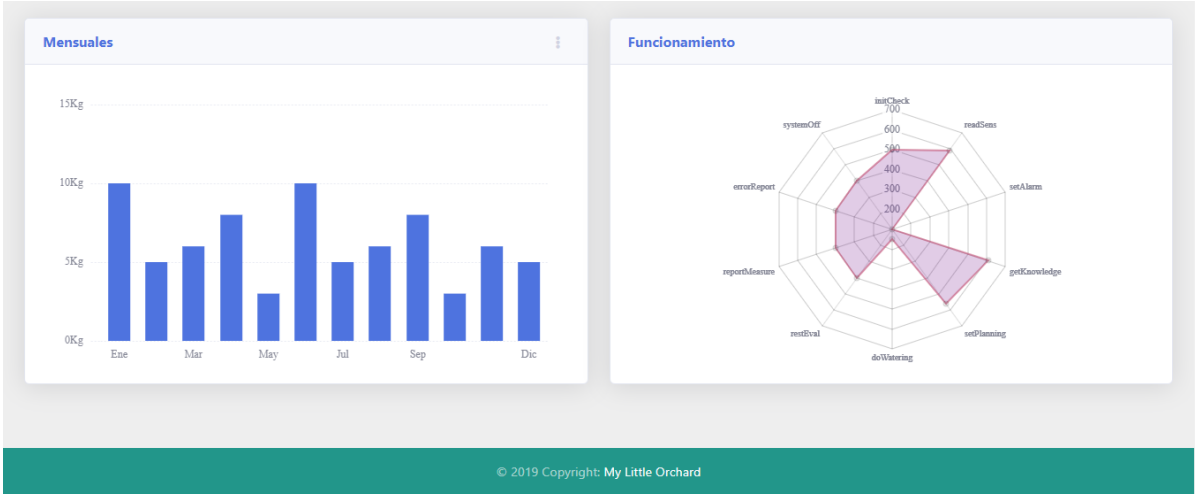

Figura 56 – Pie de página.

7.3 Interacción entre sistemas

La aplicación de control ejecuta toda su lógica en el cliente. La información mostrada por los gráficos y listados se carga dinámicamente por medio de scripts, desde Firebase y Particle Cloud. El código de la aplicación se incluye como anexo.

7.3.1 Interacción con Firebase

Se inicializa la aplicación con la configuración por defecto:

```
// Initialize the default App
Firebase.initializeApp(defaultAppConfig);
```

El SDK de Firebase permite a la aplicación controlar el estado de autenticación, asegurando el acceso a los datos de Firestore. Firebase Authentication proporciona, no sólo servicios de backend, sino también bibliotecas de interfaces de usuario elaboradas y listas para su uso.

Admite diversos tipos de identificación de usuario y es compatible con estándares como OAuth 2.0 y OpenID.

El espacio de nombres de Firebase Authentication es:

```
Firebase.auth().[…]
```

FirebaseUI Auth es una solución completa de autenticación directa, una biblioteca creada a partir del SDK de Firebase Authentication que proporciona flujos directos de IU para usar en tu App.

Se incluye en la App:

```
<script
src="https://cdn.Firebase.com/libs/Firebaseui/3.1.1/Firebaseui.js"></script>

<link type="text/css" rel="stylesheet"
href="https://cdn.Firebase.com/libs/Firebaseui/3.1.1/Firebaseui.css" />
```

Se habilita el acceso mediante mail y contraseña:

```
ui.start('#Firebaseui-auth-container', {

 signInOptions: [

   Firebase.auth.EmailAuthProvider.PROVIDER_ID

 ],

 // Other config options...

});
```

Y ya está listo el sistema de autenticación:

```
// Initialize the FirebaseUI Widget using Firebase.

var ui = new Firebaseui.auth.AuthUI(Firebase.auth());
```

El SDK de Firebase también es el encargado de las consultas al sistema de almacenamiento. Para incluir Firestore en la App:

Configurar el entorno:

```
<script src="https://www.gstatic.com/Firebasejs/4.12.1/Firebase.js"></script>

<script src="https://www.gstatic.com/Firebasejs/4.12.1/Firebase-
Firestore.js"></script>
```

Inicializar Firestore:

```
Firebase.initializeApp({
  apiKey: '### FIREBASE API KEY ###',
  authDomain: '### FIREBASE AUTH DOMAIN ###',
  projectId: '### CLOUD FIRESTORE PROJECT ID ###'
});
```

Consultar datos:

```
db.collection("micoleccion").get().then((querySnapshot) => {
    querySnapshot.forEach((doc) => {
        console.log(`${doc.id} => ${doc.data()}`);
    });
});
```

7.3.2 Interacción con Particle

El SDK de Particle facilita el acceso a las variables y funciones expuestas por el dispositivo. De forma general, el ciclo de trabajo es:

Incluir el SDK:

```
<script type="text/javascript" src="https://cdn.jsdelivr.net/npm/particle-api-
js@7/dist/particle.min.js">
</script>
```

Iniciar sesión en Particle:

```
var particle = new Particle();
var token;

particle.login({username: 'user@email.com', password: 'pass'}).then(
  function(data) {
```

```
    token = data.body.access_token;
  },
  function (err) {
    console.log('Could not log in.', err);
  }
);
```

Interactuar:

```
var fnPr = particle.callFunction({ deviceId: 'DEVICE_ID', name: 'laFuncion',
argument: 'ARGUMENTO:VALOR', auth: token });
```

7.4 Conclusión

Se ha completado el autómata y su aplicación de control. La herramienta está lista y cumple los requisitos marcados. A partir de este momento, el proyecto se centra en el testeo de la herramienta y su funcionamiento.

Capítulo 8

Pruebas y análisis

8.1 Introducción

Este capítulo se centra en la ejecución de pruebas de funcionamiento. Se ha testeado el firmware del autómata, los elementos de comunicación, y finalmente la aplicación de control. A continuación, se exponen los mecanismos de prueba empleados en cada caso.

8.2 Probando el autómata

Las pruebas del autómata han seguido un patrón común: la modificación del *setup()* y el *loop()* del script de ejecución, para incluir invocaciones a los métodos objeto de test en las que los valores de los parámetros se han preestablecido, y la comparación de la respuesta dada frente al valor correcto precalculado.

8.2.1 Pruebas y calibración de sensores

Las hojas de datos de los sensores GROVE muestran valores de referencia y procesos de calibración de los sensores. El código tipo de comprobación de sensores es:

```
// Se actualiza el valor de las variables de estado.
readSensors();
// Se actualiza el vector de estado.
checkEstado();
```

```
// Se actualiza el vector de alertas.

setAlertas();

// Mostramos el valor de los sensores.

//Particle.publish("Moisture", String(moisture), PRIVATE);

//delay(1000);

[...]

// Mostramos alertas y estados.

for (int i=0; i<NUM_SENS; i++){



                    Particle.publish(APP_NAME, "Flag: " + String(i) + " " +
String(flags[i].flag), PRIVATE);

                    delay(1000);



                    Particle.publish(APP_NAME, "MomentoFlag: " + String(i) + "
" + String(flags[i].momento), PRIVATE);

                    delay(1000);



                    Particle.publish(APP_NAME, "Alerta: " + String(i) + " Nivel:
" + String(alertas[i].nivel), PRIVATE);

                    delay(1000);

}
```

8.2.2 Pruebas unitarias de los métodos del firmware

Los métodos del firmware se han comprobado de forma individual. A modo de ejemplo, se incluye el código de prueba del método de encendido del autómata:

```
turnOnOff("On");

delay(1000);

turnOnOff("Off");

delay(1000);

turnOnOff("otracosa");
```

En los tests, el método publica información en consola como parte de su ejecución. El contraste de esa información con los valores de la llamada permite determinar la corrección de la ejecución del método.

8.2.3 Supervisión del ciclo de funcionamiento

Los métodos fueron testados individualmente de forma previa a la programación de la máquina de estados finitos. Posteriormente, se implementó el autómata y se mantuvo en ejecución durante días, simulando la comunicación con sistemas externos mediante métodos que siempre retornaban los mismos valores.

Las condiciones externas se modificaron de forma artificial, con lo que aseguró el comportamiento de la máquina de estados finitos y la corrección en la transición de estados.

En este momento se comprobó el funcionamiento y se calibró el caudal de la bomba de riego.

8.3 Probando la comunicación

La comunicación entre el autómata y los sistemas externos se testeó en dos fases:

8.3.1 Pruebas desde la consola

La consola de Google Cloud Platform permite simular el envío de mensajes a un tema. Se verificó la estructura del JSON que debía recibir la función suscriptora y se probó la ejecución de las funciones desde la consola. A modo de ejemplo, se muestra el JSON de un registro de estado:

{

"tempTierra":10.5,

"tempAire":11.1,

"humedadAire":12.0,

"luzVisible":13.0,

"luzIR":14.0,

"UVIndex":15.0,

"moisture":16.0,

"horasDeLuz":5

}

La publicación del mensaje en la consola desencadena la ejecución de la función suscriptora correspondiente y provoca la inserción de los parámetros como un objeto en la colección de estados del sistema de almacenamiento.

8.3.2 Pruebas desde el firmware

Después de comprobar las funciones suscriptoras se modificó el *loop()* del autómata para realizar publicaciones de eventos y descargar información externa. De esta forma se aseguró el funcionamiento de la integración entre plataformas. Parte de ese código de ejemplo se incluye aquí:

```
// Declaracion de funciones
void myHandlerAemet(const char *event, const char *data);
void myHandlerDatosLluvia(const char *event, const char *data);
void myHandlerSiar(const char *event, const char *data);
void myHandlerError(const char *event, const char *data);
```

```
// Configuramos el sistema
void setup() {

  hecho = false;
  URLDestino = "No_inicializado";
  endPoint = "No_inicializado";
  datosAemet = "No_inicializado";
  datosSiar = "No_inicializado";

  // Inicializamos las variables para actualizar la bd.
  sistema = "Sist01-Mod01";

  // Suscripciones a eventos
  Particle.subscribe("hook-response/aemet", myHandlerAemet, MY_DEVICES);
  Particle.subscribe("hook-response/datosLluvia", myHandlerDatosLluvia,
MY_DEVICES);
  Particle.subscribe("hook-response/siar", myHandlerSiar, MY_DEVICES);

  // El manejador de errores es el mismo para todos los webhooks
  Particle.subscribe("hook-error", myHandlerError, MY_DEVICES);

}

void loop() {

  if(!hecho){

    // Obtenemos infor externa
    getSiarInfo();
    getAemetInfo();
```

```
        // Guardamos la info en Firebase.
        sendFbAlertas();
        sendFbEstado();
        sendFbHito();
        sendFbRiego();

        hecho = true;
}}
```

Durante los test, los manejadores de las respuestas publican en Particle Cloud el resultado de su ejecución, lo que ayuda a comprobar la correcta ejecución del webhook.

8.4 Probando la aplicación de control

La aplicación de control muestra la información almacenada, supervisa el estado del autómata y permite invocar los métodos expuestos por el firmware. Debe adaptar su estructura gráfica al tamaño de la pantalla del dispositivo y ejecutarse en cualquier navegador web moderno.

8.4.1 Pruebas de usabilidad y funcionamiento

En las pruebas se ha utilizado:

1. Dispositivo portátil, con sistema operativo Windows 8.1 profesional y los navegadores: Chrome, Firefox y Edge. Tamaño de pantalla: 14”.
2. Terminal móvil, con sistema Android y los navegadores: Chrome y Firefox. Tamaño de pantalla: 5.5”,
3. Tablet, con sistema Android y los navegadores: Chrome y Firefox. Tamaño de pantalla: 10.1”
4. Terminal móvil, con sistema IOS 12 y el navegador Safari. Tamaño de pantalla: 4”.

En todos los casos se ha comprobado:

1. La adecuación de la interfaz al tamaño de pantalla.
2. La correcta presentación del contenido en pantalla.
    a. Barra de navegación.
    b. Secciones de detalle.
    c. Formularios modales.
3. La correcta ejecución de los scripts de actualización de elementos gráficos y de representación de datos.
4. El comportamiento de los scripts de integración con Firebase.
    a. Inicio y cierre de sesión.
    b. Lectura de valores en Firestore.
    c. Inserción de valores en Firestore.
    d. Actualización de valores en Firestore.
5. El comportamiento de los scripts de integración con Particle:
    a. Lectura remota de variables.
    b. Ejecución remota de métodos.

Las figuras siguientes muestran el diseño de la interfaz de la App de control:

La Figura 57 muestra la pantalla inicial. Es necesario identificarse para acceder a la aplicación.

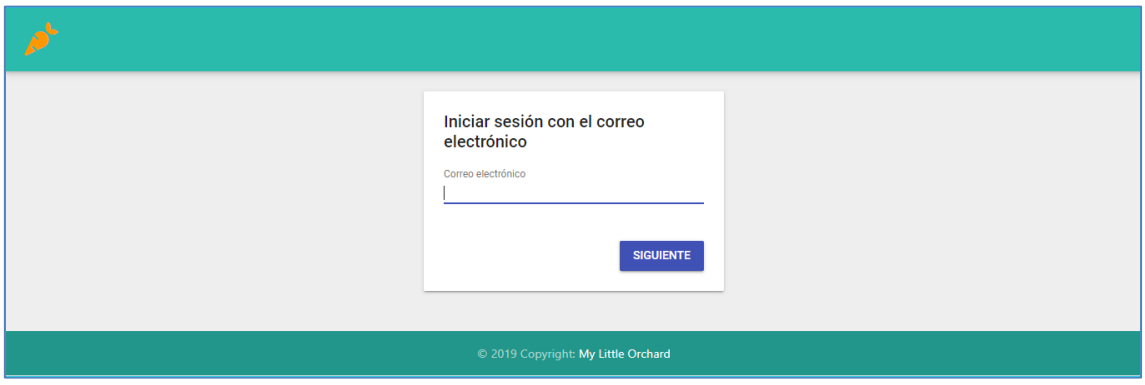

Figura 57 – Identificación.

La Figura 58 muestra uno de los formularios modales: el panel de alertas.

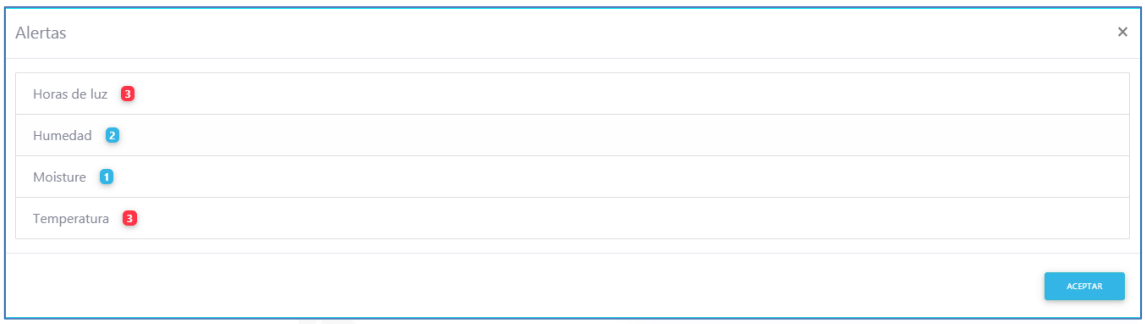

Figura 58 – Panel de alertas

La Figura 59 muestra una de las listas tabuladas: el listado de tareas.

Figura 59 – Listado de tareas.

La Figura 60 muestra los controles de visualización de estado del sistema.

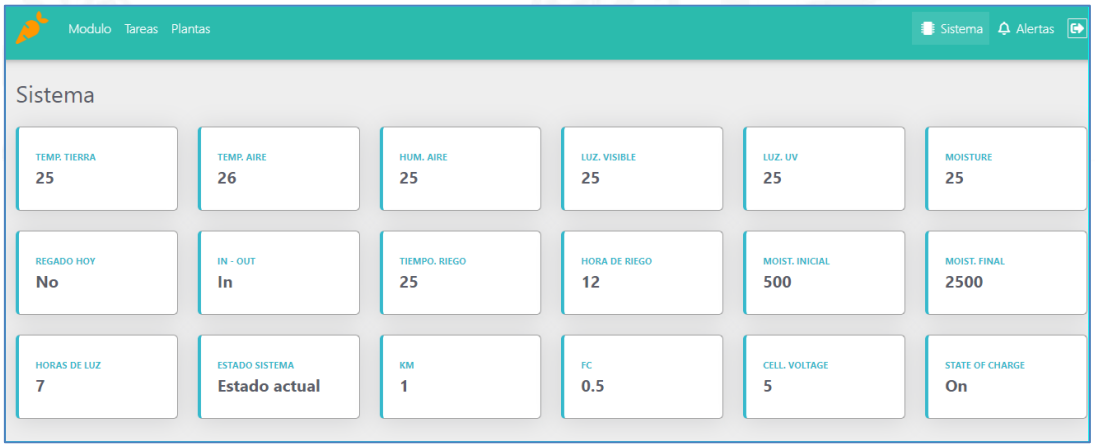

Figura 60 – Supervisión del sistema.

La Figura 61 muestra la pantalla de supervisión del estado del módulo de plantación.

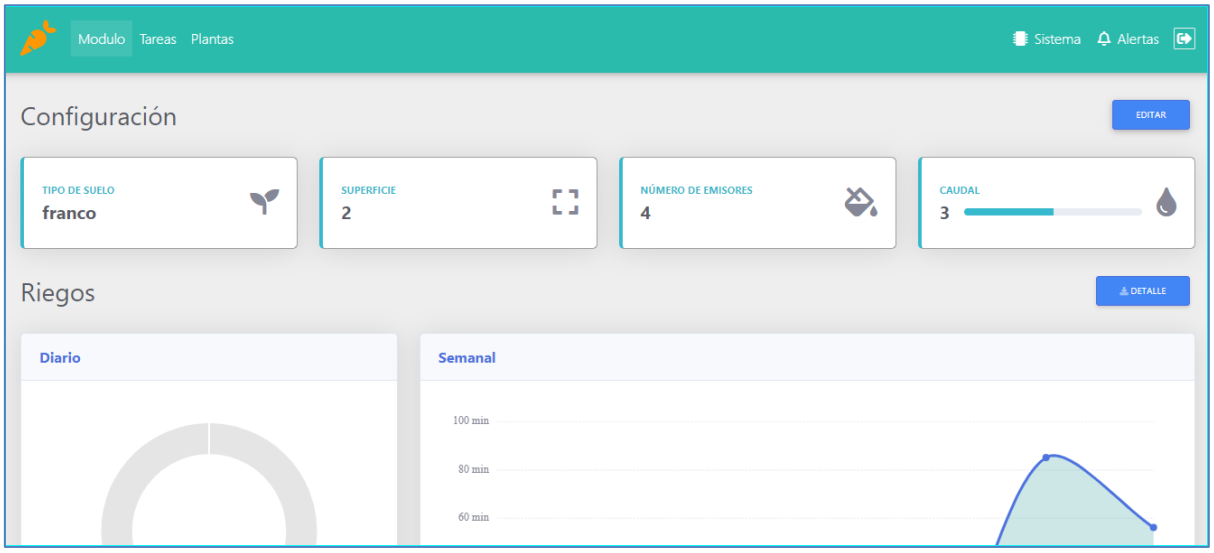

Figura 61 – Supervisión del módulo.

La ejecución de pruebas de uso ha resultado satisfactoria en cada uno de los dispositivos enumerados.

8.5 Pruebas finales. Un caso de uso real

Para la prueba final se ha creado una maqueta, compuesta por un único módulo de plantación de 30x20x14 cm, que se ha sembrado con hierba para gatos. Una lata de 500cl. incrustada en un lateral del módulo hace de depósito de agua y una instalación de microtubo flexible de 4 mm forma el sistema de riego.

La Figura 62 muestra la maqueta al completo.

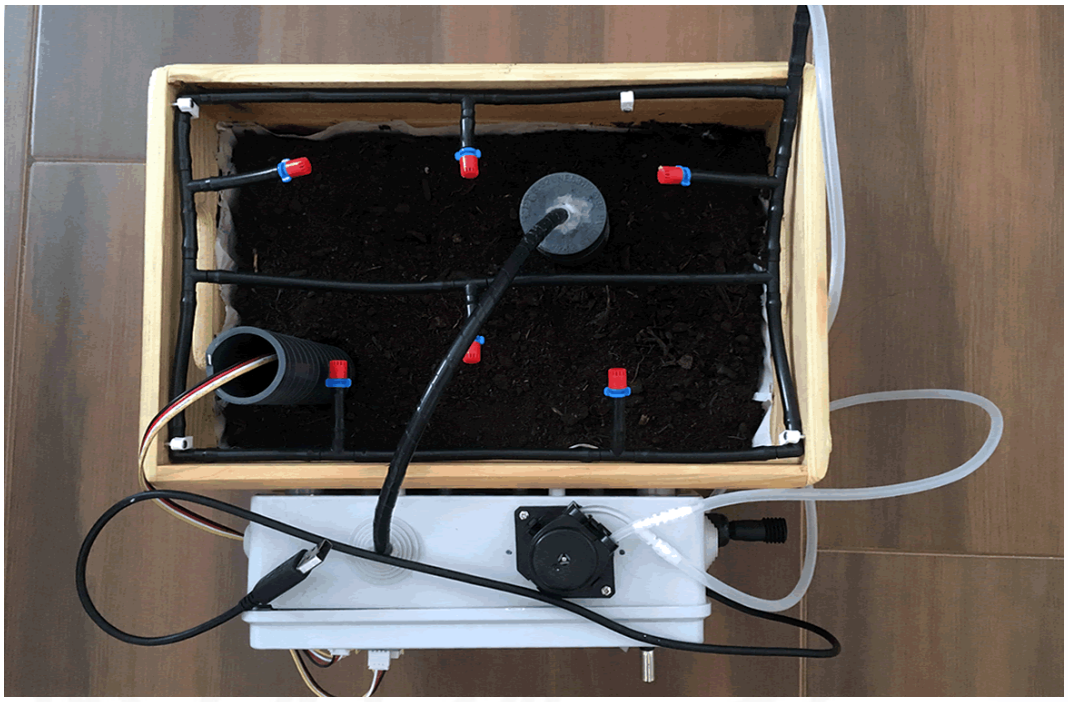

Figura 62 – Autómata y contenedor.

El autómata se ha empotrado en una caja estanca de 20x10x7 cm, con un grado de protección IP65. La caja se ha fijado en un lateral del contenedor de riego, permitiendo ubicar correctamente cada uno de los sensores. La Figura 63 muestra el Hardware ensamblado y montado.

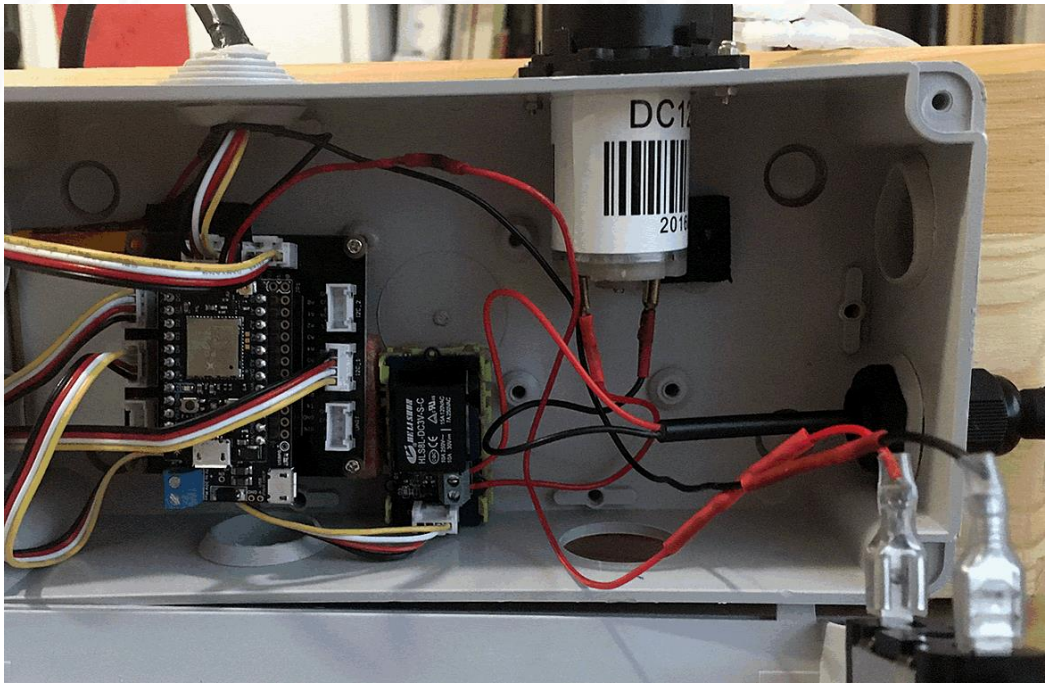

Figura 63 – Sistema Hardware.

El caso de uso se ha extendido varias semanas, durante las cuales se ha comprobado periódicamente:

1. El estado del sistema.
    a. De forma física.
    b. Mediante las consolas de administración de Google Cloud, Firebase y Particle Cloud.
    c. Haciendo uso de la aplicación de control.

2. La correcta interacción entre el autómata y la aplicación de control.
    a. Revisando la información generada por el autómata por medio de:
        i. Gráficos.
        ii. Tablas.
        iii. Otros elementos de representación de datos.
    b. Modificando la configuración del sistema.

3. La evolución del estado del módulo de plantación y su correlación con los datos aportados por el autómata.

A medida que ha transcurrido el tiempo, el autómata ha ido incluyendo su información diaria de estado y funcionamiento en Firestore. Cuando la cantidad de información ha sido suficiente, la aplicación de control ha permitido extraer conclusiones útiles sobre el estado del huerto y su evolución.

8.6 Conclusión

La ejecución de las pruebas del sistema confirma la corrección técnica del proyecto y demuestra su utilidad como herramienta de apoyo en la toma de decisiones relacionadas con el mantenimiento de un huerto urbano doméstico.

Capítulo 9

Planificación y gestión del proyecto

9.1 Introducción

El proyecto se ha extendido considerablemente en el tiempo. La falta de experiencia en el desarrollo de proyectos IoT, el afán por elegir el componente adecuado en cada momento y la utilización de elementos hardware y software de última generación han dificultado su consecución.

9.2 Planificación del proyecto

El proyecto ha sido dividido en fases, de acuerdo con la naturaleza de las tareas que comprenden. La ejecución de las fases se ha solapado en el tiempo, tratando de minimizar la repetición de tareas; no obstante, ha sido necesario terminar algunas fases antes de comenzar otras.

9.2.1 Investigación teórica

En la primera fase del proyecto se hizo un estudio pormenorizado del estado del arte en el desarrollo de proyectos IoT: microcontroladores, plataformas IoT, plataformas de desarrollo de aplicaciones Cloud, mecanismos de comunicación y almacenamiento, comunidades de desarrollo, etc. Además, se dedicó una gran cantidad de tiempo a comprender los sistemas de riego desde un punto de vista científico. Algo que determinó el desarrollo de nuestro autómata.

9.2.2 Selección y adquisición de elementos hardware y software

Se exploraron alternativas como: Google App Engine, Angular e incluso la librería de componentes web Polymer. Se descartaron por su complejidad y porque no mejoraban el resultado final de nuestro proyecto sustancialmente.

En esta fase del proyecto se decidió utilizar:

- Particle Cloud, el microcontrolador PHOTON y Workbench IDE.
- Plataforma de sensores GROVE.
- Google Firebase.
- Bootstrap 4, JQuery y el resto de las librerías JavaScript.

Se adquirieron todos los elementos enumerados en el Capítulo 3 y se crearon las cuentas necesarias en Firebase, Google Cloud, Particle y los servicios SIAR y AEMET Open Data.

9.2.3 Diseño de la interacción entre sistemas

En esta fase se estableció la división de componentes del sistema:

1. Sistema hardware: Programable, portable y con acceso wifi a Particle Cloud.
2. Autómata de estados finitos: Ejecuta la lógica del sistema.
3. Mecanismos de comunicación: Soportados por Particle Cloud.
4. Sistema de almacenamiento: Base de datos NoSQL en tiempo real.
5. Aplicación de control: Responsive y serverless.

Finalmente, se establecieron y verificaron (de forma teórica) los requisitos de comunicación e interacción entre sistemas.

9.2.4 Diseño y montaje hardware

En esta fase fue diseñando el sistema hardware, sus elementos y conexiones. Además, se instaló el dispositivo en la caja estanca y se comprobó su funcionamiento.

9.2.5 Diseño y programación del autómata

Se diseñó el autómata finito, sus estados y sus condiciones de transición. Finalmente, fue desarrollado el firmware del autómata.

Ha sido necesario desarrollar una gran cantidad de código auxiliar para dar soporte a las operaciones de configuración, control de estado, intercambio de información, almacenamiento en memoria, etc.

9.2.6 Diseño e implementación de mecanismos de comunicación

En esta fase fueron diseñados e implementados los webhooks de comunicación y las integraciones de Particle Cloud con Google Cloud. Fueron diseñadas e implementadas las funciones suscriptoras que actualizan Firestore y el resto de los elementos de comunicación.

9.2.7 Diseño e implementación del sistema de almacenamiento

En esta fase, que se solapó temporalmente con la de diseño y programación del autómata, fue diseñado e implementado todo el sistema de almacenamiento en Firestore. Lo esencial aquí era conseguir almacenar la información necesaria para que la aplicación de control cumpliese su objetivo.

9.2.8 Diseño y programación de la aplicación de control

Se diseñó e implementó la aplicación de control. Al tratarse de una aplicación web, ha sido necesario desarrollar los elementos visuales y los scripts de interacción de forma separada. La aplicación de control es la interfaz del sistema. Abstrae la complejidad subyacente y muestra al usuario la información almacenada en forma de gráficos y tablas.

9.2.9 Test global del sistema

El sistema ha sido testeado en cada fase, asegurando la corrección de cada subsistema; no obstante, se ha dedicado un apartado a la prueba final de la herramienta, utilizando para ello un módulo de plantación diseñado para ello.

9.2.10 Documentación del proyecto

Esta se ha solapado con el resto de las fases: se ha documentado de manera informal cada fase del proyecto, lo que ha servido de referencia y apoyo teórico, y se ha redactado la versión final de la memoria al finalizar el desarrollo del proyecto.

9.3 Gestión del proyecto

En cada fase del proyecto ha sido necesario dedicar una gran cantidad de tiempo a tareas de investigación. En diversas ocasiones se han explorado alternativas que no han sido implementadas, dando lugar a períodos de reflexión. Estos períodos, en apariencia improductivos, han resultado de gran valor formativo.

9.3.1 Planificación temporal

El desarrollo del proyecto ha ido acumulando retraso en cada fase. Al no ser posible la dedicación total y exclusiva han surgido múltiples tiempos muertos, tras los que resultaba costoso volver al punto en el que se había interrumpido el proceso. Como puede observarse, el proyecto estaba planteado a 12 meses, una duración que prácticamente se ha duplicado.

La Tabla 7 muestra la planificación temporal realizada y el desvío acumulado. La unidad temporal es el mes.

Se muestra el mes de comienzo previsto para cada actividad y la duración en meses estimada para completar la actividad. En las columnas siguientes muestra los datos reales: mes de inicio real de la actividad y duración real de la actividad. La última columna muestra el tanto por ciento de la actividad completado.

ACTIVIDAD	INICIO DEL PLAN	DURACIÓN DEL PLAN	INICIO REAL	DURACIÓN REAL	COMPLETADO
Investigación teórica	1	5	1	20	100%
Selección de componentes	2	1	2	2	100%
Diseño de la interacción	1	2	2	2	100%
Prototipo hardware	3	3	4	2	100%
Prototipo funcional	6	3	7	6	100%
Mecanismos de	9	2	11	4	100%
Sistema de almacenamiento	6	2	6	2	100%
App de control	11	2	15	5	100%
Test global	13	1	20	1	100%
Documentación	10	2	3	20	100%

Tabla 7 – Planificación temporal.

La Figura 64 es una representación gráfica del contenido de la Tabla 7.

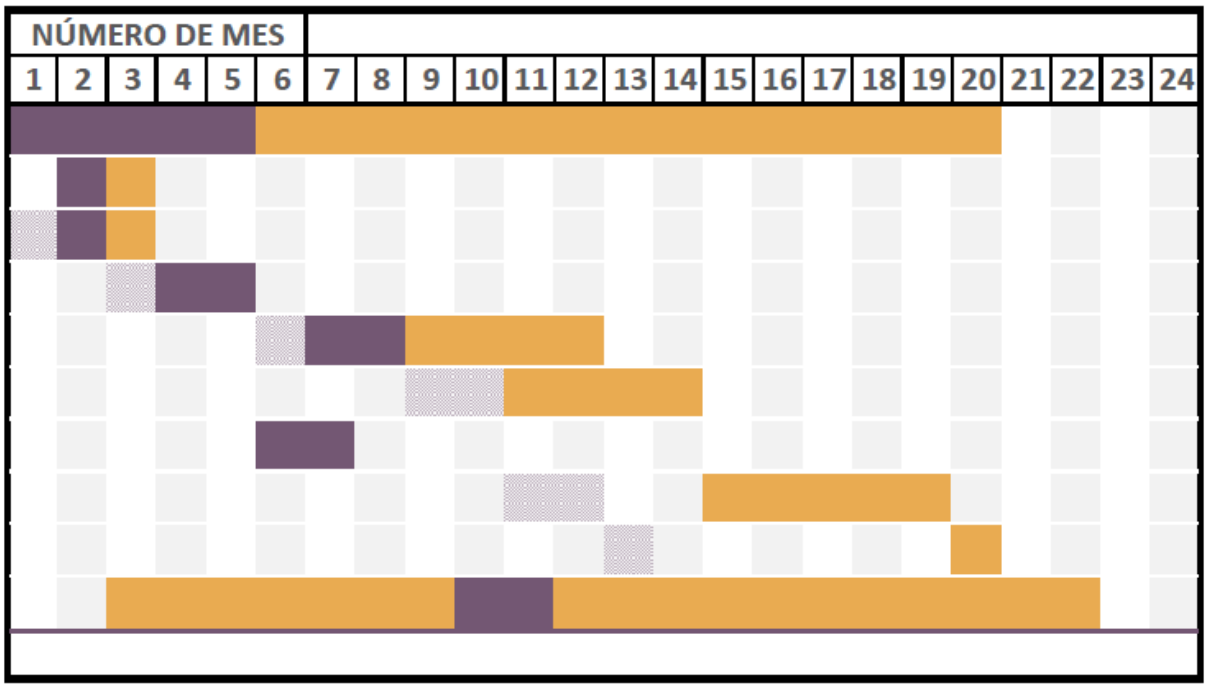

Figura 64 – Diagrama Gantt.

9.3.3 Costes de proyecto

Los costes temporales de realización del proyecto prácticamente se han duplicado. De las seiscientas cuarenta horas previstas hemos pasado a más de novecientas horas reales de trabajo. No todo el trabajo ha sido productivo: muchas horas se han dedicado a mejorar la comprensión de los fundamentos teóricos sobre los que se ha desarrollado el proyecto.

La Tabla 8 muestra la cantidad de horas dedicadas a cada actividad. Estimando un coste de 60 Euros/hora, la mano de obra del proyecto tiene un coste final aproximado de: 920 horas * 60 Euros = 55.200,00 €

ACTIVIDAD	Previsto	Real
Investigación teórica	100	200
Selección de componentes	20	20
Diseño de la interacción	40	40
Prototipo hardware	60	40
Prototipo funcional	80	120
Mecanismos de comunicación	40	100
Sistema de almacenamiento	40	40
App de control	120	200
Test global	40	40
Documentación	100	120

Tabla 8 - Horas de trabajo

Los costes económicos se han visto reducidos: el planteamiento inicial contemplaba una instalación de mayor envergadura, basada en una alternativa hardware distinta, que a la postre ha resultado inadecuada. La utilización de Firebase como plataforma Cloud también ha abaratado el proyecto, ya que la versión gratuita ha resultado suficiente para cumplir los requisitos de la especificación inicial. No se ha invertido en productos o servicios software.

El precio de adquisición de los elementos hardware empleados ha sido de 415,00 Euros, por lo que el coste total del proyecto ha sido de 55.615,00 Euros.

9.4 Conclusión

La consecución de los objetivos marcados al inicio del proyecto ha resultado ser mucho más costosa de lo que parecía. Las revisiones de documentación y proyectos similares que se hicieron para preparar el anteproyecto transmitían una idea equivocada de lo que supone abordar un proyecto así. Quizá el mayor error fue menospreciar el conocimiento teórico necesario para abordar el proyecto con garantías de éxito, dada la amplia variedad de temas abordados y la necesidad de conjuntarlos en una única herramienta. A pesar de las dificultades, el proyecto ha sido ejecutado con éxito y ha supuesto una experiencia de gran valor formativo.

Capítulo 10

Conclusiones y trabajos futuros

10.1 Introducción

Se ha creado una herramienta útil y sencilla, que aporta valor al usuario y facilita el mantenimiento de un huerto doméstico, mediante la monitorización constante de los contenedores de plantación y su entorno. Almacena y muestra gráficamente esa información, y además es configurable: permite conocer y modificar los parámetros de funcionamiento del autómata.

10.2 Conclusiones

El objetivo inicial del proyecto era la creación de una herramienta que sirviese de apoyo en la toma de decisiones durante las tareas de mantenimiento de un huerto doméstico. Este objetivo se ha conseguido.

Conceptualmente, el autómata debía ejecutarse en un microcontrolador, interactuar con sistemas externos y ser configurado de forma remota. Cualquier dispositivo en condiciones de acceder a Internet y ejecutar un navegador web moderno debía servir como cliente.

La ejecución del proyecto ha supuesto, en primer lugar, un proceso formativo y de investigación considerable, en el que la selección de tecnologías, elementos hardware y servicios utilizados no ha sido fácil. Se ha llevado a cabo un riguroso estudio del estado del arte antes de tomar cada decisión.

La elección más importante, sin duda, ha sido la plataforma IoT con la que trabajar. Después de meses de trabajo hubo que cambiar la plataforma IoT inicial, que a priori parecía ideal, lo que ha modificado todo el desarrollo final y ha retrasado considerablemente la entrega del proyecto.

Ha sido una experiencia valiosa, en la que se ha podido comprobar cómo una elección inicial equivocada a la hora de crear un producto puede comprometer seriamente el éxito del proyecto, en caso de que éste tenga un plazo improrrogable de entrega.

La elección del entorno PaaS en el que alojar el resto de los componentes del sistema, sin resultar tan determinante como la decisión anterior, ha marcado el proyecto.

El resultado ha sido una herramienta completa y funcional, compuesta por:

1. Un dispositivo que puede ser trasladado entre ubicaciones y que únicamente necesita una conexión Wifi para obtener predicciones y enviar información. Puede funcionar conectado a la red eléctrica o desconectado gracias a una batería.

2. El firmware del dispositivo, capaz de:

    a. Inicializar y comprobar el hardware.
    b. Mantener su configuración en la EEPROM.
    c. Ejecutar el autómata.
    d. Detectar condiciones de error y/o detención y actuar en consecuencia.
    e. Intercambiar información con otros sistemas.
    f. Ser configurado.

3. Mecanismos de intercambio de información entre sistemas.

4. Una aplicación web:
    a. Responsive.
    b. Basada en gráficos, tablas y controles visuales.
    c. Incorpora un mecanismo de autenticación de usuarios.
    d. Permite supervisar el estado actual del módulo.
    e. Ayuda a analizar la información almacenada a lo largo del tiempo por el autómata.
    f. Incorpora un sistema de alertas.
    g. Ayuda a llevar el control de las tareas del huerto.

La herramienta tiene una utilidad fuera de toda duda; no obstante, presenta limitaciones de carácter técnico y funcional. La mayoría de ellas relacionadas con el alcance de este proyecto.

10.3 Trabajos futuros

Al tratarse de una herramienta compuesta por varios sistemas, resulta sencillo plantear mejoras en cada uno de ellos.

10.3.1 Mejoras en el hardware

Se han empleado elementos hardware para el prototipado simple de dispositivos IoT. Los sensores GROVE son útiles y funcionales; pero son componentes de baja durabilidad y resistencia. Al tratarse de un proyecto de monitorización del medio ambiente, el hardware debería permanecer a la intemperie sin dañarse. Si se espera que la herramienta sea útil y duradera, es necesario modificar los sensores. El código del autómata debería ser corregido y adaptado; pero no es una tarea excesivamente compleja, ya que los componentes básicos de los sensores (y por lo tanto el controlador) suelen ser los mismos, variando únicamente el empaquetado de cada fabricante.

10.3.2 Mejoras en el firmware del autómata

El código del autómata monitoriza múltiples estados, pero actúa únicamente sobre la humedad del sustrato del contenedor de riego. De forma evidente, podemos implementar mecanismos de actuación sobre el resto de los parámetros controlados por el sistema de alerta. Es una opción que se ha explorado, aunque al quedar fuera del alcance del proyecto no se ha profundizado en ella. Algunas aproximaciones en este sentido son:

1. Se ha valorado la inclusión de capacidad multitarea en la gestión de los actuadores del autómata.

2. También se ha valorado separar el firmware de los dispositivos en dos versiones:
    a. Una para el control y monitorización de estado.
    b. Otra para el manejo de los actuadores y la transmisión de datos.

3. La coordinación entre dispositivos puede ser realizada a través de los mecanismos ofrecidos por Particle Cloud.

Este apartado ofrece múltiples puntos de mejora, pudiendo pasar de un sistema basado en una única versión del firmware, a un sistema multitarea complejo y distribuido, basado en distintas versiones del firmware adaptadas al rol de cada dispositivo. Todo ello sin necesidad de cambiar el sistema de almacenamiento o la aplicación de control.

10.3.3 Mejoras en la aplicación de control

La funcionalidad y usabilidad de la aplicación de control son las que deben ser; no obstante, la arquitectura es muy mejorable:

1. Mediante el uso de un framework JavaScript avanzado como Angular.
2. Empleando componentes web en su diseño.
3. Convirtiéndola en una SPA.
4. Empleando conceptos de las PWA que mejoren la interacción en dispositivos móviles.

Todas estas mejoras no alterarían el comportamiento de la aplicación; sin embargo, son las que más valor didáctico tienen.

Bibliografía

Libros

- Miguel Ángel Acera García, 2012. *CSS3 – Guía práctica*. España: Anaya Multimedia.
- Alonso Álvarez García, 2010. *HTML5 – Guía práctica*. España: Anaya Multimedia.
- Rick As, 2018. *Perfil de desarrollador Github.* Disponible en: https://github.com/rickkas7
- Massimo Banzi, 2012. *Introducción a Arduino*. España: Anaya Multimedia – O'Reilly.
- Scott Fitzgerald y Michael Shiloh (ed.), 2013. *Arduino Projects Book*. Torino: Arduino.
- Gustavo Gonnet, 2018. *Perfil de desarrollador Github*. Disponible en: https://github.com/gusgonnet
- Martin Kalin, 2013. *Java Web Services up and Running*. 2ª edn. United States Of America: O'Reilly.
- Brian W. Kernighan y Dennis M. Ritchie, 1991. *El lenguaje de programación C.* 2ª edn. España: Prentice-Hall Hispanoamericana.
- Liu M.L., 2004. *Computación distribuida. Fundamentos y aplicaciones*. España: Pearson.
- David Sawyer McFarland, 2012. *Programación JavaScript y JQuery.* 2ª edn. España: Anaya Multimedia – O'Reilly.

Páginas web

- FAO, 2018. *Cuaderno 56*. Disponible en: http://www.fao.org/3/a-x0490s.pdf
- Firebase, 2019. *Documentación para desarrolladores*. Disponible en: https://firebase.google.com/
- Google Developers, 2019. *Referencias varias*. Disponible en: https://developers.google.com/web/
- Material Design Bootstrap, 2018. *Documentación JQuery*. Disponible en: https://mdbootstrap.com/docs/jquery/
- MiriadaX, 2018. *Diseño agronómico de riego localizado*. 4ª edn, disponible en: https://miriadax.net/web/diseno-agronomico-del-riego-localizado-4-edicion-/inicio
- Particle, 2019. *Documentación para desarrolladores*. Disponible en: https://docs.particle.io/
- SIAR, 2018. *Sistema de información al regante del Ministerio de Agricultura, pesca y alimentación*. Disponible en: http://www.siar.es
- SeeedStudio, 2018. *Hojas de producto*. Disponible en: https://www.seeedstudio.com
- Udacity, 2018. *Asynchronous JavaScript Requests*. Disponible en: https://www.udacity.com/
- Udacity, 2018. *Authentication & Authorization: OAuth*. Disponible en: https://www.udacity.com/
- Udacity, 2018. *ES6*. Disponible en: https://www.udacity.com/

- Udacity 2018. *HTTP & Web Servers*. Disponible en: https://www.udacity.com/
- Udacity, 2018. *Intro to Progressive Web Apps*. Disponible en: https://www.udacity.com/
- Udacity, 2018. *JavaScript Design Patterns*. Disponible en: https://www.udacity.com/
- Udacity, 2018. *JavaScript Promises*. Disponible en: https://www.udacity.com/
- Udacity, 2018. *Mobile Web Development*. Disponible en: https://www.udacity.com/
- Udacity, 2018. *Object-Oriented JavaScript*. Disponible en: https://www.udacity.com/
- Udacity, 2018. *Responsive Web Design Fundamentals*. Disponible en: https://www.udacity.com/
- Udacity, 2018. *Web Development*. Disponible en: https://www.udacity.com/
- W3SChools, 2018. *Referencias varias*. Disponible en: https://www.w3schools.com/

Listado de abreviaturas

SIAR: Servicios de información al regante.

AEMET: Agencia española de meteorologías.

FaaS: Función como servicio.

PaaS: Plataforma como servicio.

IaaS: Infraestructura como servicio.

SPA: Single Page Application.

PWA: Progressive Web App.

ETo: Evapotranspiración de referencia.

Kc: Constante de cultivo.

ETc: Evapotranspiración del cultivo.

P: Precipitación.

Δ Hs: Incremento de la humedad del suelo entre dos riegos.

Wc: Ascenso capilar del agua.

NNr: Necesidades netas de riego.

SDK: Software development kit.

CSS: Cascading style sheets.

HTML: Hypertext markup language

CRUD: Operaciones de creación, lectura, actualización y borrado de datos.

Anexos

Anexo I. Código del autómata.

```
/*****************************************************************************
* Proyecto:    AutomataFinitoDeRiego
* Descripcion: Sistema autonomo de riego basado en un automata de
*              estados finitos.
* Autor:       Marcos Colmenero Fernandez.
* Fecha:       2018 - 2020
*****************************************************************************/

// Incluimos los archivos necesarios para trabajar con el hardware.
#include <elapsedMillis.h>
#include <FiniteStateMachine.h>
#include <Adafruit_DHT_Particle.h>
#include <Adafruit_SI1145.h>
#include <PowerShield.h>
#include <JsonParserGeneratorRK.h>
#include <SparkJson.h>
#include <math.h>

// Configuramos el sistema
SYSTEM_THREAD(ENABLED);

// Definimos algunos parametros generales.
#define APP_NAME "MiHuertoTool"
#define NUM_SENS 4
String VERSION = "Version 0.4";
String MODULO = "Sist01-Mod01";
const int TIME_ZONE = +2;

// 10 muestras por cada lectura. Calculamos la media.
const int NUMBER_OF_SAMPLES = 10;

// Permite desactivar el automata.
#define MODO_ON "On"
#define MODO_OFF "Off"
String mode = MODO_OFF;

// Si la instalacion es de interior nunca llueve.
#define MODO_IN "In"
#define MODO_OUT "Out"
String inOut = MODO_OUT;

// Constantes de estado del automata.
#define INIT_CHECK "Iniciando_sistema"
#define READ_SENS "Comprobando_estado"
#define SET_PLANNING "Planificando_riego"
#define DO_WATERING "Regando"
#define REST_EVAL "Reposo_tras_riego"
#define ERROR_REPORT "Estado_de_error"
#define SYSTEM_OFF "Sistema_desactivado"

// Mantiene el estado actual del sistema.
String estadoSistema;

// Traducciones utiles para trabajar con intervalos.
#define MILISEG_A_HORAS 3600000
#define MILISEG_A_MINUTOS 60000
#define MILISEG_A_SEGUNDOS 1000

// Tiempo de espera en las llamdas a los webhooks.
const unsigned long MAX_TIME_AEMET_WAIT = 60 * MILISEG_A_SEGUNDOS;
const unsigned long MAX_TIME_FIREBASE_WAIT = 30 * MILISEG_A_SEGUNDOS;

// Tiempo necesario para actualizar el nivel de alerta. En horas.
// Por defecto: 24 horas. De esta forma ellapsedmillis no llega al limite.
#define INTERVALO_DE_ALERTA 24

// Cantidad de luz necesaria para considerar que es de dia.
// Hay que calibrar esto.
const double LUZ_DE_DIA = 265.0;
```

```
const double UV_DE_DIA = 0.1;

// Temporizadores para monitorizar el tiempo de riego y horas de sol.
elapsedMillis intervaloChequeo;
elapsedMillis intervaloRiego;
elapsedMillis intervaloReposo;
elapsedMillis intervaloHorasLuz;
elapsedMillis intervaloPublicacion;
elapsedMillis intervaloMuestreo;

// Datos de inicializacion del sensor DHT.
#define DHTTYPE DHT22

// Tiempo asignado al estado de inicio. En segundos.
const int initTimeout = 10;

// Tiempo de lectura. En segundos.
// 10 segundos. Lee en cada loop, independientemente del estado en que se encuentre.
const int readTime = 10;

// Tiempo de publicacion. En minutos.
// Publica en cada loop, independientemente del estado en que se encuentre.
int publishTime = 15;

// Tiempo de reposo. En minutos.
// Por defecto: 20 minutos.
int restTime = 20;

// ID photon.
// No lo almacenamos. Lo obtenemos de particle.
String idPhoton = "No_inicializado";

// Comprobantes de inicializacion del sistema: umbrales y datos del modulo.
bool setupModulo;
bool setupUmbrales;
bool setupAlertas;
bool setupHW;

// Tipo de suelo.
enum tipoSuelo
{
        gravoso = 0,
        arenoso = 1,
        franco = 2,
        arcilloso = 3,
};

tipoSuelo elSuelo;

// Predicciones
int probLluvia;
float ETo;

// Datos de riego. 9 bytes.
int riegoFijo;
double Km; // Cte del modulo.
double Fc; // Factor de correcion por encharcamiento.

// Datos para el calculo de riego.
int Ne;
int q; // litros/minuto.
float m2;
float eficiencia; // Depende del tipo de suelo.

// Datos de la ventana de riego.
int elMes;
int inicioVentana;
int finVentana;

// Datos de riego efectivo.
double tiempoDeRiego; // En minutos.
int horaDeRiego;         // Valor numerico entre 0 y 23.
bool regadoHoy;

// Lineas de control de riego. Dependen del tipo de suelo.
float saturacion;
float capacidadCampo;
```

```
float ptoDeRiego;
float marchitez;

// Lineas de evaluacion de riego.
double moistureInicial;
double moistureFinal;
bool encharcado;
bool marchitando;

// Variables para el intercambio de datos
String URLDestino;
String endPoint;
String datosAemet;
String datosSiar;

// Tipos de dato y variables para almacenar estados, umbrales y alertas.
enum tipoAlerta
{
        SinAlerta = 0,
        RiesgoBajo = 1,
        RiesgoMedio = 2,
        RiesgoAlto = 3,
        RiesgoCritico = 4,
};

enum tipoSensor
{
        tempAireS = 0,
        humAireS = 1,
        tempTierraS = 2,
        moistureS = 3,
        luzVisibleAireS = 4,
        luzIRAire = 5,
        UVIndexAire = 6,
};

enum tipoUmbral
{
        umbralTemp = 0,
        umbralHumedad = 1,
        umbralHorasLuz = 2,
        umbralMoisture = 3,
};

// Mantenemos un vector con el estado actual de cada sensor.
struct flagDeEstado
{
        bool flag;
        elapsedMillis momento;
};

flagDeEstado flags[NUM_SENS];

// Mantenemos un vector de alertas. 16 bytes.
struct alerta
{
        tipoAlerta nivel;
};

alerta alertas[NUM_SENS];

// Mantenemos un vector con los umbrales de funcionamiento. 8 bytes.
struct umbral
{
        float max;
        float min;
};

umbral umbrales[NUM_SENS];

// Variables con el valor de los sensores. Almacenan el estado.
double tempTierra;
double tempAire;
double humedadAire;
double luzVisible;
double luzIR;
```

```
double UVIndex;
double moisture;
int horasDeLuz;

// Variables con el estado de la bateria.
double cellVoltage;
double stateOfCharge;

// Variable que contiene la descripcion del ERROR.
String tipoError;

#define EEPROM_VERSION 123
#define EEPROM_ADDRESS_1 0
#define EEPROM_ADDRESS_2 46
#define EEPROM_ADDRESS_3 79

// Estructura de memoria para almacenar datos del modulo y configuracion de riego.
// Los umbrales y alertas tambien se almacenan. 45 bytes.
struct EepromMemoryStructure
{
        uint8_t version = EEPROM_VERSION; // 1 byte.
        int inOut;                                              // 4 byte.
        int mode;                                               // 4 byte.
        float m2Eeprom;                                         // 4 bytes.
        double KmEEprom;                                // 8 bytes.
        double FcEEprom;                                // 8 bytes.
        int elSueloEeprom;                              // 4 bytes.
        int qEeprom;                                    // 4 bytes.
        int NeEeprom;                                   // 4 bytes.
        int riegoFijoEEprom;                    // 4 byte.
};
EepromMemoryStructure eepromMemory;

// Asignacion pines - sensores.
// A0 : Sensor de temperatura GROVE. Tierra.
// A4 : Sensor de moisture GROVE. Tierra.
// D1 : Rele de activacion de la bomba de riego.
// D2 : DHTPIN. Sensor de humedad/temperatura. Aire.
// I2C1 : Sensor de luz. GROVE. Aire.

const int tempSensor = A0;
const int moistureSensor = A4;
const int releBomba = D4;
const int tempHumSensor = D2;
const int pinLed = D7;

// Variables de control de dispositivos.
DHT dht(tempHumSensor, DHTTYPE);
Adafruit_SI1145 uv = Adafruit_SI1145();
PowerShield batteryMonitor;

// Parser JSON dinamico
JsonParser parser;

// Declaramos las funciones de entrada, actualizacion y salida de cada estado.
State InitCheck = State(InitCheckEnterFunction, InitCheckUpdateFunction,
InitCheckExitFunction);
State ReadSens = State(ReadSensEnterFunction, ReadSensUpdateFunction,
ReadSensExitFunction);
State SetPlanning = State(SetPlanningEnterFunction, SetPlanningUpdateFunction,
SetPlanningExitFunction);
State DoWatering = State(DoWateringEnterFunction, DoWateringUpdateFunction,
DoWateringExitFunction);
State RestEval = State(RestEvalEnterFunction, RestEvalUpdateFunction,
RestEvalExitFunction);
State ErrorReport = State(ErrorReportEnterFunction, ErrorReportUpdateFunction,
ErrorReportExitFunction);
State SystemOff = State(SystemOffEnterFunction, SystemOffUpdateFunction,
SystemOffExitFunction);

// Declaramos las funciones que no declara el entorno.
tipoAlerta intToAlerta(int alerta);
int alertaToInt(tipoAlerta alerta);
int numHoras(elapsedMillis momento);
tipoAlerta subirNivelAlerta(tipoAlerta nivel);
tipoAlerta bajarNivelAlerta(tipoAlerta nivel);
tipoUmbral intToUmbral(int umbral);
```

```
int umbralToInt(tipoUmbral umbral);
tipoSensor intToSensor(int sensor);
int sensorToInt(tipoSensor sensor);
tipoSuelo intToTipoSuelo(int sueloInt);
int tipoSueloToInt(tipoSuelo suelo);
String intToInOut(int inOut);
int inOutToInt(String inOut);
String intToMode(int mode);
int modeToInt(String mode);

// Declaramos los manejadores de evento.
void myHandlerAemet(const char *event, const char *data);
void myHandlerDatosLluvia(const char *event, const char *data);
void myHandlerSiar(const char *event, const char *data);
void myHandlerError(const char *event, const char *data);

// Variable de control del automata.
FSM automataFinito = FSM(InitCheck);

void setup()
{

        // Publicamos los datos de la aplicacion al inicio.
        Particle.publish(APP_NAME, VERSION, PRIVATE);
        delay(1000);

        // Registramos las variables de estado en particle cloud.
        Particle.variable("Temp_Tierra", tempTierra);
        Particle.variable("Temp_Aire", tempAire);
        Particle.variable("Humedad_Aire", humedadAire);
        Particle.variable("Luz_Visible", luzVisible);
        Particle.variable("Luz_UV", UVIndex);
        Particle.variable("Moisture", moisture);
        Particle.variable("Regado_Hoy", regadoHoy);
        Particle.variable("In_Out", inOut);
        Particle.variable("Prob_Lluvia", probLluvia);
        Particle.variable("Tiempo_Riego", tiempoDeRiego);
        Particle.variable("Hora_Riego", horaDeRiego);
        Particle.variable("Moisture_Ini", moistureInicial);
        Particle.variable("Moisture_Fin", moistureFinal);
        Particle.variable("Horas_Luz", horasDeLuz);
        Particle.variable("Estado_AF", estadoSistema);
        Particle.variable("Km", Km);
        Particle.variable("Fc", Fc);
        Particle.variable("On_Off", mode);
        Particle.variable("Carga", stateOfCharge);
        Particle.variable("ERROR_Type", tipoError);

        // Registramos las funciones de control en particle cloud.
        Particle.function("Turn_On_Off", turnOnOff);  // PARAMETRO [On-Off]
        Particle.function("Set_InOut", setInOut);             // PARAMETRO [In-Out]
        Particle.function("Set_Modulo", setModulo);           // PARAMETRO [JSON]
        Particle.function("Set_Umbrales", setUmbrales);       // PARAMETRO [JSON]
        Particle.function("Reset_Riego", resetRiego);         // 1, 1, 0.
        Particle.function("Set_Riego", setRiego);             // PARAMETRO [FIJO ||
DINAMICO]
        Particle.function("Reset_Estado", resetEstado);       // Alertas y flags.
        Particle.function("Set_Rest_T", setRestTime);         // PARAMETRO [20-60]
        Particle.function("Set_Publ_T", setPublishTime); // PARAMETRO [1-60]

        // Suscripciones a eventos
        Particle.subscribe("hook-response/aemet", myHandlerAemet, MY_DEVICES);
        Particle.subscribe("hook-response/datosLluvia", myHandlerDatosLluvia,
MY_DEVICES);
        Particle.subscribe("hook-response/siar", myHandlerSiar, MY_DEVICES);

        // El manejador de errores es el mismo para todos los webhooks
        Particle.subscribe("hook-error", myHandlerError, MY_DEVICES);

        // Un respiro antes de empezar
        delay(1000);

        // El sistema no tiene valores
        setupModulo = false;
        setupUmbrales = false;
        setupAlertas = false;
```

```
        setupHW = false;
        mode = "No_inicializado";
        tipoError = "No_inicializado";

        // Iniciamos los timers
        intervaloMuestreo = 0;
        intervaloPublicacion = 0;
        intervaloHorasLuz = 0;
}

void loop()
{

        // Actualizamos el automata
        automataFinito.update();

        if (setupHW)
        {

                // Lee los sensores y publica el estado en el loop.
                if (intervaloMuestreo > readTime * MILISEG_A_SEGUNDOS)
                {

                        // Llevamos mas de "readTime" tiempo sin leer los sensores.
                        readSensors();
                        intervaloMuestreo = 0;
                }

                // Publica el estado en firebase.
                if (intervaloPublicacion > publishTime * MILISEG_A_MINUTOS)
                {

                        // Llevamos mas de "publishTime" tiempo sin leer los sensores.
                        sendFbEstado();
                        intervaloPublicacion = 0;
                }
        }
}

/*****************************************************************************
*
* Vamos con el automata de estados. Definimos funciones de entrada, ejecucion
* y salida para cada estado.
*
*****************************************************************************/

/*****************************************************************************
  - InitCheck: Aqui se inicia el sistema.

        El sistema arranca y comienza en este estado.

        Inicia los sensores y el resto del Hardware.
        .- Si no es capaz, habria que reiniciarlo.

        Carga los datos EEPROM. [Modulo-Umbrales-Alerta].
        .- Si no es capaz, habria que configurarlo desde la App.

        Reset de los flags de estado.

        Si la inicializacion-carga es correcta:
                mode = ON
                Siguiente estado: ReadSens.

        Si la inicializacion-carga no es correcta:
                Siguiente estado: ErrorReport.

        Los actuadores permanecen inactivos.

*****************************************************************************/



void InitCheckEnterFunction()
{

        setEstado(INIT_CHECK);
```

```
        // El sistema esta encendido y no inicializado.
        setupHW = initSystem();

        if (!setupHW)
        {

                // Ha fallado la inicializacion de sensores.
                tipoError = "Inicializando - Imposible iniciar el HW";
                automataFinito.transitionTo(ErrorReport);
        }
}

void InitCheckUpdateFunction()
{

        if (!readFromEeprom())
        {

                resetAlertas();

                // No tenemos umbrales ni datos del modulo. Generamos el error.
                tipoError = "Inicializando - Imposible leer datos EEPROM";
                automataFinito.transitionTo(ErrorReport);
        }
        else
        {

                // Hemos inicializado Hw, Alertas, Modulo y Umbrales.
                resetFlags();
                regadoHoy = false;

                // Todo esta correcto. Encendemos el automata.
                turnOnOff(MODO_ON);
                automataFinito.transitionTo(ReadSens);
        }
}

void InitCheckExitFunction()
{
}

/*********************************************************************************
  - ReadSens: Aqui se mantiene el vector de estado.

        Llegamos a este estado desde InitCheck, SetPlanning, RestEval,
        o SystemOff.

        Ajusta la ventana de riego:
                Primavera-verano = 20 - 23 horas.
                Otoño-invierno   = 13 - 16 horas.

        Si mode = OFF
                Siguiente estado: EstadoOff.

        Si mode = ON

                Actualiza el vector de estado.

                Actualiza el vector de alertas.
                Si ha variado alguna alerta:
                        Publica las alertas en Firebase.

                Si estamos en la ventana de riego y se cumplen las condiciones:
                        Siguiente estado: SetPlanning.

        Los actuadores permanecen inactivos.

*********************************************************************************/



void ReadSensEnterFunction()
{
        setEstado(READ_SENS);
        intervaloChequeo = 0;
        setVentanaRiego();
```

```
}

void ReadSensUpdateFunction()
{
        if (mode == MODO_OFF)
        {

                automataFinito.transitionTo(SystemOff);
        }
        else
        {

                // El automata esta encendido
                if (intervaloChequeo > readTime * MILISEG_A_SEGUNDOS)
                {

                        // Cada readTime [10 seg] comprobamos el estado.
                        checkEstado();

                        // Actualiza las alertas.
                        if (setAlertas())
                        {

                                // Niveles de alerta modificados. Los enviamos a firebase.
                                sendFbAlertas();
                        }

                        intervaloChequeo = 0;

                        // Nuevo dia.
                        if (Time.hour() == 0)
                        {

                                regadoHoy = false;
                                probLluvia = 0;
                        }

                        if ((Time.hour() >= inicioVentana) && (Time.hour() <= finVentana))
                        {

                                // Estamos en la ventana de riego.
                                if ((!regadoHoy) && ((inOut == MODO_IN) || ((inOut ==
MODO OUT) && (probLluvia < 50))))
                                {

                                        // No hemos regado y es de interior.
                                        // ... o no hemos regado, es de exterior y no va a
llover.
                                        automataFinito.transitionTo(SetPlanning);
                                }
                        }
                }
        }
}

void ReadSensExitFunction()
{
}

/*********************************************************************************
  - SetPlanning: Aqui se planifica el riego.

        Llegamos a este estado desde ReadSens.

        Si mode = OFF
                Siguiente estado: EstadoOff.

        Si mode = ON

                Esta dentro de la ventana de riego:

                        Si el modulo es de exterior:
                                Descarga informacion externa: AEMET.

                                Si va a llover:
                                        Siguiente estado: ReadSens.
```

```
                                Si no va a llover:
                                        Descarga informacion externa: SIAR.

                        Si el modulo es de interior:
                                        Descarga informacion externa: SIAR.

                        moistureInicial = moisture.

                        Si el flag de moisture no esta activo:
                                Km -= 0.1 (Valores entre 0.5 y 1.5)
                        Si moistureInicial < marchitez:
                                Km += 0.1 (Valores entre 0.5 y 1.5)
                                marchitando = true.
                                publica hito en firebase.

                        Siguiente estado: doWatering.

                Si hay errores en la descarga de informacion externa:
                        Siguiente estado: ErrorReport.

        Los actuadores permanecen inactivos.

*************************************************************************/

void SetPlanningEnterFunction()
{
        setEstado(SET_PLANNING);
        marchitando = false;
}

void SetPlanningUpdateFunction()
{
        if (mode == MODO_OFF)
        {

                automataFinito.transitionTo(SystemOff);
        }
        else
        {

                // Hemos llegado aqui porque estamos dentro de la ventana de riego.
                if (inOut == MODO_OUT)
                {

                        // Es de exterior. Descargamos informacion AEMET.
                        if (getAemetInfo())
                        {

                                // Hemos descargado la informacion correctamente.
                                if (probLluvia >= 50)
                                {

                                        //Va a llover con una probabilidad > 50%
                                        automataFinito.transitionTo(ReadSens);
                                        return;
                                }
                        }
                        else
                        {
                                // Error al descargar informacion AEMET
                                tipoError = "Planificando - Imposible obtener datos
AEMET";
                                automataFinito.transitionTo(ErrorReport);
                                return;
                        }
                }

                // Es de interior o es de exterior y no va a llover.
                // Descargamos la informacion SIAR.
                if (getSiarInfo())
                {

                        // Hemos descargado la informacion correctamente.
                        moistureInicial = moisture;

                        if (!flags[umbralMoisture].flag)
```

```
                        {

                                // Regamos porque es la hora; pero la tierra sigue humeda.
Hay que regar menos.
                                Km = decrementarCte(Km);
                        }

                        if (moistureInicial < marchitez)
                        {

                                // La humedad ha llegado al punto de marchitez. Hay que
regar mas.
                                Km = incrementarCte(Km);
                                marchitando = true;

                                // Enviamos el hito de alerta.
                                sendFbHito("ALERTA", "Las plantas se estan marchitando");
                        }

                        // Vamos a regar.
                        automataFinito.transitionTo(DoWatering);
                }
                else
                {

                        // Error al descargar informacion SIAR.
                        tipoError = "Planificando - Imposible obtener datos SIAR";
                        automataFinito.transitionTo(ErrorReport);
                }
        }
}

void SetPlanningExitFunction()
{
}

/*********************************************************************************
  - doWatering: Aqui se ejecuta el riego.

        Llegamos a este estado desde setPlanning.

        Si mode = OFF
                Siguiente estado: EstadoOff.

        Si mode = ON
                horaDeRiego = Time().hour();
                Ejecuta el plan de accion en base a:
                        Datos dinamicos: ETo - Km - Fc.
                        Datos del modulo: q - Ne - eficiencia - m2.

                Siguiente estado: RestEval.

                Si hay errores con la bomba:
                        Siguiente estado: ErrorReport.

        Los actuadores permanecen activos.

*********************************************************************************/

void DoWateringEnterFunction()
{
        double dividendo;
        double divisor;

        setEstado(DO_WATERING);
        dividendo = (ETo * Km * Fc * m2);
        divisor = (q * Ne * eficiencia);
        horaDeRiego = Time.hour();
        tiempoDeRiego = dividendo / divisor;
        Particle.publish(APP_NAME, "ETo: " + String(ETo) + " Km: " + String(Km) + " Fc: "
+ String(Fc) + " m2: " + String(m2), PRIVATE);

        if (!startPump())
        {

                tipoError = "Regando - Imposible encender la bomba";
                automataFinito.transitionTo(ErrorReport);
```

```
        }

        intervaloRiego = 0;
}

void DoWateringUpdateFunction()
{
        if (mode == MODO_OFF)
        {

                automataFinito.transitionTo(SystemOff);
        }
        else
        {

                if (intervaloRiego > tiempoDeRiego * MILISEG_A_MINUTOS)
                {

                        // Hemos completado el ciclo de riego.
                        automataFinito.transitionTo(RestEval);
                }
        }
}

void DoWateringExitFunction()
{

        regadoHoy = true;

        // Apaga la bomba de riego.
        if (!stopPump())
        {

                tipoError = "Regando - Imposible apagar la bomba";
                automataFinito.transitionTo(ErrorReport);
        }
}

/*****************************************************************************
  - RestEval:  Aqui dejamos reposar el riego y lo evaluamos.

        Llegamos a este estado desde doWatering.

        Si mode = OFF
                Siguiente estado: EstadoOff.

        Si mode = ON
                Permanece inactivo durante restTime minutos.

                moistureFinal = moisture.

                Si moistureFinal < umbrales[moisture].max:
                        Fc += 0.1 (Valores entre 0.5 y 1.5)

                Si moistureFinal > capacidadCampo:
                        Fc -= 0.1 (Valores entre 0.5 y 1.5)

                Si moistureFinal > saturacion:
                        Fc -= 0.1 (Valores entre 0.5 y 1.5)
                        encharcado = true.
                        publica hito en firebase.

                Enviamos los tados del riego a Firebase.

                Actualiza los valores EEPROM.
                Siguiente estado: ReadSens.

                Si hay errores con la medicion, al publicar o al guardar los datos:
                        Siguiente estado: ErrorReport.

   Los actuadores permanecen inactivos.

*****************************************************************************/

void RestEvalEnterFunction()
{
```

```
        setEstado(REST_EVAL);
        intervaloReposo = 0;
        encharcado = false;
}

void RestEvalUpdateFunction()
{

        if (mode == MODO_OFF)
        {

                automataFinito.transitionTo(SystemOff);
        }
        else
        {

                if (intervaloReposo > restTime * MILISEG_A_MINUTOS)
                {

                        // Ya hemos reposado el tiempo establecido.
                        moistureFinal = moisture;

                        if (moistureFinal < umbrales[umbralMoisture].max)
                        {

                                Fc = incrementarCte(Fc);
                        }

                        if (moistureFinal > capacidadCampo)
                        {

                                Fc = decrementarCte(Fc);
                        }

                        if (moistureFinal > saturacion)
                        {

                                Fc = decrementarCte(Fc);
                                encharcado = true;

                                // Enviamos el hito de alerta.
                                sendFbHito("ALERTA", "El modulo esta encharcado");
                        }

                        // Enviamos los datos del riego.
                        sendFbRiego();

                        if (!saveSettingsInEeprom())
                        {

                                // No se ha podido guardar la configuracion en EEPROM
                                tipoError = "Reposo - Imposible guardar datos en EEPROM";
                                automataFinito.transitionTo(ErrorReport);
                        }
                        else
                        {

                                // Hemos guardado correctamente la configuracion en EEPROM
                                automataFinito.transitionTo(ReadSens);
                        }
                }
        }
}

void RestEvalExitFunction()
{
}



/*****************************************************************************
  - ErrorReport: Aqui se tratan los errores.

        Llegamos a este estado desde cualquier otro estado.
```

```
        Publica en firebase la informacion del error como hito.
        Siguiente estado: SystemOff.

        Los actuadores permanecen inactivos.

*****************************************************************************/

void ErrorReportEnterFunction()
{

        setEstado(ERROR_REPORT);

        // Enviamos el error como hito
        sendFbHito("ERROR", tipoError);
}

void ErrorReportUpdateFunction()
{

        // Apagamos el automata.
        automataFinito.transitionTo(SystemOff);
}

void ErrorReportExitFunction()
{
}

/*****************************************************************************
  - SystemOff: Aqui el sistema permanece inactivo.

        Llegamos a este estado si el usuario desactiva el sistema o se ha producido
        un error.

        Apaga el automata.

        Si mode = ON
                Siguiente estado: ReadSens.

        Los actuadores permanecen inactivos.

*****************************************************************************/

void SystemOffEnterFunction()
{

        setEstado(SYSTEM_OFF);
        turnOnOff(MODO_OFF);
}

void SystemOffUpdateFunction()
{
        if (mode == MODO_ON)
        {

                // Si se ha encendido es porque esta OK: HW, Alertas, Umbrales y Modulo.
                automataFinito.transitionTo(ReadSens);
        }
}

void SystemOffExitFunction()
{
}

/*****************************************************************************

  Fin de las funciones relacionadas con los estados del automata.

*****************************************************************************/



/*****************************************************************************

  Funciones de control.
  Nos permiten modificar el comportamiento del sistema.
```

```
  Publicadas en particle.io para poder llamarlas facilmente desde cualquier
      aplicacion.

*******************************************************************************/

/*******************************************************************************
 * Function Name  : turnOnOff
 * Description    : Activa/desactiva el sistema.
 * Parameters     : String parameter:
 *                                          .- ("On")  --> Sistema activo.
 *                                          .- ("Off") --> Sistema inactivo.
 *                    .- Otros valores son ignorados.
 * Return         : 0 Ok, -1 Fallo.
*******************************************************************************/
int turnOnOff(String parameter)
{
        if ((parameter == MODO_ON) || (parameter == "on") || (parameter == "ON"))
        {
                if (setupModulo && setupUmbrales && setupAlertas && setupHW)
                {

                        // El sistema esta inicializado y configurado. El HW funciona.
                        mode = MODO_ON;
                        saveSettingsInEeprom();
                        sendFbHito("SISTEMA", "Sistema " + idPhoton + " activado.");
                        Particle.publish(APP_NAME, "Sistema " + idPhoton + " activado: " +
parameter, PRIVATE);
                        delay(1000);
                        return 0;
                }
                else
                {
                        sendFbHito("ERROR", "Sistema " + idPhoton + " no configurado.");
                        Particle.publish(APP_NAME, "Sistema " + idPhoton + " no
configurado.", PRIVATE);
                        delay(1000);
                        return -1;
                }
        }

        if ((parameter == MODO_OFF) || (parameter == "off") || (parameter == "OFF"))
        {
                mode = MODO OFF;
                saveSettingsInEeprom();
                sendFbHito("SISTEMA", "Sistema " + idPhoton + " desactivado.");
                Particle.publish(APP_NAME, "Sistema " + idPhoton + " desactivado: " +
parameter, PRIVATE);
                delay(1000);
                return 0;
        }

        sendFbHito("ERROR", "Fallo al activar/desactivar el sistema: " + idPhoton);
        Particle.publish(APP_NAME, "Fallo al activar/desactivar el sistema: " + idPhoton
+ " " + parameter, PRIVATE);
        delay(1000);
        return -1;
}

/*******************************************************************************
 * Function Name  : setInOut
 * Description    : Configura el sistema. Interior/Exterior.
 * Parameters     : String parameter:
 *                                          .- ("In")  --> Modulo ubicado en interior.
 *                                          .- ("Out") --> Modulo ubicado en exterior.
 *                                  .- Otros valores son ignorados.
 * Return         : 0 Ok, -1 Fallo.
 *******************************************************************************/
int setInOut(String parameter)
{
        if ((parameter == MODO_IN) || (parameter == "in") || (parameter == "IN"))
        {
                inOut = MODO_IN;
                saveSettingsInEeprom();
                sendFbHito("SISTEMA", "Sistema " + idPhoton + " ubicado en interior.");
                Particle.publish(APP_NAME, "Sistema: " + idPhoton + " - Modo de
funcionamiento: " + parameter, PRIVATE);
                delay(1000);
```

```
                return 0;
        }

        if ((parameter == MODO_OUT) || (parameter == "out") || (parameter == "OUT"))
        {
                inOut = MODO_OUT;
                saveSettingsInEeprom();
                sendFbHito("SISTEMA", "Sistema " + idPhoton + " ubicado en exterior.");
                Particle.publish(APP_NAME, "Sistema: " + idPhoton + " - Modo de
funcionamiento: " + parameter, PRIVATE);
                delay(1000);
                return 0;
        }

        sendFbHito("ERROR", "Sistema: " + idPhoton + " - Fallo al fijar modo de
funcionamiento.");
        Particle.publish(APP_NAME, "Sistema: " + idPhoton + " - Fallo al fijar modo de
funcionamiento: " + parameter, PRIVATE);
        delay(1000);
        return -1;
}

/*************************************************************************
 * Function Name  : setModulo
 * Description    : Actualiza los datos del modulo
 *                  y resetea la configuracion de riego.
 * Parameters     : String parameter: Configuracion del modulo en JSON.
 * Return         : 0 Ok, -1 Fallo.
 *************************************************************************/
int setModulo(String parameter)
{
        String datosModulo;
        int elSueloModulo;
        int qModulo;
        int NeModulo;
        float m2Modulo;

        // Cargamos los datos de la llamada
        datosModulo = strdup(parameter);
        parser.clear();
        parser.addString(datosModulo);

        // Los datos contienen un JSON procesable
        if (parser.parse())
        {

                elSueloModulo = parser.getReference().key("tipoSuelo").valueInt();
                qModulo = parser.getReference().key("caudal").valueInt(); // litros/hora
                NeModulo = parser.getReference().key("emisores").valueInt();
                m2Modulo = parser.getReference().key("superficie").valueFloat();

                if ((qModulo > 0) && (NeModulo > 0) && (m2Modulo > 0))
                {

                        elSuelo = intToTipoSuelo(elSueloModulo);
                        q = qModulo;
                        Ne = NeModulo;
                        m2 = m2Modulo;

                        // Hay que calcular todos los valores de riego.
                        setCamposCalculados();

                        // Reseteamos la configuracion de riego.
                        resetRiego("CLOUD");

                        // Actualizamos los valores de control.
                        setupModulo = true;

                        // Guardamos la configuracion.
                        saveSettingsInEeprom();

                        // El parametro es correcto y hemos descargado los datos del
modulo.
                        Particle.publish(APP_NAME, "Suelo: " +
String(tipoSueloToInt(elSuelo)) + " q: " + q + " Ne: " + Ne + " m2: " + m2, PRIVATE);
                        delay(1000);
```

```
                        return 0;
                }
        }

        // El parametro no es correcto o no podemos descargar los datos del modulo.
        sendFbHito("ERROR", "Fallo al asignar el sistema: " + idPhoton + " al modulo: " +
datosModulo);
        Particle.publish(APP_NAME, "Fallo al asignar el sistema: " + idPhoton + " al
modulo: " + datosModulo, PRIVATE);
        delay(1000);

        return -1;
}

/*******************************************************************************
 * Function Name  : setUmbrales
 * Description    : Carga de umbrales.
 *                                  hace un reset de la configuracion de riego.
 * Parameters     : String parameter: Nuevos umbrales en JSON.
 * Return         : 0 Ok, -1 Fallo.
 *******************************************************************************/
int setUmbrales(String parameter)
{
        String datosUmbrales;

        // Cargamos los datos de la llamada
        datosUmbrales = strdup(parameter);
        parser.clear();
        parser.addString(datosUmbrales);

        // Los datos contienen un JSON procesable
        if (parser.parse())
        {
                for (int i = 0; i < NUM_SENS; i++)
                {
                        switch (i)
                        {
                        case 0: // Temperatura

                                umbrales[i].max =
parser.getReference().key("temperatura").key("max").valueFloat();
                                umbrales[i].min =
parser.getReference().key("temperatura").key("min").valueFloat();
                                break;

                        case 1: // Humedad

                                umbrales[i].max =
parser.getReference().key("humedad").key("max").valueFloat();
                                umbrales[i].min =
parser.getReference().key("humedad").key("min").valueFloat();
                                break;

                        case 2: // Horas de luz

                                umbrales[i].max =
parser.getReference().key("horasDeLuz").key("max").valueFloat();
                                umbrales[i].min =
parser.getReference().key("horasDeLuz").key("min").valueFloat();
                                break;

                        case 3: // moisture

                                umbrales[i].max =
parser.getReference().key("moisture").key("max").valueFloat();
                                umbrales[i].min =
parser.getReference().key("moisture").key("min").valueFloat();
                                break;
                        }
                }

                // Reset de la configuracion de riego.
                resetRiego("CLOUD");

                // Actualizamos los valores de control.
                setupUmbrales = true;
```

```
                // Guardamos la configuracion.
                saveSettingsInEeprom();

                // El parametro es correcto y hemos descargado los umbrales.
                sendFbHito("SISTEMA", "Umbrales asignados al sistema: " + idPhoton);
                Particle.publish(APP_NAME, "Umbrales asignados al sistema: " + idPhoton,
PRIVATE);
                delay(1000);
                return 0;
        }

        // El parametro no es correcto o no podemos descargar los umbrales.
        sendFbHito("ERROR", "Umbrales NO asignados al sistema: " + idPhoton);
        Particle.publish(APP_NAME, "Umbrales NO asignados al sistema: " + idPhoton,
PRIVATE);
        delay(1000);
        return -1;
}

/*******************************************************************************
 * Function Name  : resetRiego
 * Description    : Reinicia la configuracion de riego.
 * Parameters     : String parameter: CLOUD.
 * Return         : 0 Ok, -1 Fallo.
 *******************************************************************************/
int resetRiego(String parameter)
{

        // El parametro debe coincidir con nuestro modulo.
        if (parameter.equals("CLOUD"))
        {
                // El parametro es correcto.
                Km = 1.0;
                Fc = 1.0;
                riegoFijo = 0;
                saveSettingsInEeprom();
                sendFbHito("SISTEMA", "Riego restablecido en el sistema: " + idPhoton);
                Particle.publish(APP_NAME, "Riego restablecido en el sistema: " +
idPhoton + " Ok.", PRIVATE);
                delay(1000);
                return 0;
        }

        sendFbHito("ERROR", "El riego no pudo restablecerse en el sistema: " + idPhoton);
        Particle.publish(APP_NAME, "El riego no pudo restablecerse en el sistema: " +
idPhoton + " ERROR.", PRIVATE);
        delay(1000);
        return -1;
}

/*******************************************************************************
 * Function Name  : setRiego
 * Description    : fija la configuracion de riego.
 * Parameters     : String parameter: FIJO o DINAMICO.
 * Return         : 0 Ok, -1 Fallo.
 *******************************************************************************/
int setRiego(String parameter)
{

        // El parametro debe ser correcto.
        if (parameter.equals("FIJO"))
        {
                // El parametro es correcto.
                riegoFijo = 1;
                saveSettingsInEeprom();
                sendFbHito("SISTEMA", "Riego del sistema: " + idPhoton + " FIJO.");
                Particle.publish(APP_NAME, "Riego del sistema: " + idPhoton + " FIJO.",
PRIVATE);
                delay(1000);
                return 0;
        }

        if (parameter.equals("DINAMICO"))
        {
                // El parametro es correcto.
                riegoFijo = 0;
```

```
                saveSettingsInEeprom();
                sendFbHito("SISTEMA", "Riego del sistema: " + idPhoton + " DINAMICO.");
                Particle.publish(APP_NAME, "Riego del sistema: " + idPhoton + "
DINAMICO.", PRIVATE);
                delay(1000);
                return 0;
        }

        sendFbHito("ERROR", "Error al ajustar el riego del sistema: " + idPhoton);
        Particle.publish(APP_NAME, "Error al ajustar el riego del sistema: " + idPhoton +
" Param: " + parameter, PRIVATE);
        delay(1000);
        return -1;
}

/*******************************************************************************
 * Function Name  : resetEstado
 * Description    : Reinicia la configuracion de alertas y flags.
 * Parameters     : String parameter: CLOUD.
 * Return         : 0 Ok, -1 Fallo.
 *******************************************************************************/
int resetEstado(String parameter)
{

        // El parametro debe coincidir con nuestro modulo.
        if (parameter.equals("CLOUD"))
        {
                // El parametro es correcto.
                resetAlertas();
                resetFlags();
                Particle.publish(APP_NAME, "Reset de estado del sistema: " + idPhoton + "
OK", PRIVATE);
                delay(1000);
                return 0;
        }

        Particle.publish(APP_NAME, "Reset de estado del sistema: " + idPhoton + " ERROR",
PRIVATE);
        delay(1000);
        return -1;
}

/*******************************************************************************
 * Function Name  : setRestTime
 * Description    : fija el tiempo de reposo tras el riego
 * Parameters     : String parameter: tiempo en minutos
                    Admite valores entre 20 y 60 minutos
 * Return         : 0 Ok, -1 Error
 *******************************************************************************/
int setRestTime(String parameter)
{

        int variable = atoi(parameter);

        if ((variable >= 20) && (variable <= 60))
        {
                restTime = variable;
                sendFbHito("SISTEMA", "Sistema: " + idPhoton + " - Nuevo tiempo de reposo
fijado: " + parameter);
                Particle.publish(APP_NAME, "Sistema: " + idPhoton + " - Nuevo tiempo de
reposo fijado: " + parameter, PRIVATE);
                delay(1000);
                return 0;
        }

        sendFbHito("ERROR", "Sistema: " + idPhoton + " - Error al fijar el tiempo de
reposo: " + parameter + " (20<=tiempo<=60)");
        Particle.publish(APP_NAME, "Sistema: " + idPhoton + " - Error al fijar el tiempo
de reposo: " + parameter + " (20<=tiempo<=60)", PRIVATE);
        delay(1000);
        return -1;
}

/*******************************************************************************
 * Function Name  : setPublishTime
 * Description    : fija el ciclo de publicacion en Firebase
 * Parameters     : String parameter: tiempo en minutos
```

```
                    Admite valores entre 1 y 60 minutos
 * Return          : 0 Ok, -1 Error
 ******************************************************************************/
int setPublishTime(String parameter)
{

        int variable = atoi(parameter);

        if ((variable >= 1) && (variable <= 60))
        {
                publishTime = variable;
                sendFbHito("SISTEMA", "Sistema: " + idPhoton + " - Nuevo tiempo de
publicacion fijado: " + parameter);
                Particle.publish(APP_NAME, "Sistema: " + idPhoton + " - Nuevo tiempo de
publicacion fijado: " + parameter, PRIVATE);
                delay(1000);
                return 0;
        }

        sendFbHito("ERROR", "Sistema: " + idPhoton + " - Error al fijar el tiempo de
publicacion: " + parameter + " (1<=tiempo<=60)");
        Particle.publish(APP_NAME, "Sistema: " + idPhoton + " - Error al fijar el tiempo
de publicacion: " + parameter + " (1<=tiempo<=60)", PRIVATE);
        delay(1000);
        return -1;
}

/*******************************************************************************

  Fin de las funciones de control.

*******************************************************************************/

/*******************************************************************************

  Funciones de acceso a EEPROM.
  Nos permiten almacenar y recuperar el estado del sistema.
  Utilizamos algunas funciones de conversion de valores para las enumeraciones.

*******************************************************************************/

/*******************************************************************************
* Function Name  : intToAlerta
* Description    : convierte el valor entero (EEPROM)
*                  en su enumeracion correspondiente (RAM)
* Return         : tipoAlerta
*******************************************************************************/
tipoAlerta intToAlerta(int alerta)
{
        tipoAlerta laAlerta;

        switch (alerta)
        {
        case 0:

                laAlerta = SinAlerta;
                break;

        case 1:

                laAlerta = RiesgoBajo;
                break;

        case 2:

                laAlerta = RiesgoMedio;
                break;

        case 3:

                laAlerta = RiesgoAlto;
                break;

        case 4:

                laAlerta = RiesgoCritico;
```

```
                break;

        default:

                laAlerta = SinAlerta;
                break;
        }

        return laAlerta;
}

/*****************************************************************************
* Function Name  : alertaToInt
* Description    : convierte el valor enum (RAM)
*                  en su entero correspondiente (EEPROM)
* Return         : int
*****************************************************************************/
int alertaToInt(tipoAlerta alerta)
{
        int laAlerta;

        switch (alerta)
        {
        case SinAlerta:

                laAlerta = 0;
                break;

        case RiesgoBajo:

                laAlerta = 1;
                break;

        case RiesgoMedio:

                laAlerta = 2;
                break;

        case RiesgoAlto:

                laAlerta = 3;
                break;

        case RiesgoCritico:

                laAlerta = 4;
                break;
        }

        return laAlerta;
}

/*****************************************************************************
* Function Name  : intToUmbral
* Description    : convierte el valor entero (EEPROM)
*                  en su enumeracion correspondiente (RAM)
* Return         : tipoUmbral
*****************************************************************************/
tipoUmbral intToUmbral(int umbral)
{
        tipoUmbral elUmbral;

        switch (umbral)
        {
        case 0:

                elUmbral = umbralTemp;
                break;

        case 1:

                elUmbral = umbralHumedad;
                break;

        case 2:

                elUmbral = umbralHorasLuz;
```

```
                break;

        case 3:

                elUmbral = umbralMoisture;
                break;
        }

        return elUmbral;
}

/*****************************************************************************
* Function Name  : umbralToInt
* Description    : convierte el valor enum (RAM)
*                  en su entero correspondiente (EEPROM)
* Return         : int
*****************************************************************************/
int umbralToInt(tipoUmbral umbral)
{
        int elUmbral;

        switch (umbral)
        {
        case umbralTemp:

                elUmbral = 0;
                break;

        case umbralHumedad:

                elUmbral = 1;
                break;

        case umbralHorasLuz:

                elUmbral = 2;
                break;

        case umbralMoisture:

                elUmbral = 3;
                break;
        }

        return elUmbral;
}

/*****************************************************************************
* Function Name  : intToSensor
* Description    : convierte el valor entero (EEPROM)
*                  en su enumeracion correspondiente (RAM)
* Return         : tipoSensor
*****************************************************************************/
tipoSensor intToSensor(int sensor)
{
        tipoSensor elSensor;

        switch (sensor)
        {
        case 0:

                elSensor = tempAireS;
                break;

        case 1:

                elSensor = humAireS;
                break;

        case 2:

                elSensor = tempTierraS;
                break;

        case 3:
```

```
                elSensor = moistureS;
                break;

        case 4:

                elSensor = luzVisibleAireS;
                break;

        case 5:

                elSensor = luzIRAire;
                break;

        case 6:

                elSensor = UVIndexAire;
                break;
        }

        return elSensor;
}

/*****************************************************************************
* Function Name  : sensorToInt
* Description    : convierte el valor enum (RAM)
*                  en su entero correspondiente (EEPROM)
* Return         : int
*****************************************************************************/
int sensorToInt(tipoSensor sensor)
{
        int elSensor;

        switch (sensor)
        {
        case tempAireS:

                elSensor = 0;
                break;

        case humAireS:

                elSensor = 1;
                break;

        case tempTierraS:

                elSensor = 2;
                break;

        case moistureS:

                elSensor = 3;
                break;

        case luzVisibleAireS:

                elSensor = 4;
                break;

        case luzIRAire:

                elSensor = 5;
                break;

        case UVIndexAire:

                elSensor = 6;
                break;
        }

        return elSensor;
}

/*****************************************************************************
* Function Name  : intToTipoSuelo
* Description    : convierte el valor entero (EEPROM)
*                  en su enumeracion correspondiente (RAM)
```

```
* Return         : tipoSuelo
*******************************************************************************/
tipoSuelo intToTipoSuelo(int sueloInt)
{
        tipoSuelo suelo;

        switch (sueloInt)
        {
        case 0:

                suelo = gravoso;
                break;

        case 1:

                suelo = arenoso;
                break;

        case 2:

                suelo = franco;
                break;

        case 3:

                suelo = arcilloso;
                break;
        }

        return suelo;
}

/*******************************************************************************
* Function Name  : tipoSueloToInt
* Description    : convierte el valor enum (RAM)
*                  en su entero correspondiente (EEPROM)
* Return         : int
*******************************************************************************/
int tipoSueloToInt(tipoSuelo suelo)
{
        int sueloInt;

        switch (suelo)
        {
        case gravoso:

                sueloInt = 0;
                break;

        case arenoso:

                sueloInt = 1;
                break;

        case franco:

                sueloInt = 2;
                break;

        case arcilloso:

                sueloInt = 3;
                break;
        }

        return sueloInt;
}



/*******************************************************************************
 * Function Name  : intToInOut
 * Description    : transforma el valor entero (EEPROM)
 *                  en su enumeracion correspondiente (RAM)
 * Return         : String
```

```
 ********************************************************************************/
String intToInOut(int inOut)
{
        if (inOut == 1)
        {
                return MODO_IN;
        }

        //En otro caso.
        return MODO_OUT;
}

/********************************************************************************
 * Function Name  : inOutToInt
 * Description    : transforma la cadena (RAM)
 *                  en un valor numerico para almacenarlo (EEPROM).
 * Return         : int
 ********************************************************************************/
int inOutToInt(String inOut)
{
        if (inOut == MODO_IN)
        {
                return 1;
        }

        //En otro caso.
        return 0;
}

/********************************************************************************
 * Function Name  : intToMode
 * Description    : transforma el valor entero (EEPROM)
 *                  en su enumeracion correspondiente (RAM)
 * Return         : String
 ********************************************************************************/
String intToMode(int mode)
{
        if (mode == 1)
        {
                return MODO_ON;
        }

        //En otro caso.
        return MODO_OFF;
}

/********************************************************************************
 * Function Name  : modeToInt
 * Description    : transforma la cadena (RAM)
 *                  en un valor numerico para almacenarlo (EEPROM).
 * Return         : int
 ********************************************************************************/
int modeToInt(String mode)
{
        if (mode == MODO_ON)
        {
                return 1;
        }

        //En otro caso.
        return 0;
}
```

```
/********************************************************************************
 * Function Name  : saveSettingsInEeprom
 * Description    : Guarda en EEPROM la configuracion actual del sistema.
 *                    El numero de veces que puede escribirse la EEPROM es finito.
 * Return         : True si todo es correcto. False en caso de excepcion.
```

```
 ***************************************************************************/
bool saveSettingsInEeprom()
{

        int addr;
        bool resultado;

        addr = EEPROM_ADDRESS_1;
        resultado = false;

        // Este primer grupo de datos ocupa 45 bytes.
        eepromMemory.version = EEPROM_VERSION;
        eepromMemory.inOut = inOutToInt(inOut);
        eepromMemory.mode = modeToInt(mode);
        eepromMemory.riegoFijoEEprom = riegoFijo;
        eepromMemory.KmEEprom = Km;
        eepromMemory.FcEEprom = Fc;
        eepromMemory.elSueloEeprom = tipoSueloToInt(elSuelo);
        eepromMemory.qEeprom = q;
        eepromMemory.NeEeprom = Ne;
        eepromMemory.m2Eeprom = m2;

        // Guardamos la estructura inicial.
        EEPROM.put(addr, eepromMemory);
        delay(100);

        // Guardamos los umbrales. 32 bytes desde la posicion 46.
        // umbral umbrales[NUM_SENS];  // 8 * 4  --> 32 bytes.
        addr = EEPROM_ADDRESS_2;

        for (int i = 0; i < NUM_SENS; i++)
        {

                addr += (i * 8);
                EEPROM.put(addr, umbrales[i]);
                delay(100);
        }

        // Guardamos las alertas. 16 bytes desde la posicion 79.
        // alerta alertas[NUM_SENS];   // 4 * 4 --> 16 bytes.
        addr = EEPROM_ADDRESS_3;

        for (int i = 0; i < NUM_SENS; i++)
        {

                addr += (i * 4);

                // Solo guardamos el nivel de alerta.
                EEPROM.put(addr, alertaToInt(alertas[i].nivel));
                delay(100);
        }

        // 89 bytes escritos. Se almacenan unicamente cuando cambia la configuracion.
        // Podemos escribir 200 millones de bytes antes de que la flash se deteriore.
        // 200.000.000 / 100 --> podemos almacenar unos 2 millones de cambios de
configuracion.

        resultado = true;
        Particle.publish(APP_NAME, "Datos almacenados en EEPROM. Sistema: " + idPhoton +
"m2: " + eepromMemory.m2Eeprom, PRIVATE);
        delay(1000);

        return resultado;
}

/*****************************************************************************
 * Function Name  : readFromEeprom
 * Description    : carga los datos desde la memoria EEPROM.
 * Return         : True si todo es correcto. False en caso de excepcion.
 *****************************************************************************/
bool readFromEeprom()
{

        int addr;
        int alertaProv;
        EepromMemoryStructure misDatos;
```

```
        bool resultado;

        resultado = false;
        addr = EEPROM_ADDRESS_1;

        EEPROM.get(addr, misDatos);
        delay(100);

        //comprobamos que la memoria ha sido escrita con anterioridad.
        if (misDatos.version == EEPROM_VERSION)
        {
                inOut = intToInOut(misDatos.inOut);
                mode = intToMode(misDatos.mode);
                riegoFijo = misDatos.riegoFijoEEprom;
                Km = misDatos.KmEEprom;
                Fc = misDatos.FcEEprom;
                elSuelo = intToTipoSuelo(misDatos.elSueloEeprom);
                q = misDatos.qEeprom;
                Ne = misDatos.NeEeprom;
                m2 = misDatos.m2Eeprom;

                setCamposCalculados();
                setupModulo = true;

                // Cargamos los umbrales. 32 bytes desde la posicion 46.
                // umbral umbrales[NUM_SENS];  // 8 * 4  --> 32 bytes.
                addr = EEPROM_ADDRESS_2;

                for (int i = 0; i < NUM_SENS; i++)
                {

                        addr += (i * 8);

                        // cargamos la estructura.
                        EEPROM.get(addr, umbrales[i]);
                        delay(100);
                }

                setupUmbrales = true;

                // Cargamos las alertas. 16 bytes desde la posicion 79.
                // alerta alertas[NUM_SENS];   // 4 * 4 --> 16 bytes.
                addr = EEPROM_ADDRESS_3;

                for (int i = 0; i < NUM_SENS; i++)
                {

                        addr += (i * 4);

                        // cargamos el nivel.
                        EEPROM.get(addr, alertaProv);
                        delay(100);
                        alertas[i].nivel = intToAlerta(alertaProv);
                }

                setupAlertas = true;
                sendFbHito("SISTEMA", "Datos EEPROM cargados en sistema: " + idPhoton +
"Modulo: " + MODULO);
                Particle.publish(APP_NAME, "Datos EEPROM cargados en sistema: " +
idPhoton + "Modulo: " + MODULO, PRIVATE);
                delay(1000);
        }

        resultado = (setupModulo && setupUmbrales && setupAlertas);
        return resultado;
}

/*****************************************************************************

  Fin de las funciones EEPROM.

*****************************************************************************/

/*****************************************************************************

  Funciones acceso a los sensores y actuadores.
  Permiten la lectura de variables de entorno y el control e la bomba de riego.
```

```
*****************************************************************************/

/*****************************************************************************
 * Function Name  : readSensors
 * Description    : Lee el sensor cada SENSORS_SAMPLE_INTERVAL
 * Return         : True si todo es correcto. False en caso de excepcion.
 *****************************************************************************/
bool readSensors()
{

        bool resultado;

        // Valor del thermistor.
        const int B = 3975;

        // Variables de lectura e interpretacion
        double resistance;
        double temperature;
        int analogValue;

        //Tomamos NUMBER_OF_SAMPLES muestras y hacemos la media
        double moistureAv;
        double moistureSamples[NUMBER_OF_SAMPLES];
        double tempTierraAv;
        double tempTierraSamples[NUMBER_OF_SAMPLES];
        double tempAireAv;
        double tempAireSamples[NUMBER_OF_SAMPLES];
        double humedadAireAv;
        double humedadAireSamples[NUMBER_OF_SAMPLES];
        double luzVisibleAv;
        double luzVisibleSamples[NUMBER_OF_SAMPLES];
        double luzIRAv;
        double luzIRSamples[NUMBER_OF_SAMPLES];
        double UVIndexAv;
        double UVIndexSamples[NUMBER_OF_SAMPLES];

        resultado = false;

        // Tomamos las muestras
        for (int i = 0; i < NUMBER_OF_SAMPLES; i++)
        {

                // moisture
                moistureSamples[i] = analogRead(moistureSensor);
                delay(10);

                // temperatura del suelo
                analogValue = analogRead(tempSensor);

                // Calculamos la resistencia del sensor
                resistance = (double)(4095 - analogValue) * 100000 / analogValue;

                // Calculamos la temperatura [°C = (°F - 32) * 5/9] - [°K = °C - 273,16]
                temperature = 1 / (log(resistance / 100000) / B + 1 / 298.15) - 273.16;

                tempTierraSamples[i] = temperature;
                delay(10);

                // Temperatura del aire
                tempAireSamples[i] = dht.getTempCelcius();
                delay(10);

                // Humedad del aire
                humedadAireSamples[i] = dht.getHumidity();
                delay(10);

                // Luz visible
                luzVisibleSamples[i] = (double)uv.readVisible();
                delay(10);

                // Luz IR
                luzIRSamples[i] = (double)uv.readIR();
                delay(10);

                // Indice UV
```

```
                UVIndexSamples[i] = ((double)uv.readUV() / 100.0);
                delay(10);
        }

        //Particle.publish(APP_NAME, "readSensors - vectores rellenos. Sistema: " +
idPhoton, PRIVATE);
        //delay(1000);

        // Hacemos la media
        moistureAv = 0.0;
        tempTierraAv = 0.0;
        tempAireAv = 0.0;
        humedadAireAv = 0.0;
        luzVisibleAv = 0.0;
        luzIRAv = 0.0;
        UVIndexAv = 0.0;

        for (int i = 0; i < NUMBER_OF_SAMPLES; i++)
        {

                moistureAv += moistureSamples[i];
                tempTierraAv += tempTierraSamples[i];
                tempAireAv += tempAireSamples[i];
                humedadAireAv += humedadAireSamples[i];
                luzVisibleAv += luzVisibleSamples[i];
                luzIRAv += luzIRSamples[i];
                UVIndexAv += UVIndexSamples[i];
        }

        moistureAv /= NUMBER_OF_SAMPLES;
        tempTierraAv /= NUMBER_OF_SAMPLES;
        tempAireAv /= NUMBER_OF_SAMPLES;
        humedadAireAv /= NUMBER_OF_SAMPLES;
        luzVisibleAv /= NUMBER_OF_SAMPLES;
        luzIRAv /= NUMBER_OF_SAMPLES;
        UVIndexAv /= NUMBER_OF_SAMPLES;

        //Particle.publish(APP_NAME, "readSensors - medias calculadas. Sistema: " +
idPhoton, PRIVATE);
        //delay(1000);

        // Tenemos nuestros valores
        moisture = moistureAv;
        tempTierra = tempTierraAv;
        tempAire = tempAireAv;
        humedadAire = humedadAireAv;
        luzVisible = luzVisibleAv;
        luzIR = luzIRAv;
        UVIndex = UVIndexAv;
        horasDeLuz = intervaloHorasLuz / MILISEG_A_HORAS;

        if ((luzVisible < LUZ_DE_DIA) || (UVIndex < UV_DE_DIA))
        {

                if (horasDeLuz > 0)
                {
                        sendFbHito("ILUMINACION", String(horasDeLuz));
                }

                // Es de noche --> Se ha completado un ciclo de luz.
                intervaloHorasLuz = 0;
        }

        // Datos de la bateria
        cellVoltage = batteryMonitor.getVCell();
        stateOfCharge = batteryMonitor.getSoC();

        resultado = true;

        return resultado;
}

/*******************************************************************************
 * Function Name  : startPump
 * Description    : Enciende la bomba de riego.
 * Parameters     : -.
 * Return         : True si todo es correcto. False en caso de excepcion.
```

```
 ********************************************************************************/
bool startPump()
{

        bool resultado;

        resultado = false;
        digitalWrite(pinLed, HIGH);
        digitalWrite(releBomba, HIGH);

        // Hemos encendido la bomba sin problemas.
        resultado = true;
        Particle.publish(APP_NAME, "Bomba encendida. Sistema: " + idPhoton, PRIVATE);
        delay(1000);

        return resultado;
}

/********************************************************************************
 * Function Name  : stopPump
 * Description    : Detiene la bomba de riego.
 * Parameters     : -.
 * Return         : True si todo es correcto. False en caso de excepcion.
 ********************************************************************************/
bool stopPump()
{

        bool resultado;

        resultado = false;
        digitalWrite(pinLed, LOW);
        digitalWrite(releBomba, LOW);

        // Hemos detenido la bomba sin problemas.
        resultado = true;
        Particle.publish(APP_NAME, "Bomba apagada. Sistema: " + idPhoton, PRIVATE);
        delay(1000);

        return resultado;
}

/********************************************************************************

  Fin de las funciones de acceso a los sensores y actuadores.

********************************************************************************/

/********************************************************************************

  Funciones acceso a los servicios cloud.
  Nos permiten:
  .- Almacenar valores en Firebase.
  .- Consultar informacion AEMET.
  .- Consultar informacion SIAR.

********************************************************************************/

/********************************************************************************
 * Function Name  : sendFbEstado
 * Description    : Envia el estado de los sensores a Firebase para almacenarlo.
 * Parameters     : -.
 * Return         : -.
 ********************************************************************************/
void sendFbEstado()
{

        char datosEstado[600];

        snprintf(datosEstado, sizeof(datosEstado),
"{\"tempTierra\":%.1f,\"tempAire\":%.1f,\"humedadAire\":%.1f,\"luzVisible\":%.1f,\"luzIR
\":%.1f,\"UVIndex\":%.1f,\"moisture\":%.1f,\"horasDeLuz\":%d}",
                        tempTierra, tempAire, humedadAire, luzVisible, luzIR, UVIndex,
moisture, horasDeLuz);
        Particle.publish("addEstado", String(datosEstado), PRIVATE);
        delay(1000);
}
```

```
/*******************************************************************************
 * Function Name  : sendFbAlertas
 * Description    : Envia el vector de alertas a Firebase.
 * Parameters     : -.
 * Return         : -.
 *******************************************************************************/
void sendFbAlertas()
{

        char datosAlerta[600];

        // Variables de alerta.
        int alertaHorasDeLuz;
        int alertaHumedad;
        int alertaMoisture;
        int alertaTemperatura;

        // Asignamos las variables.
        alertaTemperatura = alertaToInt(alertas[0].nivel);
        alertaHumedad = alertaToInt(alertas[1].nivel);
        alertaHorasDeLuz = alertaToInt(alertas[2].nivel);
        alertaMoisture = alertaToInt(alertas[3].nivel);

        snprintf(datosAlerta, sizeof(datosAlerta),
"{\"horasDeLuz\":%d,\"humedad\":%d,\"moisture\":%d,\"temperatura\":%d}",
                        alertaHorasDeLuz, alertaHumedad, alertaMoisture,
alertaTemperatura);
        Particle.publish("setAlerta", String(datosAlerta), PRIVATE);
        delay(1000);
}

/*******************************************************************************
 * Function Name  : sendFbHito
 * Description    : Envia los hitos de sistema a Firebase para almacenarlos.
 * Parameters     : String tipoHito, String descHito.
 * Return         : -.
 *******************************************************************************/
void sendFbHito(String tipoHito, String descHito)
{

        // Variables de hitos. Falta el momento. Lo pone la funcion suscriptora.
        char datosHito[600];

        snprintf(datosHito, sizeof(datosHito),
"{\"descripcion\":\"%s\",\"tipo\":\"%s\"}",
                        descHito.c_str(), tipoHito.c_str());
        Particle.publish("addHito", String(datosHito), PRIVATE);
        delay(1000);
}

/*******************************************************************************
 * Function Name  : sendFbRiego
 * Description    : Envia los datos de riego a Firebase para almacenarlos.
 * Parameters     : -.
 * Return         : -.
 *******************************************************************************/
void sendFbRiego()
{

        char datosRiego[600];
        String estaEncharcado = (encharcado ? "true" : "false");

        snprintf(datosRiego, sizeof(datosRiego),
"{\"encharcado\":%s,\"horaDeRiego\":%d,\"tiempoDeRiego\":%.1f,\"moistureInicial\":%.1f,\
"moistureFinal\":%.1f}",
                        estaEncharcado.c_str(), horaDeRiego, tiempoDeRiego,
moistureInicial, moistureFinal);
        Particle.publish("addRiego", String(datosRiego), PRIVATE);
        delay(1000);
}

/*******************************************************************************
 * Function Name  : getAemetInfo
 * Description    : Descarga informacion meteorologica.
 * Parameters     : -.
 * Return         : Boolean. True si descarga la info. False en caso contrario.
```

```
 ******************************************************************************/
bool getAemetInfo()
{

        bool resultado = false;
        URLDestino = "No_inicializado";
        endPoint = "No_inicializado";
        datosAemet = "No_inicializado";

        // Lanzamos el webhook para obtener los datos
        Particle.publish("aemet", MODULO, PRIVATE);
        delay(MAX_TIME_AEMET_WAIT);

        if ((URLDestino != "No_inicializado") && (datosAemet != "No_inicializado"))
        {

                // Si descargamos la informacion AEMET --> resultado = true.
                resultado = true;

                Particle.publish(APP_NAME, "Descargada prediccion meteorologica: " +
String(probLluvia) + "Sistema: " + idPhoton, PRIVATE);
                delay(1000);
        }

        return resultado;
}

/*******************************************************************************
 * Function Name  : getSiarInfo
 * Description    : Descarga informacion SIAR.
 * Parameters     : -.
 * Return         : Boolean. True si descarga la info. False en caso contrario.
 ******************************************************************************/
bool getSiarInfo()
{
        bool resultado = false;
        datosSiar = "No_inicializado";

        // Lanzamos el webhook y esperamos la respuesta
        Particle.publish("siar", MODULO, PRIVATE);
        delay(MAX_TIME_FIREBASE_WAIT);

        if ((datosSiar != "No_inicializado"))
        {

                // Si descargamos la informacion SIAR --> resultado = true.
                resultado = true;
                Particle.publish(APP_NAME, "Descargada informacion SIAR: " + String(ETo)
+ "Sistema: " + idPhoton, PRIVATE);
                delay(1000);
        }

        return resultado;
}

/*******************************************************************************

  Fin de las funciones de acceso a servicios cloud.

*******************************************************************************/

/*******************************************************************************

  Funciones con utilidades varias.
  Nos permiten:
  .- Mantener el estado del sistema.
  .- Mantener las alertas del sistema.
  .- Mantener y evaluar parametros de riego.

*******************************************************************************/
/*******************************************************************************
 * Function Name  : setVentanaRiego
 * Description    : Actualiza la ventana de riego.
 * Params         : .-
 * Return         : .-
 ******************************************************************************/
```

```
void setVentanaRiego()
{
        elMes = Time.month();

        if ((elMes > 3) && (elMes <= 9))
        {

                // Estamos en primavera-verano
                inicioVentana = 20;
                finVentana = 23;
        }
        else
        {

                // Estamos en otoño-invierno
                inicioVentana = 13;
                finVentana = 16;
        }
}

/*****************************************************************************
 * Function Name  : setCamposCalculados
 * Description    : Calcula parametros necesarios para evaluar el riego.
 * Parameters     : -.
 * Return         : -.
 *****************************************************************************/
void setCamposCalculados()
{
        switch (elSuelo)
        {
        case gravoso:

                eficiencia = 0.85;
                saturacion = 3000 * 0.15;
                capacidadCampo = 3000 * 0.10;
                ptoDeRiego = 3000 * 0.05;
                marchitez = 3000 * 0.02;

                break;

        case arenoso:

                eficiencia = 0.90;
                saturacion = 3000 * 0.25;
                capacidadCampo = 3000 * 0.20;
                ptoDeRiego = 3000 * 0.10;
                marchitez = 3000 * 0.05;

                break;

        case franco:

                eficiencia = 0.95;
                saturacion = 3000 * 0.85;
                capacidadCampo = 3000 * 0.80;
                ptoDeRiego = 3000 * 0.50;
                marchitez = 3000 * 0.40;

                break;

        case arcilloso:

                eficiencia = 0.95;
                saturacion = 3000 * 0.85;
                capacidadCampo = 3000 * 0.60;
                ptoDeRiego = 3000 * 0.50;
                marchitez = 3000 * 0.40;

                break;

        default:

                eficiencia = 0.90;
                saturacion = 3000 * 0.30;
                capacidadCampo = 3000 * 0.20;
                ptoDeRiego = 3000 * 0.10;
                marchitez = 3000 * 0.05;
```

```
                break;
        }

        Particle.publish(APP_NAME, "Asignacion de campos calculados completa. Sistema: "
+ idPhoton, PRIVATE);
        delay(1000);
}

/*****************************************************************************
 * Function Name  : setEstado
 * Description    : Mantiene la variable de estado actualizada.
 * Params         : String: nuevo estado.
 * Return         : .-
 *****************************************************************************/
void setEstado(String nuevoEstado)
{
        estadoSistema = nuevoEstado;

        sendFbHito("ESTADOS", "Estado actual del sistema: " + estadoSistema);
        Particle.publish(APP_NAME, "Estado actual: " + estadoSistema + "Sistema: " +
idPhoton, PRIVATE);
        delay(1000);
}

/*****************************************************************************
 * Function Name  : checkEstado
 * Description    : Actualiza el vector de estado.
 * Parameters     : -.
 * Return         : Boolean: True si estado OK. False en caso contrario.
 *****************************************************************************/
bool checkEstado()
{

        bool resultado = true;

        for (int i = 0; i < NUM_SENS; i++)
        {
                switch (i)
                {
                case 0: // Umbral de temperatura.

                        if ((tempAire > umbrales[umbralTemp].max) || (tempAire <
umbrales[umbralTemp].min))
                        {

                                // El parametro esta fuera de la zona de control.
                                if (flags[umbralTemp].flag == false)
                                {

                                        // No estaba activo. Lo activamos y establecemos el
momento.
                                        flags[umbralTemp].flag = true;
                                        flags[umbralTemp].momento = 0;
                                }

                                // Si el parametro esta fuera de la zona de control,
devolvemos false.
                                resultado = false;
                        }
                        else
                        {

                                // El parametro esta en la zona de control.
                                if (flags[umbralTemp].flag == true)
                                {

                                        // El flag estaba activo. Lo desactivamos y
establecemos el momento.
                                        flags[umbralTemp].flag = false;
                                        flags[umbralTemp].momento = 0;
                                }
                        }

                        break;
```

```
                case 1: // Umbral de humedad.

                        if ((humedadAire > umbrales[umbralHumedad].max) || (humedadAire <
umbrales[umbralHumedad].min))
                        {

                                // El parametro esta fuera de la zona de control.
                                if (flags[umbralHumedad].flag == false)
                                {

                                        // No estaba activo. Lo activamos y establecemos el
momento.
                                        flags[umbralHumedad].flag = true;
                                        flags[umbralHumedad].momento = 0;
                                }

                                // Si el parametro esta fuera de la zona de control,
devolvemos false.
                                resultado = false;
                        }
                        else
                        {

                                // El parametro esta en la zona de control.
                                if (flags[umbralHumedad].flag == true)
                                {

                                        // El flag estaba activo. Lo desactivamos y
establecemos el momento.
                                        flags[umbralHumedad].flag = false;
                                        flags[umbralHumedad].momento = 0;
                                }
                        }

                        break;

                case 2: // Umbral de horas de luz.

                        if ((horasDeLuz > umbrales[umbralHorasLuz].max) || (horasDeLuz <
umbrales[umbralHorasLuz].min))
                        {

                                // El parametro esta fuera de la zona de control.
                                if (flags[umbralHorasLuz].flag == false)
                                {

                                        // No estaba activo. Lo activamos y establecemos el
momento.
                                        flags[umbralHorasLuz].flag = true;
                                        flags[umbralHorasLuz].momento = 0;
                                }

                                // Si el parametro esta fuera de la zona de control,
devolvemos false.
                                resultado = false;
                        }
                        else
                        {

                                // El parametro esta en la zona de control.
                                if (flags[umbralHorasLuz].flag == true)
                                {

                                        // El flag estaba activo. Lo desactivamos y
establecemos el momento.
                                        flags[umbralHorasLuz].flag = false;
                                        flags[umbralHorasLuz].momento = 0;
                                }
                        }

                        break;

                case 3: // Umbral de moisture.

                        if ((moisture > umbrales[umbralMoisture].max) || (moisture <
umbrales[umbralMoisture].min))
                        {
```

```
                                // El parametro esta fuera de la zona de control.
                                if (flags[umbralMoisture].flag == false)
                                {

                                        // No estaba activo. Lo activamos y establecemos el
momento.
                                        flags[umbralMoisture].flag = true;
                                        flags[umbralMoisture].momento = 0;
                                }

                                // Si el parametro esta fuera de la zona de control,
devolvemos false.
                                resultado = false;
                        }
                        else
                        {

                                // El parametro esta en la zona de control.
                                if (flags[umbralMoisture].flag == true)
                                {

                                        // El flag estaba activo. Lo desactivamos y
establecemos el momento.
                                        flags[umbralMoisture].flag = false;
                                        flags[umbralMoisture].momento = 0;
                                }
                        }

                        break;
                }
        }

        // Particle.publish(APP_NAME, "Vector de estado actualizado. Sistema: " +
idPhoton, PRIVATE);
        // delay(1000);

        return resultado;
}

/*******************************************************************************
 * Function Name  : resetFlags
 * Description    : Inicia el vector de estado.
 * Parameters     : -.
 * Return         : -.
 *******************************************************************************/
void resetFlags()
{

        for (int i = 0; i < NUM_SENS; i++)
        {

                flags[i].flag = false;
                flags[i].momento = 0;
        }

        Particle.publish(APP_NAME, "Vector de estado inicializado. Sistema: " + idPhoton,
PRIVATE);
        delay(1000);
}

/*******************************************************************************
 * Function Name  : setAlertas
 * Description    : Actualiza el vector de alerta.
 * Parameters     : -.
 * Return         : True si ha modificado el vector de alertas.
*******************************************************************************/
bool setAlertas()
{
        bool resultado = false;

        for (int i = 0; i < NUM_SENS; i++)
        {

                if (flags[i].flag == true)
                {
```

```
                        // El flag esta activo.
                        if (numHoras(flags[i].momento) >= INTERVALO_DE_ALERTA)
                        {

                                // Si lleva mas de un dia activo --> Subimos nivel de
alerta.
                                alertas[i].nivel = subirNivelAlerta(alertas[i].nivel);
                                flags[i].momento = 0;
                                resultado = true;

                                // Guardamos la configuracion porque la hemos modificado.
                                saveSettingsInEeprom();
                        }
                }
                else
                {

                        // El flag esta desactivado. Estamos en la zona segura.
                        if (numHoras(flags[i].momento) >= INTERVALO_DE_ALERTA)
                        {

                                // Si lleva mas de un dia en esa situacion --> bajamos
nivel de alerta.
                                alertas[i].nivel = bajarNivelAlerta(alertas[i].nivel);
                                flags[i].momento = 0;
                                resultado = true;

                                // Guardamos la configuracion porque la hemos modificado.
                                saveSettingsInEeprom();
                        }
                }
        }

        return resultado;
        Particle.publish(APP_NAME, "Vector de alertas actualizado. Sistema: " + idPhoton,
PRIVATE);
        delay(1000);
}

/*******************************************************************************
 * Function Name  : resetAlertas
 * Description    : Inicia el vector de alerta.
 * Parameters     : -.
 * Return         : -.
 *******************************************************************************/
void resetAlertas()
{

        for (int i = 0; i < NUM_SENS; i++)
        {
                alertas[i].nivel = SinAlerta;
                flags[i].momento = 0;
        }

        // Actualizamos los valores de control.
        setupAlertas = true;

        // Guardamos la configuracion.
        saveSettingsInEeprom();
        sendFbHito("ALERTA", "Vector de alertas inicializado.Sistema: " + idPhoton);
        Particle.publish(APP_NAME, "Vector de alertas inicializado. Sistema: " +
idPhoton, PRIVATE);
        delay(1000);
}

/*******************************************************************************
 * Function Name  : numHoras
 * Description    : Devuelve la edad en dias de la fecha pasada como parametro.
 * Parameters     : unsigned long fecha.
 * Return         : int: Numero de dias transcurridos desde la fecha.
 *******************************************************************************/
int numHoras(elapsedMillis momento)
{
        int horas;

        horas = momento / MILISEG_A_HORAS;
```

```
        return horas;
}

/*****************************************************************************
 * Function Name  : incrementarCte
 * Description    : Incrementa el valor de la constante.
 * Parameters     : double: valor.
 * Return         : double: Nuevo valor de la constante.
 *****************************************************************************/
double incrementarCte(double valor)
{

        const double maxCte = 1.5;

        if (valor >= maxCte)
        {

                return maxCte;
        }
        else
        {

                return valor + 0.1;
        }
}

/*****************************************************************************
 * Function Name  : decrementarCte
 * Description    : decrementa el valor de la constante.
 * Parameters     : double: valor.
 * Return         : double: Nuevo valor de la constante.
 *****************************************************************************/
double decrementarCte(double valor)
{

        const double minCte = 0.5;

        if (valor <= minCte)
        {

                return minCte;
        }
        else
        {

                return valor - 0.1;
        }
}

/*****************************************************************************
 * Function Name  : subirNivelAlerta
 * Description    : Incrementa el valor del nivel de alerta.
 * Parameters     : tipoAlerta: nivel.
 * Return         : tipoAlerta: Nuevo nivel de alerta.
 *****************************************************************************/
tipoAlerta subirNivelAlerta(tipoAlerta nivel)
{
        int variable;
        int maxAlerta = 4;
        tipoAlerta nuevoNivel;

        variable = alertaToInt(nivel);

        if (variable >= maxAlerta)
        {

                variable = 4;
        }
        else
        {

                variable += 1;
        }

        nuevoNivel = intToAlerta(variable);
```

```
        return nuevoNivel;
}

/*******************************************************************************
 * Function Name  : bajarNivelAlerta
 * Description    : Decrementa el valor del nivel de alerta.
 * Parameters     : tipoAlerta: nivel.
 * Return         : tipoAlerta: Nuevo nivel de alerta.
 *******************************************************************************/
tipoAlerta bajarNivelAlerta(tipoAlerta nivel)
{
        int variable;
        int minAlerta = 0;
        tipoAlerta nuevoNivel;

        variable = alertaToInt(nivel);

        if (variable <= minAlerta)
        {

                variable = 0;
        }
        else
        {

                variable -= 1;
        }

        nuevoNivel = intToAlerta(variable);

        return nuevoNivel;
}

/*******************************************************************************
 * Function Name  : initSystem
 * Description    : Inicializa el HW.
 * Parameters     : -.
 * Return         : true Ok, false Fallo.
*******************************************************************************/
bool initSystem()
{

        const int numIntentos = 10;
        int i;
        bool inicializadoUV, inicializadoBM, resultado;

        // Iniciamos el id del Sistema.
        idPhoton = System.deviceID();
        inicializadoUV = false;
        inicializadoBM = false;
        resultado = false;

        // Configuramos nuestra zona para los valores de tiempo.
        Time.zone(TIME_ZONE);

        // Configuramos los pines.
        pinMode(tempSensor, INPUT);
        pinMode(moistureSensor, INPUT);
        pinMode(tempHumSensor, INPUT);
        pinMode(releBomba, OUTPUT);
        pinMode(pinLed, OUTPUT);

        // Inicializamos los pines. Hay que comprobar que funcionan correctamente al
inicializarlos.

        // Sensor de temperatura y humedad DHT22.
        dht.begin();

        // Sensor de luz Si1145.
        for (i = 0; ((i < numIntentos) && (!inicializadoUV)); i++)
        {

                inicializadoUV = uv.begin();
                if (!inicializadoUV)
                {
```

```
                        sendFbHito("ERROR", "Sensor de luz no encontrado. Sistema: " +
idPhoton);
                        Particle.publish(APP_NAME, "Sensor de luz no encontrado. Sistema:
" + idPhoton, PRIVATE);
                        delay(1000);
                }
                else
                {

                        Particle.publish(APP_NAME, "Sensor de luz inicializado. Sistema: "
+ idPhoton, PRIVATE);
                        delay(1000);
                }
        }

        // Monitor de bateria.
        for (i = 0; ((i < numIntentos) && (!inicializadoBM)); i++)
        {

                inicializadoBM = batteryMonitor.begin();
                if (!inicializadoBM)
                {

                        sendFbHito("ERROR", "Monitor de bateria no encontrado. Sistema: "
+ idPhoton);
                        Particle.publish(APP_NAME, "Monitor de bateria no encontrado.
Sistema: " + idPhoton, PRIVATE);
                        delay(1000);
                }
                else
                {

                        // Inicia el monitor de bateria.
                        batteryMonitor.quickStart();
                        Particle.publish(APP_NAME, "Monitor de bateria inicializado.
Sistema: " + idPhoton, PRIVATE);
                        delay(1000);
                }
        }

        resultado = (inicializadoBM && inicializadoUV);
        return resultado;
}

/*****************************************************************************

  Fin de las funciones de utilidades.

*****************************************************************************/

/*****************************************************************************

  Manejadores de eventos.
  Nos permiten:
  .- Por medio de estas funciones extraemos la informacion de respuesta de
     los distintos webhooks lanzados.

*****************************************************************************/

/*****************************************************************************
 * Function Name  : myHandlerAemet
 * Description    : Extrae el endpoint de acceso a la prediccion actual.
 *                  Lanza el webhook datosLluvia pasandole el endpoint.
 * Parameters     : char *event - El nombre del evento.
 *                  char *data - informacion de respuesta.
 * Return         : -.
 *****************************************************************************/
void myHandlerAemet(const char *event, const char *data)
{

        int indice = -1;

        // Obtenemos el endpoint de la respuesta
        URLDestino = strdup(data);
        indice = URLDestino.lastIndexOf("/") + 1;
        if (indice > 0)
```

```
        {

                // Hemos encontrado el endpoint
                endPoint = URLDestino.substring(indice);

                // Lanzamos el segundo webhook pasando el endpoint actual
                Particle.publish("datosLluvia", endPoint, PRIVATE);
                delay(1000);
        }
}

/*******************************************************************************
 * Function Name  : myHandlerDatosLluvia
 * Description    : Extrae los datos de prediccion del JSON de respuesta.
 * Parameters     : char *event - El nombre del evento.
 *                  char *data - informacion de respuesta.
 * Return         : -.
 *******************************************************************************/
void myHandlerDatosLluvia(const char *event, const char *data)
{

        String fechaPrediccion;
        int _12_24_today;
        int _00_12_tomorrow;
        int _12_24_tomorrow;

        // Handle the integration response
        datosAemet = strdup(data);
        parser.clear();
        parser.addString(datosAemet);

        if (parser.parse())
        {

                // Los datos contienen un JSON procesable
                fechaPrediccion = parser.getReference().key("fecha").valueString();
                _12_24_today =
parser.getReference().key("datos").index(0).key("value").valueInt();
                _00_12_tomorrow =
parser.getReference().key("datos").index(1).key("value").valueInt();
                _12_24_tomorrow =
parser.getReference().key("datos").index(2).key("value").valueInt();

                if ((elMes > 3) && (elMes <= 9))
                {

                        // Estamos en primavera-verano
                        probLluvia = (_00_12_tomorrow + _12_24_tomorrow) / 2;
                }
                else
                {

                        // Estamos en otoño-invierno
                        probLluvia = (_12_24_today + _00_12_tomorrow) / 2;
                }
        }
}

/*******************************************************************************
 * Function Name  : myHandlerSiar
 * Description    : Extrae la ETo diaria de la respuesta.
 * Parameters     : char *event - El nombre del evento.
 *                  char *data - informacion de respuesta.
 * Return         : -.
 *******************************************************************************/
void myHandlerSiar(const char *event, const char *data)
{

        // Obtenemos la ETo de la respuesta
        datosSiar = strdup(data);
        ETo = atof(datosSiar);
}

/*******************************************************************************
 * Function Name  : myHandlerError
 * Description    : Manejador de errores para nuestros webhooks.
 * Parameters     : char *event - El nombre del evento.
```

```
 *                  char *data - informacion de respuesta.
 * Return         : -.
 ******************************************************************************/
void myHandlerError(const char *event, const char *data)
{

        String descripcion;

        descripcion = String(event) + " " + String(data);

        // Publicamos un hito con el error del webhook.
        sendFbHito("ERROR", descripcion);
        Particle.publish("ERROR", descripcion, PRIVATE);
        delay(1000);
}

/*******************************************************************************

  Fin de los manejadores de eventos.

*******************************************************************************/
```

Anexo II. Funciones suscriptoras.

```
/*********************************************************************************************
 * Proyecto: AutomataFinitoDeRiego
 * Descripcion: Conjunto de funciones disparadas por eventos Pub/Sub. Actualizan la base
 *              de datos del sistema.
 * Autor: Marcos Colmenero Fernandez.
 * Fecha: 2018 - 2020
*********************************************************************************************/

'use strict';

// Importamos las functions del SDK.
const functions = require('firebase-functions');

// Importamos Firebase Admin SDK para acceder a Firestore.
const admin = require('firebase-admin');

// Inicializamos la App con la configuración por defecto.
var app = admin.initializeApp();

// Inicializamos firestore.
var firedb = app.firestore();

// Escribimos en realtimeDB.
// var db = app.database();
// var ref = db.ref("/SIAR");
// ref.set({ETo: 5});

/*******************************************************************************
 *
 * Function Name  : addHito
 * Description    : Añade los datos de un nuevo Hito en el sistema.
 *
 *******************************************************************************/

exports.addHito = functions.pubsub.topic('hitos').onPublish((message) => {

  var laDescripcion;
  var elTipo;
  var momento;

  // Decodificamos el mensaje.
  const messageBody = message.data ? Buffer.from(message.data, 'base64').toString() :
null;

  // Print the message in the logs.
  console.log(`addHito:  ${messageBody || ' Creado'}!`);

  // Decodificamos el JSON.
```

```
  try {

    laDescripcion = message.json.descripcion;
    elTipo = message.json.tipo;

  } catch (e) {

    console.error('PubSub hito no es JSON', e);

  }

  // Insertamos el registro.
  // firedb.collection().doc().update();
  var addDoc = firedb.collection('Modulos').doc('Sist01-
Mod01').collection('Hitos').add({
    descripcion: laDescripcion,
    momento: admin.firestore.FieldValue.serverTimestamp(),
    tipo: elTipo,
  });

  // Devolvemos la promesa.
  return addDoc;
});

/**************************************************************************
 *
 * Function Name  : addEstado
 * Description    : Añade los datos de un nuevo Estado en el sistema.
 *
 **************************************************************************/

exports.addEstado = functions.pubsub.topic('estados').onPublish((message) => {

  // Variables de estado.
  var tempTierra;
  var tempAire;
  var humedadAire;
  var luzVisible;
  var luzIR;
  var UVIndex;
  var moisture;
  var horasDeLuz;
  var momento;

  // Decodificamos el mensaje.
  const messageBody = message.data ? Buffer.from(message.data, 'base64').toString() :
null;

  // Print the message in the logs.
  console.log(`addEstado:  ${messageBody || ' Creado'}!`);

  // Decodificamos el JSON.
  try {

    tempTierra = message.json.tempTierra;
    tempAire = message.json.tempAire;
    humedadAire = message.json.humedadAire;
    luzVisible = message.json.luzVisible;
    luzIR = message.json.luzIR;
    UVIndex = message.json.UVIndex;
    moisture = message.json.moisture;
    horasDeLuz = message.json.horasDeLuz;
    momento = admin.firestore.FieldValue.serverTimestamp();

  } catch (e) {

    console.error('PubSub estado no es JSON', e);

  }

  // Insertamos el registro.
  // firedb.collection().doc().update();
  var addDoc = firedb.collection('Modulos').doc('Sist01-
Mod01').collection('Estados').add({
    tempTierra: tempTierra,
    tempAire: tempAire,
    humedadAire: humedadAire,
```

```
    luzVisible: luzVisible,
    luzIR: luzIR,
    UVIndex: UVIndex,
    moisture: moisture,
    horasDeLuz: horasDeLuz,
    momento: momento,
  });

  // Devolvemos la promesa.
  return addDoc;
});

/**************************************************************************
 *
 * Function Name  : addRiego
 * Description    : Añade los datos de un nuevo Riego en el sistema.
 *
 **************************************************************************/

exports.addRiego = functions.pubsub.topic('riegos').onPublish((message) => {

  // Variables de riego.
  var encharcado;
  var horaDeRiego;   // Valor numerico entre 0 y 23.
  var tiempoDeRiego; // En minutos.
  var moistureInicial;
  var moistureFinal;
  var momento;

  // Decodificamos el mensaje.
  const messageBody = message.data ? Buffer.from(message.data, 'base64').toString() :
null;

  // Print the message in the logs.
  console.log(`addRiego:  ${messageBody || ' Creado'}!`);

  // Decodificamos el JSON.
  try {

    encharcado = message.json.encharcado;
    horaDeRiego = message.json.horaDeRiego;
    tiempoDeRiego = message.json.tiempoDeRiego;
    moistureInicial = message.json.moistureInicial;
    moistureFinal = message.json.moistureFinal;
    momento = admin.firestore.FieldValue.serverTimestamp();

  } catch (e) {

    console.error('PubSub riego no es JSON', e);

  }

  // Insertamos el registro.
  // firedb.collection().doc().update();
  var addDoc = firedb.collection('Modulos').doc('Sist01-
Mod01').collection('Riegos').add({
    encharcado: encharcado,
    horaDeRiego: horaDeRiego,
    tiempoDeRiego: tiempoDeRiego,
    moistureInicial: moistureInicial,
    moistureFinal: moistureFinal,
    momento: momento,
  });

  // Devolvemos la promesa.
  return addDoc;
});



/**************************************************************************
 *
 * Function Name  : setAlerta
 * Description    : Actualiza los datos de Alerta en el sistema.
 *
```

```
 ********************************************************************************/

exports.setAlerta = functions.pubsub.topic('alerta').onPublish((message) => {

  // Variables de alerta.
  var alertaHorasDeLuz;
  var alertaHumedad;
  var alertaMoisture;
  var alertaTemperatura;

  // Decodificamos el mensaje.
  const messageBody = message.data ? Buffer.from(message.data, 'base64').toString() :
null;

  // Print the message in the logs.
  console.log(`updateAlerta:  ${messageBody || ' Actualizada'}!`);

  // Decodificamos el JSON.
  try {

    alertaHorasDeLuz = message.json.horasDeLuz;
    alertaHumedad = message.json.humedad;
    alertaMoisture = message.json.moisture;
    alertaTemperatura = message.json.temperatura;

  } catch (e) {

    console.error('PubSub alerta no es JSON', e);

  }

  // Actualizamos el registro.
  var addDoc = firedb.collection('Modulos').doc('Sist01-
Mod01').collection('Alertas').doc('W4Y8In4kF1iO1urp9LoX').update({
    horasDeLuz: alertaHorasDeLuz,
    humedad: alertaHumedad,
    moisture: alertaMoisture,
    temperatura: alertaTemperatura,
  });

  // Devolvemos la promesa.
  return addDoc;
});
```

Anexo III. Aplicación de control.

Archivo: huerto-charts.js

```
// Ajusta la fuente y el color de fondo
Chart.defaults.global.defaultFontFamily = 'Nunito', '-apple-system,system-
ui,BlinkMacSystemFont,"Segoe UI",Roboto,"Helvetica Neue",Arial,sans-serif';
Chart.defaults.global.defaultFontColor = '#858796';

function number_format(number, decimals, dec_point, thousands_sep) {
  // *     example: number_format(1234.56, 2, ',', ' ');
  // *     return: '1 234,56'
  number = (number + '').replace(',', '').replace(' ', '');
  var n = !isFinite(+number) ? 0 : +number,
    prec = !isFinite(+decimals) ? 0 : Math.abs(decimals),
    sep = (typeof thousands_sep === 'undefined') ? ',' : thousands_sep,
    dec = (typeof dec_point === 'undefined') ? '.' : dec_point,
    s = '',
    toFixedFix = function (n, prec) {
      var k = Math.pow(10, prec);
      return '' + Math.round(n * k) / k;
    };
  // Fix for IE parseFloat(0.55).toFixed(0) = 0;
  s = (prec ? toFixedFix(n, prec) : '' + Math.round(n)).split('.');
  if (s[0].length > 3) {
    s[0] = s[0].replace(/\B(?=(?:\d{3})+(?!\d))/g, sep);
  }
```

```
  if ((s[1] || '').length < prec) {
    s[1] = s[1] || '';
    s[1] += new Array(prec - s[1].length + 1).join('0');
  }
  return s.join(dec);
}

// Pie Chart Riegos
var ctxRiegoDiario = document.getElementById('myPieChartRiegos');
var myPieChartRiegos = new Chart(ctxRiegoDiario, {
  type: 'doughnut',
  data: {
    labels: ["Reposo", "Riego", "Reposo"],
    datasets: [{
      data: [20, 2, 2], // horas de reposo - riego
      backgroundColor: [, '#36b9cc'],
      hoverBackgroundColor: [, '#2c9faf'],
      hoverBorderColor: "rgba(234, 236, 244, 1)"
    }]
  },
  options: {
    maintainAspectRatio: false,
    responsive: true,
    tooltips: {
      backgroundColor: "rgb(255,255,255)",
      bodyFontColor: "#858796",
      borderColor: '#dddfeb',
      borderWidth: 1,
      xPadding: 15,
      yPadding: 15,
      displayColors: false,
      caretPadding: 10
    },
    legend: {
      display: false
    },
    cutoutPercentage: 70
  }
});

// Area Chart Riegos
var ctxRiegos = document.getElementById('myAreaChartRiegos');
var myLineChartRiegos = new Chart(ctxRiegos, {
  type: 'line',
  data: {
    labels: ["7d", "6d", "5d", "4d", "3d", "2d", "1d"],
    datasets: [{
      data: [1, 5, 3, 5, 2, 3, 4],
      label: "Semanal",
      lineTension: 0.3,
      backgroundColor: "rgba(0, 137, 132, .2)",
      borderColor: "rgba(78, 115, 223, 1)",
      pointRadius: 3,
      pointBackgroundColor: "rgba(78, 115, 223, 1)",
      pointBorderColor: "rgba(78, 115, 223, 1)",
      pointHoverRadius: 3,
      pointHoverBackgroundColor: "rgba(78, 115, 223, 1)",
      pointHoverBorderColor: "rgba(78, 115, 223, 1)",
      pointHitRadius: 10,
      pointBorderWidth: 2
    }]
  },
  options: {
    responsive: true,
    maintainAspectRatio: false,
    layout: {
      padding: {
        left: 10,
        right: 25,
        top: 25,
        bottom: 0
      }
    },
    scales: {
      xAxes: [{
        gridLines: {
```

```
          display: false,
          drawBorder: false
        },
        ticks: {
          maxTicksLimit: 6
        }
      }],
      yAxes: [{
        ticks: {
          maxTicksLimit: 6,
          padding: 10,
          // Personaliza el msg
          callback: function (value, index, values) {
            return number_format(value) + ' min';
          }
        },
        gridLines: {
          color: "rgb(234, 236, 244)",
          zeroLineColor: "rgb(234, 236, 244)",
          drawBorder: false,
          borderDash: [2],
          zeroLineBorderDash: [2]
        }
      }]
    },
    legend: {
      display: false
    },
    tooltips: {
      backgroundColor: "rgb(255,255,255)",
      bodyFontColor: "#858796",
      titleMarginBottom: 10,
      titleFontColor: '#6e707e',
      titleFontSize: 14,
      borderColor: '#dddfeb',
      borderWidth: 1,
      xPadding: 15,
      yPadding: 15,
      displayColors: false,
      intersect: false,
      mode: 'index',
      caretPadding: 10,
      callbacks: {
        label: function (tooltipItem, chart) {
          var datasetLabel = chart.datasets[tooltipItem.datasetIndex].label || '';
          return number_format(tooltipItem.yLabel) + ' min';
        }
      }
    }
  }
});

// Bar Chart Recolecciones
var ctxRecolecciones = document.getElementById('myBarChartRecolecciones');
var myBarChartRecolecciones = new Chart(ctxRecolecciones, {
  type: 'bar',
  data: {
    labels: ["Ene", "Feb", "Mar", "Abr", "May", "Jun", "Jul", "Ago", "Sep", "Oct",
"Nov", "Dic"],
    datasets: [{
      label: "Kg ",
      backgroundColor: "#4e73df",
      hoverBackgroundColor: "#2e59d9",
      borderColor: "#4e73df",
      data: [10, 5, 6, 8, 3, 10, 5, 6, 8, 3, 6, 5],
    }]
  },
  options: {
    responsive: true,
    maintainAspectRatio: false,
    layout: {
      padding: {
        left: 10,
        right: 25,
        top: 25,
        bottom: 0
      }
```

```
    },
    scales: {
      xAxes: [{
        time: {
          unit: 'month'
        },
        gridLines: {
          display: false,
          drawBorder: false
        },
        ticks: {
          maxTicksLimit: 12
        },
        maxBarThickness: 25,
      }],
      yAxes: [{
        ticks: {
          min: 0,
          maxTicksLimit: 5,
          padding: 10,
          // Personaliza el msg.
          callback: function (value, index, values) {
            return ' ' + number_format(value) + 'Kg';
          }
        },
        gridLines: {
          color: "rgb(234, 236, 244)",
          zeroLineColor: "rgb(234, 236, 244)",
          drawBorder: false,
          borderDash: [2],
          zeroLineBorderDash: [2]
        }
      }]
    },
    legend: {
      display: false
    },
    tooltips: {
      titleMarginBottom: 10,
      titleFontColor: '#6e707e',
      titleFontSize: 14,
      backgroundColor: "rgb(255,255,255)",
      bodyFontColor: "#858796",
      borderColor: '#dddfeb',
      borderWidth: 1,
      xPadding: 15,
      yPadding: 15,
      displayColors: false,
      caretPadding: 10,
      callbacks: {
        label: function (tooltipItem, chart) {
          var datasetLabel = chart.datasets[tooltipItem.datasetIndex].label || '';
          return number_format(tooltipItem.yLabel) + datasetLabel;
        }
      }
    }
  }
});

// Pie Chart Siembras
var ctxSiembras = document.getElementById('myPieChartSiembras');
var myPieChartSiembras = new Chart(ctxSiembras, {
  type: 'doughnut',
  data: {
    labels: ["Sembrado", "Libre"],
    datasets: [{
      data: [55, 45],
      backgroundColor: ['#1cc88a', '#dddfeb'],
      hoverBackgroundColor: ['#17a673', '#858796'],
      hoverBorderColor: "rgba(234, 236, 244, 1)",
    }]
  },
  options: {
    responsive: true,
    maintainAspectRatio: false,
    tooltips: {
```

```
      backgroundColor: "rgb(255,255,255)",
      bodyFontColor: "#858796",
      borderColor: '#dddfeb',
      borderWidth: 1,
      xPadding: 15,
      yPadding: 15,
      displayColors: false,
      caretPadding: 10
    },
    legend: {
      display: false
    },
    cutoutPercentage: 70
  }
});

// Bar Chart Horas de luz
var ctxHorasDeLuz = document.getElementById('myBarChartHorasDeLuz');
var myBarChartHorasDeLuz = new Chart(ctxHorasDeLuz, {
  type: 'bar',
  data: {
    labels: ["Lunes", "Martes", "Miercoles", "Jueves", "Viernes", "Sabado", "Domingo"],
    datasets: [{
      label: "H ",
      backgroundColor: "#4e73df",
      hoverBackgroundColor: "#2e59d9",
      borderColor: "#4e73df",
      data: [10, 5, 6, 8, 3, 10, 5],
    }]
  },
  options: {
    responsive: true,
    maintainAspectRatio: false,
    layout: {
      padding: {
        left: 10,
        right: 25,
        top: 25,
        bottom: 0
      }
    },
    scales: {
      xAxes: [{
        gridLines: {
          display: false,
          drawBorder: false
        },
        ticks: {
          maxTicksLimit: 6
        },
        maxBarThickness: 25,
      }],
      yAxes: [{
        ticks: {
          min: 0,
          max: 15,
          maxTicksLimit: 5,
          padding: 10,
          // Personaliza el msg.
          callback: function (value, index, values) {
            return ' ' + number_format(value) + 'H';
          }
        },
        gridLines: {
          color: "rgb(234, 236, 244)",
          zeroLineColor: "rgb(234, 236, 244)",
          drawBorder: false,
          borderDash: [2],
          zeroLineBorderDash: [2]
        }
      }]
    },
    legend: {
      display: false
    },
    tooltips: {
      titleMarginBottom: 10,
```

```
      titleFontColor: '#6e707e',
      titleFontSize: 14,
      backgroundColor: "rgb(255,255,255)",
      bodyFontColor: "#858796",
      borderColor: '#dddfeb',
      borderWidth: 1,
      xPadding: 15,
      yPadding: 15,
      displayColors: false,
      caretPadding: 10,
      callbacks: {
        label: function (tooltipItem, chart) {
          var datasetLabel = chart.datasets[tooltipItem.datasetIndex].label || '';
          return number_format(tooltipItem.yLabel) + datasetLabel;
        }
      }
    }
  }
});

// Radar Chart Estados
var ctxDeHitos = document.getElementById('radarChartFuncionamiento');
var myRadarChartHitos = new Chart(ctxDeHitos, {
  type: 'radar',
  data: {
    labels: ["ALERTA", "ERROR", "SISTEMA", "ESTADOS", "ILUMINACION"],
    datasets: [{
      label: "Repeticiones ",
      data: [50, 50, 50, 50, 50],
      backgroundColor: [
        'rgba(105, 0, 132, .2)',
      ],
      borderColor: [
        'rgba(200, 99, 132, .7)',
      ],
      borderWidth: 2
    }]
  },
  options: {
    responsive: true,
    maintainAspectRatio: false,
    tooltips: {
      backgroundColor: "rgb(255,255,255)",
      bodyFontColor: "#858796",
      borderColor: '#dddfeb',
      borderWidth: 1,
      xPadding: 15,
      yPadding: 15,
      displayColors: false,
      caretPadding: 10
    },
    legend: {
      display: false
    }
  }
});

// Area Chart Ambiente
var ctxAmbiente = document.getElementById('myLineChartAmbiente');
var myLineChartAmb = new Chart(ctxAmbiente, {
  type: 'line',
  data: {
    labels: ["Lunes", "Martes", "Miercoles", "Jueves", "Viernes", "Sabado", "Domingo"],
    datasets: [{
      label: "Humedad",
      lineTension: 0.3,
      backgroundColor: "rgba(0, 137, 132, .2)",
      borderColor: "rgba(78, 115, 223, 1)",
      pointRadius: 3,
      pointBackgroundColor: "rgba(78, 115, 223, 1)",
      pointBorderColor: "rgba(78, 115, 223, 1)",
      pointHoverRadius: 3,
      pointHoverBackgroundColor: "rgba(78, 115, 223, 1)",
      pointHoverBorderColor: "rgba(78, 115, 223, 1)",
      pointHitRadius: 10,
      pointBorderWidth: 2,
```

```
      data: [0, 5000, 5000, 10000, 5000, 5000, 10000],
    },
    {
      label: "Temperatura",
      lineTension: 0.3,
      backgroundColor: "rgba(105, 0, 132, .2)",
      borderColor: "rgba(200, 99, 132, .7)",
      pointRadius: 3,
      pointBackgroundColor: "rgba(200, 99, 132, .7)",
      pointBorderColor: "rgba(200, 99, 132, .7)",
      pointHoverRadius: 3,
      pointHoverBackgroundColor: "rgba(200, 99, 132, .7)",
      pointHoverBorderColor: "rgba(200, 99, 132, .7)",
      pointHitRadius: 10,
      pointBorderWidth: 2,
      data: [0, 5000, 10000, 20000, 25000, 30000, 40000],
    }]
  },
  options: {
    responsive: true,
    maintainAspectRatio: false,
    layout: {
      padding: {
        left: 10,
        right: 25,
        top: 25,
        bottom: 0
      }
    },
    scales: {
      xAxes: [{
        gridLines: {
          display: false,
          drawBorder: false
        },
        ticks: {
          maxTicksLimit: 6
        }
      }],
      yAxes: [{
        ticks: {
          maxTicksLimit: 6,
          padding: 10,
          // Personaliza el msg
          callback: function (value, index, values) {
            return number_format(value);
          }
        },
        gridLines: {
          color: "rgb(234, 236, 244)",
          zeroLineColor: "rgb(234, 236, 244)",
          drawBorder: false,
          borderDash: [2],
          zeroLineBorderDash: [2]
        }
      }]
    },
    legend: {
      display: false
    },
    tooltips: {
      backgroundColor: "rgb(255,255,255)",
      bodyFontColor: "#858796",
      titleMarginBottom: 10,
      titleFontColor: '#6e707e',
      titleFontSize: 14,
      borderColor: '#dddfeb',
      borderWidth: 1,
      xPadding: 15,
      yPadding: 15,
      displayColors: false,
      intersect: false,
      mode: 'index',
      caretPadding: 10,
      callbacks: {
        label: function (tooltipItem, chart) {
          var datasetLabel = chart.datasets[tooltipItem.datasetIndex].label || '';
```

```
          return datasetLabel + ' ' + number_format(tooltipItem.yLabel);
        }
      }
    }
  }
});

// Area Chart Contenedor
var ctxContenedor = document.getElementById('myLineChartContenedor');
var myLineChartCont = new Chart(ctxContenedor, {
  type: 'line',
  data: {
    labels: ["Lunes", "Martes", "Miercoles", "Jueves", "Viernes", "Sabado", "Domingo"],
    datasets: [{
      label: "Maxima",
      lineTension: 0.3,
      backgroundColor: "rgba(0, 137, 132, .2)",
      borderColor: "rgba(78, 115, 223, 1)",
      pointRadius: 3,
      pointBackgroundColor: "rgba(78, 115, 223, 1)",
      pointBorderColor: "rgba(78, 115, 223, 1)",
      pointHoverRadius: 3,
      pointHoverBackgroundColor: "rgba(78, 115, 223, 1)",
      pointHoverBorderColor: "rgba(78, 115, 223, 1)",
      pointHitRadius: 10,
      pointBorderWidth: 2,
      data: [0, 5000, 5000, 10000, 5000, 5000, 10000],
    },
    {
      label: "Minima",
      lineTension: 0.3,
      backgroundColor: "rgba(105, 0, 132, .2)",
      borderColor: "rgba(200, 99, 132, .7)",
      pointRadius: 3,
      pointBackgroundColor: "rgba(200, 99, 132, .7)",
      pointBorderColor: "rgba(200, 99, 132, .7)",
      pointHoverRadius: 3,
      pointHoverBackgroundColor: "rgba(200, 99, 132, .7)",
      pointHoverBorderColor: "rgba(200, 99, 132, .7)",
      pointHitRadius: 10,
      pointBorderWidth: 2,
      data: [0, 5000, 10000, 20000, 25000, 30000, 40000],
    }]
  },
  options: {
    responsive: true,
    maintainAspectRatio: false,
    layout: {
      padding: {
        left: 10,
        right: 25,
        top: 25,
        bottom: 0
      }
    },
    scales: {
      xAxes: [{
        gridLines: {
          display: false,
          drawBorder: false
        },
        ticks: {
          maxTicksLimit: 6
        }
      }],
      yAxes: [{
        ticks: {
          maxTicksLimit: 6,
          padding: 10,
          // Personaliza el msg
          callback: function (value, index, values) {
            return number_format(value);
          }
        },
        gridLines: {
          color: "rgb(234, 236, 244)",
```

```
          zeroLineColor: "rgb(234, 236, 244)",
          drawBorder: false,
          borderDash: [2],
          zeroLineBorderDash: [2]
        }
      }]
    },
    legend: {
      display: false
    },
    tooltips: {
      backgroundColor: "rgb(255,255,255)",
      bodyFontColor: "#858796",
      titleMarginBottom: 10,
      titleFontColor: '#6e707e',
      titleFontSize: 14,
      borderColor: '#dddfeb',
      borderWidth: 1,
      xPadding: 15,
      yPadding: 15,
      displayColors: false,
      intersect: false,
      mode: 'index',
      caretPadding: 10,
      callbacks: {
        label: function (tooltipItem, chart) {
          var datasetLabel = chart.datasets[tooltipItem.datasetIndex].label || '';
          return datasetLabel + ' ' + number_format(tooltipItem.yLabel);
        }
      }
    }
  }
});
```

Archivo: huerto-fb.js

```
// Web app's Firebase configuracion
var firebaseConfig = {OBJETO DE CONFIGURACION FIREBASE};

// Inicializamos Firebase
firebase.initializeApp(firebaseConfig);

// Estado de autenticacion
auth = firebase.auth();

/**************************************************************************
 * Function Name  : initApp
 * Description    : Controla el estado de sesion de usuario.
 * Parameters     :
 * Return         :
 **************************************************************************/
initApp = function () {

  // conectamos a la base de datos
  db = firebase.firestore();

  auth.onAuthStateChanged(function (user) {

    if (user) {

      // El usuario ha iniciado sesion.
      document.getElementById("panelPrincipal").style.display = "block";

    } else {

      document.getElementById("panelPrincipal").style.display = "none";

      // El usuario NO ha iniciado sesion.
      window.location = "signin.html";
    }

  }, function (error) {

    console.log(error);
```

```
  });

};

/**************************************************************************
 * Function Name  : clickSignoutButton
 * Description    : Cierra la sesion de usuario
 * Parameters     :
 * Return         :
 **************************************************************************/
// Sign out
function clickSignoutButton() {

  if (auth.currentUser) {

    auth.signOut();
  }

}

/**************************************************************************
 * Function Name  : getModulo
 * Description    : Obtiene los valores de configuracion del modulo en fb.
 *                  Actualiza la vista.
 * Parameters     :
 * Return         :
 **************************************************************************/
async function getModulo() {

  // Iniciamos la referencia a la base de datos.
  const moduloRef = db.collection('Modulos').doc('Sist01-Mod01');

  var elModulo = moduloRef
    .get()
    .then(function (modData) {
      $("#valNe").text(modData.data().Ne);
      $("#valM2").text(modData.data().m2);
      $("#valCaudal").text(modData.data().q);

      switch (modData.data().tipoSuelo) {

        case "0":
          $("#valTipoSuelo").text("Gravoso");
          break;

        case "1":
          $("#valTipoSuelo").text("Arenoso");
          break;

        case "2":
          $("#valTipoSuelo").text("Franco");
          break;

        case "3":
          $("#valTipoSuelo").text("Arcilloso");
          break;
      }

    })
    .catch(function (err) {
      console.log('Error getting modulo', err);
    });
}

/**************************************************************************
 * Function Name  : getAlertas
 * Description    : Obtiene el valor de las alertas. Actualiza la vista.
 * Parameters     :
 * Return         :
 **************************************************************************/
async function getAlertas() {

  var horasDeLuz;
  var humedad;
  var moisture;
  var temperatura;
```

```
  // Iniciamos la referencia a la base de datos.
  const alertasRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Alertas').doc('W4Y8In4kF1iO1urp9LoX');

  var lasAlertas = alertasRef
    .get()
    .then(function (alertaData) {

      horasDeLuz = alertaData.data().horasDeLuz;
      humedad = alertaData.data().humedad;
      moisture = alertaData.data().moisture;
      temperatura = alertaData.data().temperatura;

      // Badge horasDeLuz
      if (horasDeLuz > 2) {

        $("#valHorasDeLuz").removeClass("badge-info");
        $("#valHorasDeLuz").addClass("badge-danger");
      } else {

        $("#valHorasDeLuz").removeClass("badge-danger");
        $("#valHorasDeLuz").addClass("badge-info");
      }

      // Badge humedad
      if (humedad > 2) {

        $("#valHumedad").removeClass("badge-info");
        $("#valHumedad").addClass("badge-danger");
      } else {

        $("#valHumedad").removeClass("badge-danger");
        $("#valHumedad").addClass("badge-info");
      }

      // Badge moisture
      if (moisture > 2) {

        $("#valMoisture").removeClass("badge-info");
        $("#valMoisture").addClass("badge-danger");
      } else {

        $("#valMoisture").removeClass("badge-danger");
        $("#valMoisture").addClass("badge-info");
      }

      // Badge temperatura
      if (temperatura > 2) {

        $("#valTemperatura").removeClass("badge-info");
        $("#valTemperatura").addClass("badge-danger");
      } else {

        $("#valTemperatura").removeClass("badge-danger");
        $("#valTemperatura").addClass("badge-info");
      }

      // Les asignamos los valores
      $("#valHorasDeLuz").text(horasDeLuz);
      $("#valHumedad").text(humedad);
      $("#valMoisture").text(moisture);
      $("#valTemperatura").text(temperatura);
    })
    .catch(function (err) {
      console.log('Error getting alertas', err);
    });
}

/*****************************************************************************
 * Function Name  : getRiegos
 * Description    : Obtiene el listado de riegos. Los ultimos 7. Actualiza la
 *                  vista.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getRiegos() {
```

```
  var horaDeRiego;
  var tiempoDeRiego;
  var reposoFinal;
  var riegosSemanal = [0, 0, 0, 0, 0, 0, 0];

  // Iniciamos la referencia a la base de datos. Obtenemos los 7 ultimos riegos.
  const riegosRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Riegos').orderBy('momento', 'desc').limit(7);

  var losRiegos = riegosRef
    .get()
    .then(function (riegosData) {

      // Obtenemos los datos del ultimo riego.
      horaDeRiego = riegosData.docs[0].data().horaDeRiego;
      tiempoDeRiego = riegosData.docs[0].data().tiempoDeRiego / 60;
      reposoFinal = 24 - (horaDeRiego + tiempoDeRiego);

      // Actualizamos el grafico de riego diario con los datos del ultimo riego.
      myPieChartRiegos.data.datasets[0].data = [horaDeRiego, tiempoDeRiego,
reposoFinal];

      // Actualiza la vista.
      myPieChartRiegos.update();

      // Actualizamos el grafico de riego semanal con los datos de los ultimos 7 riegos.
      var i = 6;
      riegosData.forEach(function (doc) {

        riegosSemanal[i] = doc.data().tiempoDeRiego;
        i--;
      });

      // Actualiza la vista.
      myLineChartRiegos.data.datasets[0].data = riegosSemanal;
      myLineChartRiegos.update();
    })
    .catch(function (err) {

      console.log('Error getting riegos', err);
    });
}

/*****************************************************************************
 * Function Name  : getRecolecciones
 * Description    : Obtiene el listado anual de recolecciones.
 *                  Actualiza la vista.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getRecolecciones() {

  var elMes;
  var recoleccionesMes = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0];
  // Recolecciones del año
  const anio = new Date().getFullYear().toString();
  const inicio = firebase.firestore.Timestamp.fromDate(new Date(anio));
  const recoleccionesRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Recolecciones')
    .where('momento', '>=', inicio)
    .orderBy('momento', 'desc');

  recoleccionesRef
    .get()
    .then(function (recoleccionesData) {

      recoleccionesData.forEach(function (doc) {
        elMes = doc.data().momento.toDate().getMonth();
        recoleccionesMes[elMes] += parseFloat(doc.data().Kg);
      });

      console.log(recoleccionesMes);
      // Actualiza la vista.
      myBarChartRecolecciones.data.datasets[0].data = recoleccionesMes;
      myBarChartRecolecciones.update();
```

```
    })
    .catch(function (err) {

      console.log('Error getting recolecciones', err);
    });
}

/*****************************************************************************
 * Function Name  : getSiembras
 * Description    : Obtiene el listado de siembras. Actualiza la vista.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getSiembras() {

  // Todas las siembras del modulo
  const siembrasRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Siembras');
  var m2Acumulado;

  var lasSiembras = siembrasRef
    .get()
    .then(function (siembrasData) {

      m2Acumulado = 0;
      siembrasData.forEach(function (doc) {

        m2Acumulado += parseFloat(doc.data().m2);
      });
      var libre = parseFloat($("#valM2").text()) - m2Acumulado;

      // Actualizamos el grafico de siembras.
      myPieChartSiembras.data.datasets[0].data = [m2Acumulado, libre];
      myPieChartSiembras.update();

      // return libre;
    })
    .catch(function (err) {

      console.log('Error getting siembras', err);
    });
}

/*****************************************************************************
 * Function Name  : getHorasDeLuz
 * Description    : Obtiene las horas de luz diarias de la ultima semana.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getHorasDeLuz() {

  // Vector para actualizar el grafico
  var horasDeLuzSemanal = [0, 0, 0, 0, 0, 0, 0];
  var ahora = new Date();
  var diff = ahora.getDate() - ahora.getDay() + (ahora.getDay() === 0 ? -6 : 1);
  const lunes = firebase.firestore.Timestamp.fromDate(new Date(ahora.setDate(diff)));

  // Horas de luz semanales
  const horasDeLuzRef = db.collection('Modulos').doc('Sist01-Mod01').collection('Hitos')
    .where('tipo', '==', 'ILUMINACION')
    .where('momento', '>=', lunes)
    .orderBy('momento');

  horasDeLuzRef
    .get()
    .then(function (horasDeLuzData) {

      var i = 0;
      horasDeLuzData.forEach(function (doc) {

        horasDeLuzSemanal[i] = doc.data().descripcion;
        i++;
      });

      // Actualizamos la vista
      myBarChartHorasDeLuz.data.datasets[0].data = horasDeLuzSemanal;
      myBarChartHorasDeLuz.update();
```

```
    })
    .catch(function (err) {

      console.log('Error getting horas de luz: ', err);
    });
}

/*****************************************************************************
 * Function Name  : getHitos
 * Description    : Obtiene el listado de Hitos. Calcula el numero de hitos por
 *                  tipologia y actualiza el grafico.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getHitos() {

  var numAlerta;
  var numError;
  var numSistema;
  var numEstados;
  var numHorasLuz;

  // Hitos del año en curso.
  const anio = new Date().getFullYear().toString();
  const inicio = firebase.firestore.Timestamp.fromDate(new Date(anio));
  const hitosRef = db.collection('Modulos').doc('Sist01-Mod01').collection('Hitos')
    .where('momento', '>=', inicio)
    .orderBy('momento');

  hitosRef
    .get()
    .then(function (hitosData) {

      numAlerta = 0;
      numError = 0;
      numSistema = 0;
      numEstados = 0;
      numHorasLuz = 0;

      hitosData.forEach(function (doc) {

        if (doc.data().tipo == "ALERTA") { numAlerta += 1; }

        if (doc.data().tipo == "ERROR") { numError += 1; }

        if (doc.data().tipo == "SISTEMA") { numSistema += 1; }

        if (doc.data().tipo == "ESTADOS") { numEstados += 1; }

        if (doc.data().tipo == "ILUMINACION") { numHorasLuz += 1; }

      });

      // Actualizamos la vista
      var dataSetHitos = [numAlerta, numError, numSistema, numEstados, numHorasLuz];
      myRadarChartHitos.data.datasets[0].data = dataSetHitos;
      myRadarChartHitos.update();
    })
    .catch(function (err) {

      console.log('Error getting hitos', err);
    });
}

/*****************************************************************************
 * Function Name  : getEstadosAmbiente
 * Description    : Obtiene el listado de estados de la ultima semana. Actualiza
 *                  el grafico de estados del ambiente. Representa humedad y
 *                  temperatura.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getEstadosAmbiente() {

  var tempAmbSemanal = [0, 0, 0, 0, 0, 0, 0];
  var humAmbSemanal = [0, 0, 0, 0, 0, 0, 0];
```

```
  var tempSemanalAcumulado = [0, 0, 0, 0, 0, 0, 0];
  var humSemanalAcumulado = [0, 0, 0, 0, 0, 0, 0];
  var itemsTemp = [0, 0, 0, 0, 0, 0, 0];
  var itemsHum = [0, 0, 0, 0, 0, 0, 0];

  // Obtenemos los estados de la ultima semana
  var ahora = new Date();
  var diff = ahora.getDate() - ahora.getDay() + (ahora.getDay() === 0 ? -6 : 1);
  const lunes = firebase.firestore.Timestamp.fromDate(new Date(ahora.setDate(diff)));

  const estadosAmbienteRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Estados')
    .where('momento', '>=', lunes)
    .orderBy('momento');

  estadosAmbienteRef
    .get()
    .then(function (estadosAmbienteData) {

      var i;
      estadosAmbienteData.forEach(function (doc) {

        // Recopilamos los datos de temperatura y humedad semanales
        i = doc.data().momento.toDate().getDay() - 1;
        tempSemanalAcumulado[i] += doc.data().tempAire;
        itemsTemp[i] += 1;
        humSemanalAcumulado[i] += doc.data().humedadAire;
        itemsHum[i] += 1;

      });

      //Calculamos los valores medios
      var x;
      for (x = 0; x < 7; x++) {

        // Nos aseguramos de no dividir por cero
        if (itemsTemp[x] > 0 && itemsHum[x] > 0) {

          tempAmbSemanal[x] = tempSemanalAcumulado[x] / itemsTemp[x];
          humAmbSemanal[x] = humSemanalAcumulado[x] / itemsHum[x];
        }
        else {

          tempAmbSemanal[x] = 0;
          humAmbSemanal[x] = 0;
        }

      }

      // Actualiza la vista
      myLineChartAmb.data.datasets[0].data = humAmbSemanal;
      myLineChartAmb.data.datasets[1].data = tempAmbSemanal;
      myLineChartAmb.update();
    })
    .catch(function (err) {

      console.log('Error getting estados ambiente: ', err);
    });
}

/**************************************************************************
 * Function Name  : getEstadosContenedor
 * Description    : Obtiene el listado de estados de la ultima semana. Actualiza
 *                  el grafico de estados del contenedor. Representa humedad
 *                  maxima y minima.
 * Parameters     :
 * Return         :
 **************************************************************************/
async function getEstadosContenedor() {

  var maxMoistSemanal = [0, 0, 0, 0, 0, 0, 0];
  var minMoistSemanal = [3000, 3000, 3000, 3000, 3000, 3000, 3000];

  // Obtenemos los estados de la ultma semana
  var ahora = new Date();
  var diff = ahora.getDate() - ahora.getDay() + (ahora.getDay() === 0 ? -6 : 1);
  const lunes = firebase.firestore.Timestamp.fromDate(new Date(ahora.setDate(diff)));
```

```
  const estadosContenedorRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Estados')
    .where('momento', '>=', lunes)
    .orderBy('momento');

  estadosContenedorRef
    .get()
    .then(function (estadosContenedorData) {

      var i, moistureActual;
      estadosContenedorData.forEach(function (doc) {

        i = doc.data().momento.toDate().getDay() - 1;
        moistureActual = doc.data().moisture;

        // Actualizamos los vectores de maximos y minimos
        if (moistureActual > maxMoistSemanal[i]) { maxMoistSemanal[i] = moistureActual;
}
        if (moistureActual < minMoistSemanal[i]) { minMoistSemanal[i] = moistureActual;
}

      });

      // Actualizamos la vista
      myLineChartCont.data.datasets[0].data = maxMoistSemanal;
      myLineChartCont.data.datasets[1].data = minMoistSemanal;
      myLineChartCont.update();
    })
    .catch(function (err) {

      alert('Error getting estados contenedor: ', err);
    });
}

/*****************************************************************************
 * Function Name  : getListadoTareas
 * Description    : Obtiene el listado de tareas. Actualiza la tabla con la
 *                  representacion grafica.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getListadoTareas() {

  // Tabla de tareas
  tableTareas = $('#dtTablaTareas').DataTable({

    //"data" : dataSetTareas,
    "columns": [
      { "data": "select" },
      { "data": "id" },
      { "data": "Prioridad" },
      { "data": "Descripción" }
    ],
    "columnDefs": [
      { "visible": false, "targets": 1 },
      { "orderable": false, "className": 'select-checkbox', "targets": 0 }
    ],
    "select": { "style": 'single', "selector": 'td:first-child' },
    "pagingType": "full_numbers",
    "language": {
      "sProcessing": "Procesando...",
      "sLengthMenu": "Mostrar _MENU_ registros",
      "sZeroRecords": "No se encontraron resultados",
      "sEmptyTable": "Ningún dato disponible en esta tabla",
      "sInfo": "Del _START_ al _END_ de un total de _TOTAL_",
      "sInfoEmpty": "0 de un total de 0 registros",
      "sInfoFiltered": "(filtrado de un total de _MAX_ registros)",
      "sInfoPostFix": "",
      "sSearch": "Buscar:",
      "sUrl": "",
      "sInfoThousands": ",",
      "sLoadingRecords": "Cargando...",
      "oAria": {
        "sSortAscending": ": Activar para ordenar la columna de manera ascendente",
        "sSortDescending": ": Activar para ordenar la columna de manera descendente"
```

```
      }

    }

  });
  var laPrioridad;

  // Iniciamos la referencia a la base de datos.
  const tareasRef = db.collection('Tareas').orderBy('criticidad', 'desc');

  tareasRef
    .get()
    .then(function (lasTareas) {

      lasTareas.forEach(function (tarea) {

        switch (tarea.data().criticidad) {

          case "0":
            laPrioridad = "Baja";
            break;

          case "1":
            laPrioridad = "Media";
            break;

          case "2":
            laPrioridad = "Alta";
            break;

          case "3":
            laPrioridad = "Urgente";
            break;
        }

        tableTareas.row.add({

          select: "",
          id: tarea.id,
          Prioridad: laPrioridad,
          Descripción: tarea.data().descripcion
        });
      });
      tableTareas.draw();
    })
    .catch(function (err) {

      alert('Error obteniendo el listado de tareas', err);
    });
}

/*****************************************************************************
 * Function Name  : getListadoPlantas
 * Description    : Obtiene el listado de plantas. Actualiza la tabla con la
 *                  representacion grafica.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getListadoPlantas() {

  // Tabla de plantas
  tablePlantas = $('#dtTablaPlantas').DataTable({

    //"data" : dataSetPlantas,
    "columns": [
      { "data": "select" },
      { "data": "id" },
      { "data": "descripcion" },
      { "data": "fechaInicioSiembra" },
      { "data": "fechaFinSiembra" },
      { "data": "horasDeLuz" },
      { "data": "humMax" },
      { "data": "humMin" },
      { "data": "moistMax" },
      { "data": "moistMin" },
      { "data": "tempMax" },
      { "data": "tempMin" }
```

```
    ],
    "columnDefs": [
      { "visible": false, "targets": 1 },
      { "orderable": false, "className": 'select-checkbox', "targets": 0 }
    ],
    "select": { "style": 'single', "selector": 'td:first-child' },
    "pagingType": "full_numbers",
    "language": {
      "sProcessing": "Procesando...",
      "sLengthMenu": "Mostrar _MENU_ registros",
      "sZeroRecords": "No se encontraron resultados",
      "sEmptyTable": "Ningún dato disponible en esta tabla",
      "sInfo": "Del _START_ al _END_ de un total de _TOTAL_",
      "sInfoEmpty": "0 de un total de 0 registros",
      "sInfoFiltered": "(filtrado de un total de _MAX_ registros)",
      "sInfoPostFix": "",
      "sSearch": "Buscar:",
      "sUrl": "",
      "sInfoThousands": ",",
      "sLoadingRecords": "Cargando...",
      "oAria": {
        "sSortAscending": ": Activar para ordenar la columna de manera ascendente",
        "sSortDescending": ": Activar para ordenar la columna de manera descendente"
      }

    }

  });

  // Iniciamos la referencia a la base de datos.
  const plantasRef = db.collection('Plantas').orderBy('descripcion', 'desc');

  plantasRef
    .get()
    .then(function (lasPlantas) {

      lasPlantas.forEach(function (planta) {

        tablePlantas.row.add({

          select: "",
          id: planta.id,
          descripcion: planta.data().descripcion,
          fechaInicioSiembra: planta.data().fechaInicioSiembra,
          fechaFinSiembra: planta.data().fechaFinSiembra,
          horasDeLuz: planta.data().horasDeLuz,
          humMax: planta.data().humMax,
          humMin: planta.data().humMin,
          moistMax: planta.data().moistMax,
          moistMin: planta.data().moistMin,
          tempMax: planta.data().tempMax,
          tempMin: planta.data().tempMin
        });
      });
      tablePlantas.draw();
    })
    .catch(function (err) {

      alert('Error obteniendo el listado de plantas', err);
    });
}

/*****************************************************************************
 * Function Name  : getListadoRiegos
 * Description    : Obtiene el listado de riegos. Actualiza la tabla con la
 *                  representacion grafica.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getListadoRiegos() {

  // Tabla de riegos.
  var tableRiegos = $('#dtTablaRiegos').DataTable({

    "columns": [
      { "data": "encharcado" },
```

```
      { "data": "horaDeRiego" },
      { "data": "moistureFinal" },
      { "data": "moistureInicial" },
      { "data": "momento" },
      { "data": "tiempoDeRiego" }
    ],
    "pagingType": "full_numbers",
    "language": {
      "sProcessing": "Procesando...",
      "sLengthMenu": "Mostrar _MENU_ registros",
      "sZeroRecords": "No se encontraron resultados",
      "sEmptyTable": "Ningún dato disponible en esta tabla",
      "sInfo": "Del _START_ al _END_ de un total de _TOTAL_",
      "sInfoEmpty": "0 de un total de 0 registros",
      "sInfoFiltered": "(filtrado de un total de _MAX_ registros)",
      "sInfoPostFix": "",
      "sSearch": "Buscar:",
      "sUrl": "",
      "sInfoThousands": ",",
      "sLoadingRecords": "Cargando...",
      "oAria": {
        "sSortAscending": ": Activar para ordenar la columna de manera ascendente",
        "sSortDescending": ": Activar para ordenar la columna de manera descendente"
      }

    }

  });

  // Iniciamos la referencia a la base de datos.
  const riegosRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Riegos').orderBy('momento', 'desc');

  riegosRef
    .get()
    .then(function (riegosData) {

      riegosData.forEach(function (doc) {

        tableRiegos.row.add({

          encharcado: ((doc.data().encharcado) ? "Si" : "No"),
          horaDeRiego: doc.data().horaDeRiego,
          moistureFinal: doc.data().moistureFinal,
          moistureInicial: doc.data().moistureInicial,
          momento: doc.data().momento.toDate().toLocaleDateString(),
          tiempoDeRiego: doc.data().tiempoDeRiego
        });
      });
      tableRiegos.draw();
    })
    .catch(function (err) {

      alert('Error obteniendo el listado de riegos', err);
    });
}

/*****************************************************************************
 * Function Name  : getListadoHitos
 * Description    : Obtiene el listado de hitos. Actualiza la tabla con la
 *                  representacion grafica.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getListadoHitos() {

  // Tabla de hitos.
  var tableHitos = $('#dtTablaHitos').DataTable({

    "columns": [
      { "data": "tipo" },
      { "data": "descripcion" },
      { "data": "momento" }
    ],
    "pagingType": "full_numbers",
    "language": {
      "sProcessing": "Procesando...",
```

```
      "sLengthMenu": "Mostrar _MENU_ registros",
      "sZeroRecords": "No se encontraron resultados",
      "sEmptyTable": "Ningún dato disponible en esta tabla",
      "sInfo": "Del  START  al  END  de un total de _TOTAL_",
      "sInfoEmpty": "0 de un total de 0 registros",
      "sInfoFiltered": "(filtrado de un total de _MAX_ registros)",
      "sInfoPostFix": "",
      "sSearch": "Buscar:",
      "sUrl": "",
      "sInfoThousands": ",",
      "sLoadingRecords": "Cargando...",
      "oAria": {
        "sSortAscending": ": Activar para ordenar la columna de manera ascendente",
        "sSortDescending": ": Activar para ordenar la columna de manera descendente"
      }

    }

  });

  // Iniciamos la referencia a la base de datos.
  const hitosRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Hitos').orderBy('momento', 'desc');

  hitosRef
    .get()
    .then(function (hitosData) {

      hitosData.forEach(function (doc) {

        tableHitos.row.add({

          tipo: doc.data().tipo,
          descripcion: doc.data().descripcion,
          momento: doc.data().momento.toDate().toLocaleDateString()
        });
      });
      tableHitos.draw();
    })
    .catch(function (err) {

      alert('Error obteniendo el listado de hitos', err);
    });
}

/*****************************************************************************
 * Function Name  : getListadoEstados
 * Description    : Obtiene el listado de estados. Actualiza la tabla con la
 *                  representacion grafica.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getListadoEstados() {

  // Tabla de hitos.
  var tableEstados = $('#dtTablaEstados').DataTable({

    "columns": [
      { "data": "luzVisible" },
      { "data": "luzUV" },
      { "data": "humedad" },
      { "data": "temperatura" },
      { "data": "moisture" },
      { "data": "momento" }
    ],
    "pagingType": "full_numbers",
    "language": {
      "sProcessing": "Procesando...",
      "sLengthMenu": "Mostrar _MENU_ registros",
      "sZeroRecords": "No se encontraron resultados",
      "sEmptyTable": "Ningún dato disponible en esta tabla",
      "sInfo": "Del _START_ al _END_ de un total de _TOTAL_",
      "sInfoEmpty": "0 de un total de 0 registros",
      "sInfoFiltered": "(filtrado de un total de _MAX_ registros)",
      "sInfoPostFix": "",
      "sSearch": "Buscar:",
```

```
      "sUrl": "",
      "sInfoThousands": ",",
      "sLoadingRecords": "Cargando...",
      "oAria": {
        "sSortAscending": ": Activar para ordenar la columna de manera ascendente",
        "sSortDescending": ": Activar para ordenar la columna de manera descendente"
      }

    }

  });

  // Iniciamos la referencia a la base de datos. Lo limitamos a 500 estados
  const estadosRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Estados').orderBy('momento', 'desc').limit(500);

  estadosRef
    .get()
    .then(function (estadosData) {

      estadosData.forEach(function (doc) {

        tableEstados.row.add({

          luzVisible: doc.data().luzVisible,
          luzUV: doc.data().UVIndex,
          humedad: doc.data().humedadAire,
          temperatura: doc.data().tempAire,
          moisture: doc.data().moisture,
          momento: doc.data().momento.toDate()
        });
      });
      tableEstados.draw();
    })
    .catch(function (err) {

      alert('Error obteniendo el listado de estados', err);
    });
}

/*****************************************************************************
 * Function Name  : getListadoSiembras
 * Description    : Obtiene el listado de siembras. Actualiza la tabla con la
 *                  representacion grafica.
 * Parameters     :
 * Return         :
 *****************************************************************************/
async function getListadoSiembras() {

  // Tabla de siembras
  tableSiembras = $('#dtTablaSiembras').DataTable({

    "columns": [
      { "data": "select" },
      { "data": "id" },
      { "data": "idPlanta" },
      { "data": "m2" },
      { "data": "momento" }
    ],
    "columnDefs": [
      { "visible": false, "targets": 1 },
      { "orderable": false, "className": 'select-checkbox', "targets": 0 }
    ],
    "select": { "style": 'single', "selector": 'td:first-child' },
    "pagingType": "full_numbers",
    "language": {
      "sProcessing": "Procesando...",
      "sLengthMenu": "Mostrar _MENU_ registros",
      "sZeroRecords": "No se encontraron resultados",
      "sEmptyTable": "Ningún dato disponible en esta tabla",
      "sInfo": "Del _START_ al _END_ de un total de _TOTAL_",
      "sInfoEmpty": "0 de un total de 0 registros",
      "sInfoFiltered": "(filtrado de un total de _MAX_ registros)",
      "sInfoPostFix": "",
      "sSearch": "Buscar:",
      "sUrl": "",
      "sInfoThousands": ",",
```

```
      "sLoadingRecords": "Cargando...",
      "oAria": {
        "sSortAscending": ": Activar para ordenar la columna de manera ascendente",
        "sSortDescending": ": Activar para ordenar la columna de manera descendente"
      }

    }

  });

  // Todas las siembras del modulo
  const siembrasRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Siembras').orderBy('momento', 'desc');

  // Tenemos que obtener la descripcion de las plantas de la tabla de plantas
  var laPlanta;

  siembrasRef
    .get()
    .then(function (lasSiembras) {

      lasSiembras.forEach(function (siembra) {

        // Vamos a obtener la descripcion de la planta
        laPlanta = db.collection('Plantas').doc(siembra.data().idPlanta);
        laPlanta.get()
          .then(function (planta) {

            if (planta.exists) {

              console.log("Buscando planta", planta.data().descripcion);
              tableSiembras.row.add({

                select: "",
                id: siembra.id,
                idPlanta: planta.data().descripcion,
                m2: siembra.data().m2,
                momento: siembra.data().momento.toDate().toLocaleDateString()
              });
              tableSiembras.draw();
            }
            else {

              console.log("No se ha encontrado la descripción de la planta.
GetListadoSiembras");
            }

          });
      });
    })
    .catch(function (err) {

      console.log('Error getting listado de siembras', err);
    });
}

/**************************************************************************
* Function Name  : getListadoRecolecciones
* Description    : Obtiene el listado de recolecciones. Actualiza la tabla con
*                  la representacion grafica.
* Parameters     :
* Return         :
**************************************************************************/
async function getListadoRecolecciones() {

  // Tabla de recolecciones
  tableRecolecciones = $('#dtTablaRecolecciones').DataTable({

    "columns": [
      { "data": "id" },
      { "data": "idPlanta" },
      { "data": "Kg" },
      { "data": "momento" }
    ],
    "columnDefs": [
      { "visible": false, "targets": 0 }
```

```
    ],
    "select": { "style": 'single', "selector": 'td:first-child' },
    "pagingType": "full_numbers",
    "language": {
      "sProcessing": "Procesando...",
      "sLengthMenu": "Mostrar _MENU_ registros",
      "sZeroRecords": "No se encontraron resultados",
      "sEmptyTable": "Ningún dato disponible en esta tabla",
      "sInfo": "Del _START_ al _END_ de un total de _TOTAL_",
      "sInfoEmpty": "0 de un total de 0 registros",
      "sInfoFiltered": "(filtrado de un total de _MAX_ registros)",
      "sInfoPostFix": "",
      "sSearch": "Buscar:",
      "sUrl": "",
      "sInfoThousands": ",",
      "sLoadingRecords": "Cargando...",
      "oAria": {
        "sSortAscending": ": Activar para ordenar la columna de manera ascendente",
        "sSortDescending": ": Activar para ordenar la columna de manera descendente"
      }

    }

  });

  // Todas las recolecciones del modulo
  const recoleccionesRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Recolecciones').orderBy('momento', 'desc');

  recoleccionesRef
    .get()
    .then(function (lasRecolecciones) {

      lasRecolecciones.forEach(function (recoleccion) {

        tableRecolecciones.row.add({

          id: recoleccion.id,
          idPlanta: recoleccion.data().idPlanta,
          Kg: recoleccion.data().Kg,
          momento: recoleccion.data().momento.toDate().toLocaleDateString()
        });
        tableRecolecciones.draw();
      });
    })
    .catch(function (err) {

      console.log('Error getting listado de recolecciones', err);
    });
}

/*************************************************************************
 * Function Name  : guardarModulo
 * Description    : Guarda los valores del modulo en firebase y en el firmware.
 * Parameters     :
 * Return         :
 *************************************************************************/
function guardarModulo() {

  var datosModuloParticle = {
    "tipoSuelo": 2,
    "caudal": 1,
    "emisores": 1,
    "superficie": 0.6
  };
  var datosModuloFirebase = {
    "Ne": 1,
    "m2": 0.6,
    "q": 1,
    "tipoSuelo": 2
  };

  // Obtenemos los datos del modal.
  datosModuloParticle.tipoSuelo = datosModuloFirebase.tipoSuelo =
$('#inputTipoSuelo').val();
  datosModuloParticle.caudal = datosModuloFirebase.q = $('#inputCaudal').val();
  datosModuloParticle.emisores = datosModuloFirebase.Ne = $('#inputEmisores').val();
```

```
  datosModuloParticle.superficie = datosModuloFirebase.m2 = $('#inputSuperficie').val();

  // Iniciamos la operacion
  alert("Va a iniciarse una operación asíncrona. Espere el resultado.");
  guardarModuloEnParticle(JSON.stringify(datosModuloParticle))
    .then(function (data) {

      if (data == 0) {

        // Se ha actualizado el firmware. Actualizamos la base de datos
        alert("Los datos de configuración van a ser guardados.");
        return guardarModuloEnFirebase(datosModuloFirebase);
      }
      else {

        // El firmware NO ha sido actualizado
        alert('Los datos no han sido actualizados');
      }
    })
    .then(function (data) {

      // Actualizamos la vista
      getModulo();
      $('#modalDeConfiguracion').modal('hide');
    })
    .catch(function (err) {

      // Se ha producido un error en la cadena de acciones asincronas
      alert('Error guardando los datos de configuración.', err);
    });

}

/**************************************************************************
 * Function Name  : guardarModuloEnFirebase
 * Description    : Guarda los valores del modulo en firebase.
 * Parameters     : datosModulo. Objeto con los datos del modulo.
 * Return         : promise.
 **************************************************************************/
async function guardarModuloEnFirebase(datosModulo) {

  // Iniciamos la referencia a la base de datos.
  const moduloRef = db.collection('Modulos').doc('Sist01-Mod01');

  return moduloRef
    .update(datosModulo)
    .then(function () {

      console.log("Modulo actualizado en Firebase.");
    });

}

/**************************************************************************
 * Function Name  : guardarTareaEnFirebase
 * Description    : Guarda los valores de la tarea en firebase.
 * Parameters     : datosTarea. Objeto con los datos de la tarea.
 * Return         : ID nueva tarea.
 **************************************************************************/
async function guardarTareaEnFirebase(datosTarea) {

  // Iniciamos la referencia a la base de datos.
  const tareasRef = db.collection('Tareas');

  return tareasRef
    .add(datosTarea)
    .then(function (tareaRef) {

      // Devuelve el ID de la tarea creada.
      return tareaRef.id;
    });

}

/**************************************************************************
 * Function Name  : borrarTareaEnFirebase
```

```
 * Description    : Borra la tarea de firebase.
 * Parameters     : idTarea. El ID de la tarea a eliminar.
 * Return         : promise.
 ******************************************************************************/
async function borrarTareaEnFirebase(idTarea) {

  // Iniciamos la referencia a la base de datos.
  const tareaRef = db.collection('Tareas').doc(idTarea);

  return tareaRef
    .delete()
    .then(function () {

      console.log("Tarea suprimida.");
    });

}

/******************************************************************************
 * Function Name  : guardarPlantaEnFirebase
 * Description    : Guarda una nueva planta en firebase.
 * Parameters     : datosPlanta. Objeto con los datos de la planta.
 * Return         : ID nueva planta.
 ******************************************************************************/
async function guardarPlantaEnFirebase(datosPlanta) {

  // Iniciamos la referencia a la base de datos.
  const plantasRef = db.collection('Plantas');

  return plantasRef
    .add(datosPlanta)
    .then(function (plantaRef) {

      // Devuelve el ID de la planta creada.
      return plantaRef.id;
    });

}

/******************************************************************************
 * Function Name  : borrarPlantaEnFirebase
 * Description    : Borra la planta de firebase.
 * Parameters     : idPlanta. El ID de la planta a eliminar.
 * Return         : promise.
 ******************************************************************************/
async function borrarPlantaEnFirebase(idPlanta) {

  // Iniciamos la referencia a la base de datos.
  const plantaRef = db.collection('Plantas').doc(idPlanta);

  return plantaRef
    .delete()
    .then(function () {

      console.log("Planta suprimida.");
    });

}

/******************************************************************************
 * Function Name  : sembrarPlantaEnFirebase
 * Description    : Añade los datos de una nueva siembra en firebase.
 * Parameters     : datosSiembra. Objeto con los datos de la siembra.
 * Return         : ID nueva siembra.
 ******************************************************************************/
async function sembrarPlantaEnFirebase(datosSiembra) {

  // Iniciamos la referencia a la base de datos.
  const siembrasRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Siembras');

  return siembrasRef
    .add(datosSiembra)
    .then(function (siembra) {

      // Devuelve el ID de la planta creada.
      return siembra.id;
```

```
    });
}

/*****************************************************************************
 * Function Name  : recolectarPlantaEnFirebase
 * Description    : Añade los datos de una nueva recoleccion en firebase.
 * Parameters     : datosRecoleccion. Objeto con los datos de la recoleccion.
 * Return         : ID nueva recoleccion.
 *****************************************************************************/
async function recolectarPlantaEnFirebase(datosRecoleccion) {

  // Iniciamos la referencia a la base de datos.
  const recoleccionesRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Recolecciones');

  return recoleccionesRef
    .add(datosRecoleccion)
    .then(function (recoleccion) {

      // Devuelve el ID de la planta creada.
      return recoleccion.id;
    });
}

/*****************************************************************************
 * Function Name  : borrarSiembraEnFirebase
 * Description    : Borra la siembra de firebase.
 * Parameters     : idSiembra. El ID de la siembra a eliminar.
 * Return         : promise.
 *****************************************************************************/
async function borrarSiembraEnFirebase(idSiembra) {

  // Iniciamos la referencia a la base de datos.
  const siembraRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Siembras').doc(idSiembra);

  return siembraRef
    .delete()
    .then(function () {

      console.log("Siembra suprimida.");
    });

}

/*****************************************************************************
 * Function Name  : getM2Libre
 * Description    : Calcula la superficie libre del modulo de plantacion.
 * Parameters     :
 * Return         : m2 libres.
 *****************************************************************************/
async function getM2Libre() {

  var m2Totales;
  var libre;

  // Iniciamos la referencia a la base de datos.
  const moduloRef = db.collection('Modulos').doc('Sist01-Mod01');

  return moduloRef
    .get()
    .then(function (modData) {

      // Devuelve los m2 totales del modulo.
      return modData.data().m2;
    })
    .then(function (m2Total) {

      m2Totales = parseFloat(m2Total);
      const siembrasRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Siembras');
      return siembrasRef
        .get()
        .then(function (siembrasData) {

          var m2Acumulado = 0;
```

```
          siembrasData.forEach(function (siembra) {

            // Calcula los m2 sembrados.
            m2Acumulado += parseFloat(siembra.data().m2);
          });
          console.log(m2Totales - m2Acumulado);

          // Devuelve los m2 libres.
          return (m2Totales - m2Acumulado);
        });

    });

}

/**************************************************************************
 * Function Name  : estaSembrada
 * Description    : Determina si laPlanta esta sembrada antes de eliminarla.
 * Parameters     : laPlanta. ID de la planta a consultar.
 * Return         : true - false.
 **************************************************************************/
async function estaSembrada(laPlanta) {

  // Todas las siembras del modulo
  const siembrasRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Siembras');
  var resultado = false;

  return siembrasRef
    .get()
    .then(function (siembrasData) {

      siembrasData.forEach(function (siembra) {

        //Comprobamos si esta sembrada
        if (siembra.data().idPlanta == laPlanta) {

          resultado = true;
        }

      });
      return resultado;
    })
    .catch(function (err) {

      console.log('Error determinando si esta sembrada.', err);
    });
}

/**************************************************************************
 * Function Name  : setUmbrales
 * Description    : Recalcula los umbrales en base a las plantas sembradas.
 *                  Actualiza el firmware.
 * Parameters     :
 * Return         : promise.
 **************************************************************************/
async function setUmbrales() {

  // Todas las siembras del modulo
  const siembrasRef = db.collection('Modulos').doc('Sist01-
Mod01').collection('Siembras');

  // Establecemos unos umbrales de base.
  var umbrales = {

    "temperatura":
    {
      "max": 35.0,
      "min": 5.0
    },
    "humedad":
    {
      "max": 50.0,
      "min": 10.0
    },
    "horasDeLuz":
    {
```

```
      "max": 20,
      "min": 2.5
    },
    "moisture":
    {
      "max": 2400.0,
      "min": 1000.0
    }

  };

  return siembrasRef
    .get()
    .then(function (lasSiembras) {

      lasSiembras.forEach(function (siembra) {

        // Vamos a obtener los umbrales de la planta
        laPlanta = db.collection('Plantas').doc(siembra.data().idPlanta);
        laPlanta.get()
          .then(function (planta) {

            if (planta.exists) {

              // Recalculamos los umbrales
              if (umbrales.temperatura.max < parseFloat(planta.data().tempMax)) {
umbrales.temperatura.max = parseFloat(planta.data().tempMax) }
              if (umbrales.temperatura.min > parseFloat(planta.data().tempMin)) {
umbrales.temperatura.min = parseFloat(planta.data().tempMin) }
              if (umbrales.humedad.max < parseFloat(planta.data().humMax)) {
umbrales.humedad.max = parseFloat(planta.data().humMax) }
              if (umbrales.humedad.min > parseFloat(planta.data().humMin)) {
umbrales.humedad.min = parseFloat(planta.data().humMin) }
              if (umbrales.moisture.max < parseFloat(planta.data().moistMax)) {
umbrales.moisture.max = parseFloat(planta.data().moistMax) }
              if (umbrales.moisture.min > parseFloat(planta.data().moistMin)) {
umbrales.moisture.min = parseFloat(planta.data().moistMin) }
              if (umbrales.horasDeLuz.min < parseFloat(planta.data().horasDeLuz)) {
umbrales.horasDeLuz.min = parseFloat(planta.data().horasDeLuz) }
            }
            else {

              console.log("No se han podido ajustar umbrales en el firmware.");
            }

          });
      });
    })
    .then(function () {

      return setUmbralesEnParticle(JSON.stringify(umbrales));
      //return umbrales;
    })
    .then(function (resultado) {

      return resultado;
    })
    .catch(function (err) {

      console.log('Error getting listado de siembras', err);
    });
}

/*******************************************************************************
 *
 *            Funciones de validacion de datos de entrada.
 *            Validan la entrada de usuario en los formularios modales.
 *
 *******************************************************************************/

// Valida cantidades numericas enteras
function validarEntero(valor, minimo, maximo) {

  var validar = true;

  // Comprobamos que no este vacio
```

```
  if (valor == null || valor.length == 0 || /^\s+$/.test(valor)) {

    validar = false;
  }
  // Comprobamos la estructura de la cadena
  if (!(/^\d{1,4}$/.test(valor))) {

    validar = false;
  }
  // Comprobamos que el valor este en el intervalo
  if ((valor < minimo) || (valor > maximo) || isNaN(valor)) {

    validar = false;
  }
  return validar;
}

// Valida cantidades numericas decimales
function validarDecimal(valor, minimo, maximo) {

  var validar = true;

  // Comprobamos que no este vacio
  if (valor == null || valor.length == 0 || /^\s+$/.test(valor)) {

    validar = false;
  }
  // Comprobamos la estructura de la cadena
  if (!(/^\d+\.\d*$/.test(valor))) {

    validar = false;
  }
  // Comprobamos que el valor este en el intervalo
  if ((valor < minimo) || (valor > maximo) || isNaN(valor)) {

    validar = false;
  }
  return validar;
}

// Valida los datos del formulario de configuracion  del modulo
function validarModulo() {

  var validar = true;
  var caudal, emisores, superficie;

  caudal = $('#inputCaudal').val();
  emisores = $('#inputEmisores').val();
  superficie = $('#inputSuperficie').val();

  validar = validarDecimal(caudal, 1, 100) && validarDecimal(superficie, 0, 10) &&
validarEntero(emisores, 1, 100);

  return validar;
}

// Valida los datos del formulario de creacion de planta
function validarPlanta(laPlanta) {

  var validar = true;

  validar = validarEntero(laPlanta.fechaInicioSiembra, 1, 12)
    && validarEntero(laPlanta.fechaFinSiembra, 1, 12)
    && (laPlanta.fechaFinSiembra > laPlanta.fechaInicioSiembra)
    && validarEntero(laPlanta.humMax, 1, 100)
    && validarEntero(laPlanta.humMin, 1, 100)
    && (laPlanta.humMax > laPlanta.humMin)
    && validarEntero(laPlanta.moistMax, 1, 3000)
    && validarEntero(laPlanta.moistMin, 1, 3000)
    && (laPlanta.moistMax > laPlanta.moistMin)
    && validarEntero(laPlanta.tempMax, 1, 100)
    && validarEntero(laPlanta.tempMin, 1, 100)
    && (laPlanta.tempMax > laPlanta.tempMin);

  return validar;
}
```

```
// Valida los datos del formulario de creacion de una nueva siembra
function validarSiembra(superficie) {

  var validar = true;

  validar = validarDecimal(superficie, 0, 10);

  return validar;
}

// Valida los datos del formulario de creacion de una nueva recoleccion
function validarRecoleccion(cantidad) {

  var validar = true;

  validar = validarDecimal(cantidad, 0, 100);

  return validar;
}
```

Archivo: huerto-particle.js

```
var miPHOTON = "PHOTONID";
var token = "ACCESSTOKEN";

// Instanciamos el objeto Particle para acceder al API
particle = new Particle();

/*****************************************************************************
 * Function Name  : obtenerVariable
 * Description    : Obtiene el valor de una variable del firmware
 * Parameters     : laVariable. Variable a descargar.
 * Return         : Valor Ok, Err Fallo.
 *****************************************************************************/
function obtenerVariable(laVariable) {

  return particle.getVariable({ deviceId: miPHOTON, name: laVariable, auth: token })
    .then(function (data) {

      console.log(laVariable + ': ', data);
      return data.body.result;

    })
    .catch(function (err) {

      console.log('Error al descargar variable: ' + laVariable + ': ', err);
      return "Err";
    });

}

/*****************************************************************************
 * Function Name  : guardarModuloEnParticle
 * Description    : Actualiza los datos del modulo
 *                  y resetea la configuracion de riego.
 * Parameters     : elModulo. Configuracion del modulo.
 * Return         : 0 Ok, -1 Fallo.
 *****************************************************************************/
function guardarModuloEnParticle(elModulo) {

  return particle.callFunction({ deviceId: miPHOTON, name: "Set_Modulo", argument:
elModulo, auth: token })
    .then(function (data) {

      return data.body.return_value;

    });

}

/*****************************************************************************
 * Function Name  : setUmbralesEnParticle
 * Description    : Actualiza los umbrales de funcionamiento
 *                  y resetea la configuracion de riego.
 * Parameters     : losUmbrales. Nuevos umbrales de funcionamiento.
```

```
 * Return         : 0 Ok, -1 Fallo.
 ********************************************************************************/
function setUmbralesEnParticle(losUmbrales) {

  return particle.callFunction({ deviceId: miPHOTON, name: "Set_Umbrales", argument:
losUmbrales, auth: token })
    .then(function (data) {

      return data.body.return_value;

    });

}
```

Archivo: style.css

```
 body {
    background-color: rgb(130, 201, 206);
    display: none;
}

.side-nav .logo-sn {
    padding-bottom: 1rem;
    padding-top: 1rem;
}

.side-nav .logo-sn img {
    height: 38px;
}

.side-nav .search-form input[type=text] {
    margin-top: 0;
    padding-top: 12px;
    padding-bottom: 12px;
    border-top: 1px solid rgba(255, 255, 255, 0.3);
    border-bottom: 1px solid rgba(255, 255, 255, 0.3);
}

body {
    background-color: #eee;
}

.accordion .card {
    margin-bottom: 1.2rem;
    box-shadow: 0 2px 5px 0 rgba(0, 0, 0, 0.16), 0 2px 10px 0 rgba(0, 0, 0, 0.12);
}

.accordion .card .card-body {
    border-top: 1px solid #eee;
}
```

Archivo: estados.html

```
<!DOCTYPE html>
<html lang="en">

<head>

  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta http-equiv="x-ua-compatible" content="ie=edge">

  <title>App Control. Estados.</title>

  <!-- Font Awesome -->
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css">
  <!-- Bootstrap core CSS -->
  <link href="assets/css/bootstrap.min.css" rel="stylesheet">
  <!-- Material Design Bootstrap -->
  <link href="assets/css/mdb.min.css" rel="stylesheet">
  <!-- SB Admin styles -->
  <link href="assets/css/sb-admin-2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables  -->
```

```
  <link href="assets/vendor/DataTables/datatables2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables Select  -->
  <link href="assets/vendor/DataTables/datatables-select2.min.css" rel="stylesheet">
  <!-- Huerto tool styles -->
  <link href="assets/css/style.css" rel="stylesheet">

</head>

<body id="panelPrincipal" class="fixed-sn light-blue-skin">

  <!--Main Navigation-->
  <header>
    <!--NavBar-->
    <nav class="navbar fixed-top navbar-expand-md navbar-dark default-color scrolling-
navbar">
      <a class="navbar-brand" href="#"><strong><i class="fas fa-carrot fa-2x ml-2
orange-text"></i></strong></a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
        aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
          <li class="nav-item">
            <a class="nav-link" href="index.html">Modulo</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="tareas.html">Tareas</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="plantas.html">Plantas</a>
          </li>
        </ul>
        <ul class="navbar-nav ml-auto nav-flex-icons">
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" href="sistema.html">
              <i class="fas fa-microchip"></i>
              <span class="clearfix d-none d-sm-inline-block">Sistema</span>
            </a>
          </li>
          <li class="nav-item">
            <a id="botonAlertas" class="nav-link waves-effect waves-light" data-
toggle="modal"
              data-target="#modalShowAlertas">
              <i class="far fa-bell"></i>
              <span class="clearfix d-none d-sm-inline-block">Alertas</span>
            </a>
          </li>
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" data-toggle="modal" data-
target="#modalConfirmSignOut">
              <i class="fas fa-sign-out-alt fa-border"></i>
            </a>
          </li>
        </ul>
      </div>
    </nav>
    <!--/.NavBar-->
  </header>
  <!--/ .Main Navigation-->
  <!--Main layout-->
  <main>
    <div class="container-fluid" style="margin-top:100px;">

      <!--Section: Modals-->
      <section>
        <!-- Modal. Alertas -->
        <div class="modal fade top" id="modalShowAlertas" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloAlertas">Alertas</h5>
```

```
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <!-- List group links -->
                <div class="list-group">
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Horas
                    de luz
                    <span id="valHorasDeLuz" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Humedad
                    <span id="valHumedad" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Moisture
                    <span id="valMoisture" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Temperatura
                    <span id="valTemperatura" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                </div>
                <!-- List group links -->
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-sm btn-info" data-
dismiss="modal">Aceptar</button>
              </div>
            </div>
          </div>
        </div>
        <!--/ .Modal. Alertas -->
        <!--Modal: modalSignOut-->
        <div class="modal fade top" id="modalConfirmSignOut" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <!--Content-->
            <div class="modal-content text-center">
              <!--Header-->
              <div class="modal-header d-flex justify-content-center">
                <p class="heading">Salir de la app?</p>
              </div>

              <!--Body-->
              <div class="modal-body">
                <i class="fas fa-bell fa-2x animated rotateIn"></i>
              </div>

              <!--Footer-->
              <div class="modal-footer justify-content-center">
                <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                <button type="button" id="botonSignOut" class="btn btn-sm btn-
primary">Salir</button>
              </div>
            </div>
            <!--/.Content-->
          </div>
        </div>
        <!--/ Modal: modalSignOut-->
      </section>
      <!--/ Section: Modals-->

      <!--Section: Table-->
      <section class="mb-5">
```

```
        <!-- Encabezado. Tabla. -->
        <div class="d-sm-flex align-items-center justify-content-between mb-4">
          <h1 class="h3 mb-0 text-gray-800">Listado de estados</h1>
        </div>
        <!--/ .Encabezado. Tabla. -->
        <!--Tabla de riegos-->
        <div class="table-responsive">
          <table id="dtTablaEstados" class="table table-hover table-bordered"
width="100%">
            <thead class="default-color white-text">
              <tr>
                <th class="th-sm"> - Luz. Visible</th>
                <th class="th-sm"> - Indice UV</th>
                <th class="th-sm"> - Humedad</th>
                <th class="th-sm"> - Temperatura</th>
                <th class="th-sm"> - Moisture</th>
                <th class="th-sm"> - Momento</th>
              </tr>
            </thead>
            <tbody>
            </tbody>
            <tfoot class="default-color white-text">
              <tr>
                <th class="th-sm"> - Luz. Visible</th>
                <th class="th-sm"> - Indice UV</th>
                <th class="th-sm"> - Humedad</th>
                <th class="th-sm"> - Temperatura</th>
                <th class="th-sm"> - Moisture</th>
                <th class="th-sm"> - Momento</th>
              </tr>
            </tfoot>
          </table>
        </div>
        <!--/ Tabla de riegos-->
      </section>
      <!--Section: Table-->

    </div>
  </main>
  <!--Main layout-->
  <!-- Footer -->
  <footer class="page-footer font-small default-color">

    <!-- Copyright -->
    <div class="footer-copyright text-center py-3">© 2019 Copyright:
      <a href="https://huertotool.firebaseapp.com/"> My Little Orchard</a>
    </div>
    <!-- Copyright -->

  </footer>
  <!-- Footer -->

</body>

<!-- SCRIPTS -->
<!-- JQuery -->
<script type="text/javascript" src="assets/js/jquery-3.3.1.min.js"></script>
<!-- Bootstrap tooltips -->
<script type="text/javascript" src="assets/js/popper.min.js"></script>
<!-- Bootstrap core JavaScript -->
<script type="text/javascript" src="assets/js/bootstrap.min.js"></script>
<!-- MDB core JavaScript -->
<script type="text/javascript" src="assets/js/mdb.js"></script>
<!-- JQUERY Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/jquery.dataTables.min.js"></script>
<!-- MDBootstrap Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/datatables2.min.js"></script>
<!-- DataTables Select JS -->
<script type="text/javascript" src="assets/vendor/DataTables/datatables-
select2.min.js"></script>
<!-- Load Firebase and Firestore -->
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-auth.js"></script>
```

```
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-firestore.js"></script>
<!-- Nuestra app firebase -->
<script type="text/javascript" src="./assets/js/huerto-fb.js"></script>

<script>

  window.addEventListener('load', function () {

    initApp();

  });

  $(document).ready(function () {

    // Modal Sign-out
    $('#botonSignOut').on('click', function () {

      clickSignoutButton();
    });

    // Modal de alertas
    $('#botonAlertas').on('click', function () {

      getAlertas();
    });

    // Actualizamos la vista
    getListadoEstados();

  });

</script>

</html>
```

Archivo: hitos.html

```
<!DOCTYPE html>
<html lang="en">

<head>

  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta http-equiv="x-ua-compatible" content="ie=edge">

  <title>App Control. Hitos.</title>

  <!-- Font Awesome -->
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css">
  <!-- Bootstrap core CSS -->
  <link href="assets/css/bootstrap.min.css" rel="stylesheet">
  <!-- Material Design Bootstrap -->
  <link href="assets/css/mdb.min.css" rel="stylesheet">
  <!-- SB Admin styles -->
  <link href="assets/css/sb-admin-2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables  -->
  <link href="assets/vendor/DataTables/datatables2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables Select  -->
  <link href="assets/vendor/DataTables/datatables-select2.min.css" rel="stylesheet">
  <!-- Huerto tool styles -->
  <link href="assets/css/style.css" rel="stylesheet">

</head>

<body id="panelPrincipal" class="fixed-sn light-blue-skin">

  <!--Main Navigation-->
  <header>
    <!--NavBar-->
    <nav class="navbar fixed-top navbar-expand-md navbar-dark default-color scrolling-
navbar">
```

```
      <a class="navbar-brand" href="#"><strong><i class="fas fa-carrot fa-2x ml-2
orange-text"></i></strong></a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
        aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
          <li class="nav-item">
            <a class="nav-link" href="index.html">Modulo</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="tareas.html">Tareas</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="plantas.html">Plantas</a>
          </li>
        </ul>
        <ul class="navbar-nav ml-auto nav-flex-icons">
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" href="sistema.html">
              <i class="fas fa-microchip"></i>
              <span class="clearfix d-none d-sm-inline-block">Sistema</span>
            </a>
          </li>
          <li class="nav-item">
            <a id="botonAlertas" class="nav-link waves-effect waves-light" data-
toggle="modal"
              data-target="#modalShowAlertas">
              <i class="far fa-bell"></i>
              <span class="clearfix d-none d-sm-inline-block">Alertas</span>
            </a>
          </li>
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" data-toggle="modal" data-
target="#modalConfirmSignOut">
              <i class="fas fa-sign-out-alt fa-border"></i>
            </a>
          </li>
        </ul>
      </div>
    </nav>
    <!--/.NavBar-->
  </header>
  <!--/ .Main Navigation-->
  <!--Main layout-->
  <main>
    <div class="container-fluid" style="margin-top:100px;">

      <!--Section: Modals-->
      <section>
        <!-- Modal. Alertas -->
        <div class="modal fade top" id="modalShowAlertas" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloAlertas">Alertas</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <!-- List group links -->
                <div class="list-group">
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Horas
                    de luz
                    <span id="valHorasDeLuz" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
```

```
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Humedad
                    <span id="valHumedad" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Moisture
                    <span id="valMoisture" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Temperatura
                    <span id="valTemperatura" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                </div>
                <!-- List group links -->
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-sm btn-info" data-
dismiss="modal">Aceptar</button>
              </div>
            </div>
          </div>
        </div>
        <!--/ .Modal. Alertas -->
        <!--Modal: modalSignOut-->
        <div class="modal fade top" id="modalConfirmSignOut" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <!--Content-->
            <div class="modal-content text-center">
              <!--Header-->
              <div class="modal-header d-flex justify-content-center">
                <p class="heading">Salir de la app?</p>
              </div>

              <!--Body-->
              <div class="modal-body">
                <i class="fas fa-bell fa-2x animated rotateIn"></i>
              </div>

              <!--Footer-->
              <div class="modal-footer justify-content-center">
                <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                <button type="button" id="botonSignOut" class="btn btn-sm btn-
primary">Salir</button>
              </div>
            </div>
            <!--/.Content-->
          </div>
        </div>
        <!--/ Modal: modalSignOut-->
      </section>
      <!--/ Section: Modals-->

      <!--Section: Table-->
      <section class="mb-5">
        <!-- Encabezado. Tabla. -->
        <div class="d-sm-flex align-items-center justify-content-between mb-4">
          <h1 class="h3 mb-0 text-gray-800">Listado de hitos</h1>
        </div>
        <!--/ .Encabezado. Tabla. -->
        <!--Tabla de riegos-->
        <div class="table-responsive">
          <table id="dtTablaHitos" class="table table-hover table-bordered"
width="100%">
            <thead class="default-color white-text">
              <tr>
                <th class="th-sm"> - Tipo</th>
                <th class="th-sm"> - Descripción</th>
                <th class="th-sm"> - Momento</th>
              </tr>
            </thead>
```

```
            <tbody>
            </tbody>
            <tfoot class="default-color white-text">
              <tr>
                <th class="th-sm"> - Tipo</th>
                <th class="th-sm"> - Descripción</th>
                <th class="th-sm"> - Momento</th>
              </tr>
            </tfoot>
          </table>
        </div>
        <!--/ Tabla de riegos-->
      </section>
      <!--Section: Table-->

    </div>
  </main>
  <!--Main layout-->
  <!-- Footer -->
  <footer class="page-footer font-small default-color">

    <!-- Copyright -->
    <div class="footer-copyright text-center py-3">© 2019 Copyright:
      <a href="https://huertotool.firebaseapp.com/"> My Little Orchard</a>
    </div>
    <!-- Copyright -->

  </footer>
  <!-- Footer -->

</body>

<!-- SCRIPTS -->
<!-- JQuery -->
<script type="text/javascript" src="assets/js/jquery-3.3.1.min.js"></script>
<!-- Bootstrap tooltips -->
<script type="text/javascript" src="assets/js/popper.min.js"></script>
<!-- Bootstrap core JavaScript -->
<script type="text/javascript" src="assets/js/bootstrap.min.js"></script>
<!-- MDB core JavaScript -->
<script type="text/javascript" src="assets/js/mdb.js"></script>
<!-- JQUERY Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/jquery.dataTables.min.js"></script>
<!-- MDBootstrap Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/datatables2.min.js"></script>
<!-- DataTables Select JS -->
<script type="text/javascript" src="assets/vendor/DataTables/datatables-
select2.min.js"></script>
<!-- Load Firebase and Firestore -->
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-auth.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-firestore.js"></script>
<!-- Nuestra app firebase -->
<script type="text/javascript" src="./assets/js/huerto-fb.js"></script>

<script>

  window.addEventListener('load', function () {

    initApp();

  });

  $(document).ready(function () {

    // Modal Sign-out
    $('#botonSignOut').on('click', function () {

      clickSignoutButton();
    });

    // Modal de alertas
    $('#botonAlertas').on('click', function () {
```

```
      getAlertas();
    });

    // Actualizamos la vista
    getListadoHitos();

  });

</script>

</html>
```

Archivo: index.html

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta http-equiv="x-ua-compatible" content="ie=edge">

  <title>App Control. Huerto.</title>

  <!-- Font Awesome -->
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css">
  <!-- Bootstrap core CSS -->
  <link href="assets/css/bootstrap.min.css" rel="stylesheet">
  <!-- Material Design Bootstrap -->
  <link href="assets/css/mdb.min.css" rel="stylesheet">
  <!-- SB Admin styles -->
  <link href="assets/css/sb-admin-2.min.css" rel="stylesheet">
  <!-- Huerto tool styles -->
  <link href="assets/css/style.css" rel="stylesheet">

</head>

<body id="panelPrincipal" class="fixed-sn light-blue-skin">

  <!--Barra de menu-->
  <header>
    <!--NavBar-->
    <nav class="navbar fixed-top navbar-expand-md navbar-dark default-color scrolling-
navbar">
      <a class="navbar-brand" href="#"><strong><i class="fas fa-carrot fa-2x ml-2
orange-text"></i></strong></a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
        aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
          <li class="nav-item active">
            <a class="nav-link" href="index.html">Modulo</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="tareas.html">Tareas</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="plantas.html">Plantas</a>
          </li>
        </ul>
        <ul class="navbar-nav ml-auto nav-flex-icons">
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" href="sistema.html">
              <i class="fas fa-microchip"></i>
              <span class="clearfix d-none d-sm-inline-block">Sistema</span>
            </a>
          </li>
          <li class="nav-item">
            <a id="botonAlertas" class="nav-link waves-effect waves-light" data-
toggle="modal"
```

```
              data-target="#modalShowAlertas">
              <i class="far fa-bell"></i>
              <span class="clearfix d-none d-sm-inline-block">Alertas</span>
            </a>
          </li>
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" data-toggle="modal" data-
target="#modalConfirmSignOut">
              <i class="fas fa-sign-out-alt fa-border"></i>
            </a>
          </li>
        </ul>
      </div>
    </nav>
    <!--/.NavBar-->
  </header>
  <!--/ .Barra de menu-->

  <!--Main layout-->
  <main>

    <div class="container-fluid" style="margin-top:100px;">

      <!--Section: Modals-->
      <section>
        <!-- Modal. Configuración -->
        <div class="modal fade top" id="modalDeConfiguracion" tabindex="-1"
role="dialog" aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloConfiguracion">Configuración</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <form class="border border-light">

                  <label for="inputTipoSuelo">Tipo de suelo</label>
                  <select class="browser-default custom-select mb-1"
id="inputTipoSuelo">
                    <option value="0">Gravoso</option>
                    <option value="1">Arenoso</option>
                    <option value="2" selected>Franco</option>
                    <option value="3">Arcilloso</option>
                  </select>

                  <label for="inputSuperficie">Superficie</label>
                  <input type="text" maxlength="5" id="inputSuperficie" class="form-
control mb-1"
                    placeholder="M2 - Decimal [0 - 10]">

                  <label for="inputEmisores">Número de emisores</label>
                  <input type="text" maxlength="3" id="inputEmisores" class="form-
control mb-1"
                    placeholder="Entero [1 - 100]">

                  <label for="inputCaudal">Caudal</label>
                  <input type="text" maxlength="6" id="inputCaudal" class="form-control
mb-1"
                    placeholder="Litros/min - Decimal [1 - 100]">

                </form>
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                <button type="button" class="btn btn-sm btn-primary"
id="guardarModulo">Guardar</button>
              </div>
            </div>
          </div>
        </div>
        <!--/ .Modal. Configuración -->
```

```
        <!-- Modal. Alertas -->
        <div class="modal fade top" id="modalShowAlertas" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloAlertas">Alertas</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <!-- List group links -->
                <div class="list-group">
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Horas
                    de luz
                    <span id="valHorasDeLuz" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Humedad
                    <span id="valHumedad" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Moisture
                    <span id="valMoisture" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Temperatura
                    <span id="valTemperatura" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                </div>
                <!-- List group links -->
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-sm btn-info" data-
dismiss="modal">Aceptar</button>
              </div>
            </div>
          </div>
        </div>
        <!--/ .Modal. Alertas -->
        <!--Modal: modalSignOut-->
        <div class="modal fade top" id="modalConfirmSignOut" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <!--Content-->
            <div class="modal-content text-center">
              <!--Header-->
              <div class="modal-header d-flex justify-content-center">
                <p class="heading">Salir de la app?</p>
              </div>

              <!--Body-->
              <div class="modal-body">
                <i class="fas fa-bell fa-2x animated rotateIn"></i>
              </div>

              <!--Footer-->
              <div class="modal-footer justify-content-center">
                <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                <button type="button" id="botonSignOut" class="btn btn-sm btn-
primary">Salir</button>
              </div>
            </div>
            <!--/.Content-->
          </div>
```

```
        </div>
        <!--/ Modal: modalSignOut-->
      </section>
      <!--Section: Modals-->

      <!--Section: Main panel-->
      <section class="mb-5">
        <!-- Encabezado. Configuración. -->
        <div class="d-sm-flex align-items-center justify-content-between mb-4">
          <h1 class="h3 mb-0 text-gray-800">Configuración</h1>
          <button type="button" class="btn btn-primary btn-sm waves-effect waves-light"
data-toggle="modal"
            data-target="#modalDeConfiguracion">Editar</button>
        </div>
        <!--/ .Encabezado. Configuración. -->
        <!-- Panel de configuración.  -->
        <div class="row">

          <!-- Tipo de suelo -->
          <div class="col-xl-3 col-md-6 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Tipo de suelo</div>
                    <div id="valTipoSuelo" class="h5 mb-0 font-weight-bold text-gray-
800">Franco</div>
                  </div>
                  <div class="col-auto">
                    <i class="fas fa-seedling fa-2x text-gray-600"></i>
                  </div>
                </div>
              </div>
            </div>
          </div>

          <!-- Superficie -->
          <div class="col-xl-3 col-md-6 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Superficie</div>
                    <div id="valM2" class="h5 mb-0 font-weight-bold text-gray-800">0.5
M2.</div>
                  </div>
                  <div class="col-auto">
                    <i class="fas fa-expand fa-2x text-gray-600"></i>
                  </div>
                </div>
              </div>
            </div>
          </div>

          <!-- Número de emisores -->
          <div class="col-xl-3 col-md-6 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Número de emisores</div>
                    <div id="valNe" class="h5 mb-0 font-weight-bold text-gray-800">10
Uds.</div>
                  </div>
                  <div class="col-auto">
                    <i class="fas fa-fill-drip fa-2x text-gray-600"></i>
                  </div>
                </div>
              </div>
            </div>
          </div>

          <!-- Caudal -->
```

```
          <div class="col-xl-3 col-md-6 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Caudal</div>
                    <div class="row no-gutters align-items-center">
                      <div class="col-auto">
                        <div id="valCaudal" class="h5 mb-0 mr-3 font-weight-bold text-
gray-800">10 litros/hora</div>
                      </div>
                      <div class="col">
                        <div class="progress progress-sm mr-2">
                          <div class="progress-bar bg-info" role="progressbar"
style="width: 50%" aria-valuenow="50"
                               aria-valuemin="0" aria-valuemax="100"></div>
                        </div>
                      </div>
                    </div>
                  </div>
                  <div class="col-auto">
                    <i class="fas  fa-tint fa-2x text-gray-600"></i>
                  </div>
                </div>
              </div>
            </div>
          </div>

        </div>
        <!--/ .Panel de configuración. -->
        <!-- Encabezado. Riegos. -->
        <div class="d-sm-flex align-items-center justify-content-between mb-4">
          <h1 class="h3 mb-0 text-gray-800">Riegos</h1>
          <a id="elBotonRiegos" href="./riegos.html" target="_blank"
            class="d-sm-inline-block btn btn-sm btn-primary shadow-sm">
            <i class="fas fa-download fa-sm text-white-50"></i> Detalle</a>
        </div>
        <!--/ .Encabezado. Riegos. -->
        <!-- Panel de riegos. -->
        <div class="row">
          <!-- Gráfico diario -->
          <div class="col-xl-4 col-lg-5">
            <div class="card shadow mb-4">
              <!-- Card Header -->
              <div class="card-header py-3 d-flex flex-row align-items-center justify-
content-between">
                <h6 class="m-0 font-weight-bold text-primary">Diario</h6>
              </div>
              <!-- Card Body -->
              <div class="card-body">
                <div class="chart-pie pt-4 pb-2">
                  <div class="chartjs-size-monitor">
                    <div class="chartjs-size-monitor-expand">
                      <div class=""></div>
                    </div>
                    <div class="chartjs-size-monitor-shrink">
                      <div class=""></div>
                    </div>
                  </div>
                  <canvas id="myPieChartRiegos" width="486" height="245" class="chartjs-
render-monitor"
                    style="display: block; width: 486px; height: 245px;"></canvas>
                </div>
                <div class="mt-4 text-center small">
                  <span class="mr-2">
                    <i class="fas fa-circle text-info"></i> Regando
                  </span>
                </div>
              </div>
            </div>
          </div>
          <!-- ./Gráfico diario -->
          <!-- Gráfico semanal -->
          <div class="col-xl-8 col-lg-7">
            <div class="card shadow mb-4">
              <!-- Card Header - Dropdown -->
```

```
              <div class="card-header py-3 d-flex flex-row align-items-center justify-
content-between">
                <h6 class="m-0 font-weight-bold text-primary">Semanal</h6>
              </div>
              <!-- Card Body -->
              <div class="card-body">
                <div class="chart-area">
                  <div class="chartjs-size-monitor">
                    <div class="chartjs-size-monitor-expand">
                      <div class=""></div>
                    </div>
                    <div class="chartjs-size-monitor-shrink">
                      <div class=""></div>
                    </div>
                  </div>
                  <canvas id="myAreaChartRiegos" style="display: block; width: 1037px;
height: 320px;" width="1037"
                    height="320" class="chartjs-render-monitor"></canvas>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Gráfico semanal -->
        </div>
        <!--/ .Panel de riegos. -->
        <!-- Panel de aprovechamiento. -->
        <div class="row">
          <!-- Gráfico de Recolecciones -->
          <div class="col-xl-8 col-lg-7">
            <!-- Encabezado. Recolecciones -->
            <div class="d-sm-flex align-items-center justify-content-between mb-4">
              <h1 class="h3 mb-0 text-gray-800">Recolecciones</h1>
              <a href="./recolecciones.html" target="_blank" class="d-sm-inline-block
btn btn-sm btn-primary shadow-sm">
                <i class="fas fa-download fa-sm text-white-50"></i> Detalle</a>
            </div>
            <!--/ .Encabezado. Recolecciones -->
            <!-- Panel de Recolecciones -->
            <div class="card shadow mb-4">
              <div class="card-header py-3 d-flex flex-row align-items-center justify-
content-between">
                <h6 class="m-0 font-weight-bold text-primary">Mensuales</h6>
              </div>
              <!-- Card Body -->
              <div class="card-body">
                <div class="chart-bar">
                  <div class="chartjs-size-monitor">
                    <div class="chartjs-size-monitor-expand">
                      <div class=""></div>
                    </div>
                    <div class="chartjs-size-monitor-shrink">
                      <div class=""></div>
                    </div>
                  </div>
                  <canvas id="myBarChartRecolecciones" width="1037" height="320"
class="chartjs-render-monitor"
                    style="display: block; width: 1037px; height: 320px;"></canvas>
                </div>
              </div>
            </div>
            <!--/ .Panel de Recolecciones -->
          </div>
          <!--/ .Gráfico de Recolecciones -->
          <!-- Gráfico de siembras -->
          <div class="col-xl-4 col-lg-5">
            <!-- Encabezado. Siembras -->
            <div class="d-sm-flex align-items-center justify-content-between mb-4">
              <h1 class="h3 mb-0 text-gray-800">Siembras</h1>
              <a href="./siembras.html" target="_blank" class="d-sm-inline-block btn
btn-sm btn-primary shadow-sm">
                <i class="fas fa-download fa-sm text-white-50"></i> Detalle</a>
            </div>
            <!--/ .Encabezado. Siembras -->
            <div class="card shadow mb-4">
              <!-- Card Header -->
```

```
              <div class="card-header py-3 d-flex flex-row align-items-center justify-
content-between">
                <h6 class="m-0 font-weight-bold text-primary">Aprovechamiento</h6>
              </div>
              <!-- Card Body -->
              <div class="card-body">
                <div class="chart-pie pt-4 pb-2">
                  <div class="chartjs-size-monitor">
                    <div class="chartjs-size-monitor-expand">
                      <div class=""></div>
                    </div>
                    <div class="chartjs-size-monitor-shrink">
                      <div class=""></div>
                    </div>
                  </div>
                  <canvas id="myPieChartSiembras" width="486" height="245"
class="chartjs-render-monitor"
                    style="display: block; width: 486px; height: 245px;"></canvas>
                </div>
                <div class="mt-4 text-center small">
                  <span class="mr-2">
                    <i class="fas fa-circle text-success"></i> % Sembrado
                  </span>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Gráfico de siembras -->
        </div>
        <!--/ .Panel de hitos. -->
        <!-- Encabezado. Luz e hitos. -->
        <div class="d-sm-flex align-items-center justify-content-between mb-4">
          <h1 class="h3 mb-0 text-gray-800">Hitos de sistema</h1>
          <a id="elBotonHitos" href="./hitos.html" target="_blank"
            class="d-sm-inline-block btn btn-sm btn-primary shadow-sm">
            <i class="fas fa-download fa-sm text-white-50"></i> Detalle</a>
        </div>
        <!--/ .Encabezado. Luz e hitos. -->
        <!-- Luz e hitos -->
        <div class="row">
          <!-- Gráfico de horas de luz -->
          <div class="col-lg-6">
            <div class="card shadow mb-4">
              <!-- Card Header - Dropdown -->
              <div class="card-header py-3 d-flex flex-row align-items-center justify-
content-between">
                <h6 class="m-0 font-weight-bold text-primary">Horas de luz</h6>
              </div>
              <!-- Card Body -->
              <div class="card-body">
                <div class="chart-area">
                  <div class="chartjs-size-monitor">
                    <div class="chartjs-size-monitor-expand">
                      <div class=""></div>
                    </div>
                    <div class="chartjs-size-monitor-shrink">
                      <div class=""></div>
                    </div>
                  </div>
                  <canvas id="myBarChartHorasDeLuz" style="display: block; width:
1037px; height: 320px;" width="1037"
                    height="320" class="chartjs-render-monitor"></canvas>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Gráfico de horas de luz -->
          <!-- Gráfico de hitos -->
          <div class="col-lg-6">
            <!--  Panel de hitos -->
            <div class="card shadow mb-4">
              <!-- Card Header -->
              <div class="card-header py-3 d-flex flex-row align-items-center justify-
content-between">
                <h6 class="m-0 font-weight-bold text-primary">Funcionamiento</h6>
              </div>
              <!-- Card Body -->
```

```
              <div class="card-body">
                <div class="chart-area">
                  <div class="chartjs-size-monitor">
                    <div class="chartjs-size-monitor-expand">
                      <div class=""></div>
                    </div>
                    <div class="chartjs-size-monitor-shrink">
                      <div class=""></div>
                    </div>
                  </div>
                  <canvas id="radarChartFuncionamiento" style="display: block; width:
1037px; height: 320px;"
                    width="1037" height="320" class="chartjs-render-monitor"></canvas>
                </div>
              </div>
            </div>
            <!--/ .Panel de hitos -->
          </div>
          <!--/ .Gráfico de hitos -->
        </div>
        <!--/ .Luz e hitos -->
        <!--/ .Panel de hitos-->
        <!-- Ambiente y contenedor -->
        <!-- Encabezado. Ambiente y contenedor. -->
        <div class="d-sm-flex align-items-center justify-content-between mb-4">
          <h1 class="h3 mb-0 text-gray-800">Sensores de estado</h1>
          <a id="elBotonEstados" href="./estados.html" target="_blank"
            class="d-sm-inline-block btn btn-sm btn-primary shadow-sm">
            <i class="fas fa-download fa-sm text-white-50"></i> Detalle</a>
        </div>
        <!--/ .Encabezado. Ambiente y contenedor. -->
        <div class="row">
          <!-- Gráfico de ambiente -->
          <div class="col-lg-6">
            <!-- Gráfico. Humedad y temperatura ambiente -->
            <div class="card shadow mb-4">
              <div class="card-header py-3 d-flex flex-row align-items-center justify-
content-between">
                <h6 class="m-0 font-weight-bold text-primary">Ambiente</h6>
              </div>
              <div class="card-body">
                <div class="chart-area">
                  <div class="chartjs-size-monitor">
                    <div class="chartjs-size-monitor-expand">
                      <div class=""></div>
                    </div>
                    <div class="chartjs-size-monitor-shrink">
                      <div class=""></div>
                    </div>
                  </div>
                  <canvas id="myLineChartAmbiente" style="display: block; width: 1037px;
height: 320px;" width="1037"
                    height="320" class="chartjs-render-monitor"></canvas>
                </div>
              </div>
            </div>
            <!-- / .Gráfico. Humedad y temperatura ambiente -->
          </div>
          <!--/ .Gráfico de ambiente -->
          <!-- Gráfico de contenedor -->
          <div class="col-lg-6">
            <!--  Gráfico. Moisture y temperatura contenedor -->
            <div class="card shadow mb-4">
              <div class="card-header py-3 d-flex flex-row align-items-center justify-
content-between">
                <h6 class="m-0 font-weight-bold text-primary">Contenedor</h6>
              </div>
              <div class="card-body">
                <div class="chart-area">
                  <div class="chartjs-size-monitor">
                    <div class="chartjs-size-monitor-expand">
                      <div class=""></div>
                    </div>
                    <div class="chartjs-size-monitor-shrink">
                      <div class=""></div>
                    </div>
```

```
                </div>
                <canvas id="myLineChartContenedor" style="display: block; width:
1037px; height: 320px;" width="1037"
                  height="320" class="chartjs-render-monitor"></canvas>
              </div>
            </div>
          </div>
          <!--/ .Gráfico. Moisture y temperatura contenedor -->
        </div>
        <!--/ .Gráfico de contenedor -->
      </div>
      <!--/ .Anbiente y contenedor -->
    </section>
    <!--Section: Main panel-->

  </div>

</main>
<!--Main layout-->

<!-- Footer -->
<footer class="page-footer font-small default-color">

  <!-- Copyright -->
  <div class="footer-copyright text-center py-3">© 2019 Copyright:
    <a href="https://huertotool.firebaseapp.com/"> My Little Orchard</a>
  </div>
  <!-- Copyright -->

</footer>
<!-- Footer -->

</body>

<!-- SCRIPTS -->
<!-- JQuery -->
<script type="text/javascript" src="assets/js/jquery-3.3.1.min.js"></script>
<!-- Bootstrap tooltips -->
<script type="text/javascript" src="assets/js/popper.min.js"></script>
<!-- Bootstrap core JavaScript -->
<script type="text/javascript" src="assets/js/bootstrap.min.js"></script>
<!-- MDB core JavaScript -->
<script type="text/javascript" src="assets/js/mdb.js"></script>
<!-- Personalizaciones de los gráficos -->
<script type="text/javascript" src="assets/js/huerto-charts.js"></script>
<!-- Particle JS API -->
<script type="text/javascript" src="https://cdn.jsdelivr.net/npm/particle-api-
js@8/dist/particle.min.js"></script>
<!-- Particle HUERTO API -->
<script type="text/javascript" src="assets/js/huerto-particle.js"></script>
<!-- Load Firebase and Firestore -->
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-auth.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-firestore.js"></script>
<!-- Nuestra app firebase -->
<script type="text/javascript" src="./assets/js/huerto-fb.js"></script>

<script>

  window.addEventListener('load', function () {

    initApp();

  });

  $(document).ready(function () {

    // Inicializamos los tooltips
    $(function () {

      $('[data-toggle="tooltip"]').tooltip();
    });

    // Modal Sign-out
    $('#botonSignOut').on('click', function () {
```

```
      clickSignoutButton();
    });

    // Modal de alertas
    $('#botonAlertas').on('click', function () {

      getAlertas();
    });

    // Modal de configuracion del modulo
    $('#guardarModulo').on('click', function () {

      if (validarModulo()) {

        guardarModulo();
      }
      else {

        alert("Revise los datos del formulario.")
      }

    });

    // Actualizamos la vista.
    getModulo();
    getRiegos();
    getRecolecciones();
    getSiembras();
    getHorasDeLuz();
    getHitos();
    getEstadosAmbiente();
    getEstadosContenedor();
  });

</script>

</html>
```

Archivo: plantas.html

```
<!DOCTYPE html>
<html lang="en">

<head>

  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta http-equiv="x-ua-compatible" content="ie=edge">

  <title>App Control. Plantas.</title>

  <!-- Font Awesome -->
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css">
  <!-- Bootstrap core CSS -->
  <link href="assets/css/bootstrap.min.css" rel="stylesheet">
  <!-- Material Design Bootstrap -->
  <link href="assets/css/mdb.min.css" rel="stylesheet">
  <!-- SB Admin styles -->
  <link href="assets/css/sb-admin-2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables  -->
  <link href="assets/vendor/DataTables/datatables2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables Select  -->
  <link href="assets/vendor/DataTables/datatables-select2.min.css" rel="stylesheet">
  <!-- Huerto tool styles -->
  <link href="assets/css/style.css" rel="stylesheet">

</head>

<body id="panelPrincipal" class="fixed-sn light-blue-skin">

  <!--Barra de menu-->
  <header>
    <!--NavBar-->
    <nav class="navbar fixed-top navbar-expand-md navbar-dark default-color scrolling-
navbar">
```

```
      <a class="navbar-brand" href="#"><strong><i class="fas fa-carrot fa-2x ml-2
orange-text"></i></strong></a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
        aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
          <li class="nav-item">
            <a class="nav-link" href="index.html">Modulo</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="tareas.html">Tareas</a>
          </li>
          <li class="nav-item active">
            <a class="nav-link" href="plantas.html">Plantas</a>
          </li>
        </ul>
        <ul class="navbar-nav ml-auto nav-flex-icons">
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" href="sistema.html">
              <i class="fas fa-microchip"></i>
              <span class="clearfix d-none d-sm-inline-block">Sistema</span>
            </a>
          </li>
          <li class="nav-item">
            <a id="botonAlertas" class="nav-link waves-effect waves-light" data-
toggle="modal"
              data-target="#modalShowAlertas">
              <i class="far fa-bell"></i>
              <span class="clearfix d-none d-sm-inline-block">Alertas</span>
            </a>
          </li>
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" data-toggle="modal" data-
target="#modalConfirmSignOut">
              <i class="fas fa-sign-out-alt fa-border"></i>
            </a>
          </li>
        </ul>
      </div>
    </nav>
    <!--/.NavBar-->
  </header>
  <!--/ .Barra de menu-->

  <!--Main layout-->
  <main>
    <div class="container-fluid" style="margin-top:100px;">

      <!--Section: Modals-->
      <section>
        <!-- Modal. Alertas -->
        <div class="modal fade top" id="modalShowAlertas" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloAlertas">Alertas</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <!-- List group links -->
                <div class="list-group">
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Horas
                    de luz
                    <span id="valHorasDeLuz" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
```

```
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Humedad
                    <span id="valHumedad" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Moisture
                    <span id="valMoisture" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Temperatura
                    <span id="valTemperatura" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                </div>
                <!-- List group links -->
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-sm btn-info" data-
dismiss="modal">Aceptar</button>
              </div>
            </div>
          </div>
        </div>
        <!--/ .Modal. Alertas -->
        <!--Modal: modalSignOut-->
        <div class="modal fade top" id="modalConfirmSignOut" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <!--Content-->
            <div class="modal-content text-center">
              <!--Header-->
              <div class="modal-header d-flex justify-content-center">
                <p class="heading">Salir de la app?</p>
              </div>

              <!--Body-->
              <div class="modal-body">
                <i class="fas fa-bell fa-2x animated rotateIn"></i>
              </div>

              <!--Footer-->
              <div class="modal-footer justify-content-center">
                <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                <button type="button" id="botonSignOut" class="btn btn-sm btn-
primary">Salir</button>
              </div>
            </div>
            <!--/.Content-->
          </div>
        </div>
        <!--/ Modal: modalSignOut-->
        <!-- Modal. Nueva planta -->
        <div class="modal fade top" id="modalNuevaPlanta" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloNuevaPlanta">Nueva Planta</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <form class="border border-light">

                  <label for="inputDesPlanta">Descripción</label>
                  <input type="text" maxlength="50" id="inputDesPlanta" class="form-
control mb-1"
                    placeholder="Descripción">
                  <label for="inputInicioPlanta">Inicio</label>
```

```
                    <input type="text" maxlength="2" id="inputInicioPlanta" class="form-
control mb-1"
                      placeholder="Mes. 1 a 12." min="1" max="12">
                    <label for="inputFinPlanta">Fin</label>
                    <input type="text" maxlength="2" id="inputFinPlanta" class="form-
control mb-1"
                      placeholder="Mes. 1 a 12." min="1" max="12">
                    <label for="inputHorasLuz">Horas de luz</label>
                    <input type="text" maxlength="2" id="inputHorasLuz" class="form-
control mb-1" placeholder="1 a 20."
                      min="1" max="20">
                    <label for="inputHumMax">Humedad máxima</label>
                    <input type="text" maxlength="3" id="inputHumMax" class="form-control
mb-1"
                      placeholder="Entero  [1 - 100]">
                    <label for="inputHumMin">Humedad mínima</label>
                    <input type="text" maxlength="3" id="inputHumMin" class="form-control
mb-1"
                      placeholder="Entero  [1 - 100]">
                    <label for="inputMoistMax">Moisture máxima</label>
                    <input type="text" maxlength="4" id="inputMoistMax" class="form-
control mb-1"
                      placeholder="Entero  [1 - 3000]">
                    <label for="inputMoistMin">Moisture mínima</label>
                    <input type="text" maxlength="4" id="inputMoistMin" class="form-
control mb-1"
                      placeholder="Entero  [1 - 3000]">
                    <label for="inputTempMax">Temp máxima</label>
                    <input type="text" maxlength="3" id="inputTempMax" class="form-control
mb-1"
                      placeholder="Entero  [1 - 100]">
                    <label for="inputTempMin">Temp mínima</label>
                    <input type="text" maxlength="3" id="inputTempMin" class="form-control
mb-1"
                      placeholder="Entero  [1 - 100]">

                  </form>
                </div>
                <div class="modal-footer">
                  <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                  <button type="button" class="btn btn-sm btn-primary"
id="guardarPlanta">Guardar</button>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Modal. Nueva Planta -->
          <!-- Modal. Nueva siembra -->
          <div class="modal fade top" id="modalNuevaSiembra" tabindex="-1" role="dialog"
aria-hidden="true">
            <div class="modal-dialog modal-frame modal-top" role="document">
              <div class="modal-content">
                <div class="modal-header">
                  <h5 class="modal-title" id="tituloNuevaSiembra">Nueva siembra</h5>
                  <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                    <span aria-hidden="true">&times;</span>
                  </button>
                </div>
                <div class="modal-body">
                  <form class="border border-light">

                    <label for="inputM2Siembra">Superficie</label>
                    <input type="text" maxlength="5" id="inputM2Siembra" class="form-
control mb-1"
                      placeholder="M2. Decimal. [0 - 10]">

                  </form>
                </div>
                <div class="modal-footer">
                  <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                  <button type="button" class="btn btn-sm btn-primary"
id="guardarSiembra">Guardar</button>
                </div>
              </div>
```

```
          </div>
        </div>
        <!--/ .Modal. Nueva siembra -->
      </section>
      <!--/ Section: Modals-->

      <!--Section: Table-->
      <section class="mb-5">
        <!-- Encabezado. Tabla. -->
        <div class="d-sm-flex align-items-center justify-content-between mb-4">
          <h1 class="h3 mb-0 text-gray-800">Listado de Plantas</h1>
          <div class="d-sm-flex align-items-left justify-content-between">
            <button type="button" class="btn btn-success btn-sm waves-effect waves-
light"
              id="sembrarPlanta">Sembrar</button>
            <button type="button" class="btn btn-primary btn-sm waves-effect waves-
light" id="addPlanta"
              data-toggle="modal" data-target="#modalNuevaPlanta">Nueva</button>
            <button type="button" class="btn btn-danger btn-sm waves-effect waves-light"
              id="borrarPlanta">Eliminar</button>
          </div>
        </div>
        <!--/ .Encabezado. Tabla. -->
        <!--Tabla de Plantas-->
        <div class="table-responsive">
          <table id="dtTablaPlantas" class="table table-hover table-bordered"
width="100%">
            <thead class="default-color white-text">
              <tr>
                <th></th>
                <th>ID</th>
                <th class="th-sm"> - Descripción</th>
                <th class="th-sm"> - Ini. Siembra.</th>
                <th class="th-sm"> - Fin. Siembra.</th>
                <th class="th-sm"> - Horas. Luz.</th>
                <th class="th-sm"> - Hum. Máx.</th>
                <th class="th-sm"> - Hum. Min.</th>
                <th class="th-sm"> - Moist. Max.</th>
                <th class="th-sm"> - Moist. Min.</th>
                <th class="th-sm"> - Temp. Max.</th>
                <th class="th-sm"> - Temp. Min.</th>
              </tr>
            </thead>
            <tbody>
            </tbody>
            <tfoot class="default-color white-text">
              <tr>
                <th></th>
                <th>ID</th>
                <th class="th-sm"> - Descripción</th>
                <th class="th-sm"> - Ini. Siembra.</th>
                <th class="th-sm"> - Fin. Siembra.</th>
                <th class="th-sm"> - Horas. Luz.</th>
                <th class="th-sm"> - Hum. Máx.</th>
                <th class="th-sm"> - Hum. Min.</th>
                <th class="th-sm"> - Moist. Max.</th>
                <th class="th-sm"> - Moist. Min.</th>
                <th class="th-sm"> - Temp. Max.</th>
                <th class="th-sm"> - Temp. Min.</th>
              </tr>
            </tfoot>
          </table>
        </div>
        <!--/ Tabla de plantas-->
      </section>
      <!--Section: Table-->

    </div>
  </main>
  <!--Main layout-->

  <!-- Footer -->
  <footer class="page-footer font-small default-color">

    <!-- Copyright -->
    <div class="footer-copyright text-center py-3">© 2019 Copyright:
```

```
      <a href="https://huertotool.firebaseapp.com/"> My Little Orchard</a>
    </div>
    <!-- Copyright -->

  </footer>
  <!-- Footer -->

</body>

<!-- SCRIPTS -->
<!-- JQuery -->
<script type="text/javascript" src="assets/js/jquery-3.3.1.min.js"></script>
<!-- Bootstrap tooltips -->
<script type="text/javascript" src="assets/js/popper.min.js"></script>
<!-- Bootstrap core JavaScript -->
<script type="text/javascript" src="assets/js/bootstrap.min.js"></script>
<!-- MDB core JavaScript -->
<script type="text/javascript" src="assets/js/mdb.js"></script>
<!-- JQUERY Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/jquery.dataTables.min.js"></script>
<!-- MDBootstrap Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/datatables2.min.js"></script>
<!-- DataTables Select JS -->
<script type="text/javascript" src="assets/vendor/DataTables/datatables-
select2.min.js"></script>
<!-- Particle JS API -->
<script type="text/javascript" src="https://cdn.jsdelivr.net/npm/particle-api-
js@8/dist/particle.min.js"></script>
<!-- Particle HUERTO API -->
<script type="text/javascript" src="assets/js/huerto-particle.js"></script>
<!-- Load Firebase and Firestore -->
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-auth.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-firestore.js"></script>
<!-- Nuestra app firebase -->
<script type="text/javascript" src="./assets/js/huerto-fb.js"></script>

<script>

  var tablePlantas;

  window.addEventListener('load', function () {

    initApp();

  });

  $(document).ready(function () {

    // Modal Sign-out
    $('#botonSignOut').on('click', function () {

      clickSignoutButton();
    });

    // Modal de alertas
    $('#botonAlertas').on('click', function () {

      getAlertas();
    });

    // Modal de siembra
    $("#sembrarPlanta").on('click', function () {

      if (!tablePlantas.rows('.selected').any()) {

        alert("Seleccione una planta.");

      } else {

        $("#modalNuevaSiembra").modal('show');
      }

    });
```

```
// Guardar nueva planta
$('#guardarPlanta').on('click', function () {

  var datosPlanta = {

    "descripcion": "Descripción de la planta",
    "fechaInicioSiembra": "",
    "fechaFinSiembra": "",
    "horasDeLuz": "",
    "humMax": "",
    "humMin": "",
    "moistMax": "",
    "moistMin": "",
    "tempMax": "",
    "tempMin": ""
  };

  datosPlanta.descripcion = $('#inputDesPlanta').val();
  datosPlanta.fechaInicioSiembra = $('#inputInicioPlanta').val();
  datosPlanta.fechaFinSiembra = $('#inputFinPlanta').val();
  datosPlanta.horasDeLuz = $('#inputHorasLuz').val();
  datosPlanta.humMax = $('#inputHumMax').val();
  datosPlanta.humMin = $('#inputHumMin').val();
  datosPlanta.moistMax = $('#inputMoistMax').val();
  datosPlanta.moistMin = $('#inputMoistMin').val();
  datosPlanta.tempMax = $('#inputTempMax').val();
  datosPlanta.tempMin = $('#inputTempMin').val();

  if (validarPlanta(datosPlanta)) {

    // Debe añadir la planta a firebase
    guardarPlantaEnFirebase(datosPlanta).then(function (idNuevaPlanta) {

      // Actualizamos la tabla
      tablePlantas.row.add({

        select: "",
        id: idNuevaPlanta,
        descripcion: datosPlanta.descripcion,
        fechaInicioSiembra: datosPlanta.fechaInicioSiembra,
        fechaFinSiembra: datosPlanta.fechaFinSiembra,
        horasDeLuz: datosPlanta.horasDeLuz,
        humMax: datosPlanta.humMax,
        humMin: datosPlanta.humMin,
        moistMax: datosPlanta.moistMax,
        moistMin: datosPlanta.moistMin,
        tempMax: datosPlanta.tempMax,
        tempMin: datosPlanta.tempMin
      });

      // Actualizamos la vista.
      tablePlantas.draw();

      // Cerramos el formulario
      $('#modalNuevaPlanta').modal('hide');
    }).catch(function (err) {

      // Se ha producido un error en la cadena de acciones asincronas
      alert('Error al crear nueva planta.', err);
    });
  }
  else {

    alert("Revise los datos del formulario.")
  }

});

// Eliminar planta
$('#borrarPlanta').on('click', function () {

  if (!tablePlantas.rows('.selected').any()) {

    alert("Seleccione una planta.");
```

```
    } else {

      var row = tablePlantas.row('.selected');
      var data = row.data();

      // No eliminamos plantas sembradas
      estaSembrada(data.id).then(function (sembrada) {

        if (sembrada) {

          alert("Imposible elminar: está sembrada.")
        } else {

          borrarPlantaEnFirebase(data.id)
            .then(function () {

              // Eliminamos la fila del dataSet.
              row.remove();

              // Actualizamos la vista sin repaginar.
              tablePlantas.draw(false);

            })
        }

      })
        .catch(function (err) {

          // Se ha producido un error en la cadena de acciones asincronas
          alert('Error al borrar planta.', err);
        });

    }
  });

  // Sembrar planta
  $('#guardarSiembra').on('click', function () {

    var datosSiembra = {

      "idPlanta": "",
      "m2": "",
      "momento": ""
    };

    // Obtenemos los valores de siembra
    const momento = firebase.firestore.Timestamp.fromDate(new Date());
    var row = tablePlantas.row('.selected');
    var data = row.data();

    // Asignamos los valores al objeto
    datosSiembra.idPlanta = data.id;
    datosSiembra.m2 = $('#inputM2Siembra').val();
    datosSiembra.momento = momento;


    if (validarSiembra(datosSiembra.m2)) {

      getM2Libre()
        .then(function (m2libre) {

          console.log("Libre: ", m2libre);
          if (m2libre >= datosSiembra.m2) {

            alert("Va a realizarse la siembra");
            return sembrarPlantaEnFirebase(datosSiembra);
          }
          else {

            alert("Error. No hay espacio suficiente.");
          }

        })
        .then(function (idSiembra) {

          //Actualizamos la vista
```

```
            $('#modalNuevaSiembra').modal('hide');
            if (idSiembra != null) {

              //Aqui recalculamos los umbrales
              alert("Van a reajustarse los umbrales en el firmware");
              return setUmbrales();
            }

          })
          .then(function (losUmbrales) {

            if (losUmbrales == 0) {

              alert("Firmware actualizado de forma correcta.");
            }
            else {

              // Ha ocurrido algun error al actualizar el firmware
              alert("No ha sido posible actualizar el firmware.");
            }

          })
          .catch(function (err) {

            // Se ha producido un error en la cadena de acciones asincronas
            alert('Error al sembrar planta.', err);
          });
      }
      else {

        alert("Revise los datos del formulario.")
      }

    });

    // Actualizamos la vista
    getListadoPlantas();

  });

</script>

</html>
```

Archivo: recolecciones.html

```
<!DOCTYPE html>
<html lang="en">

<head>

  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta http-equiv="x-ua-compatible" content="ie=edge">

  <title>App Control. Recolecciones.</title>

  <!-- Font Awesome -->
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css">
  <!-- Bootstrap core CSS -->
  <link href="assets/css/bootstrap.min.css" rel="stylesheet">
  <!-- Material Design Bootstrap -->
  <link href="assets/css/mdb.min.css" rel="stylesheet">
  <!-- SB Admin styles -->
  <link href="assets/css/sb-admin-2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables  -->
  <link href="assets/vendor/DataTables/datatables2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables Select  -->
  <link href="assets/vendor/DataTables/datatables-select2.min.css" rel="stylesheet">
  <!-- Huerto tool styles -->
  <link href="assets/css/style.css" rel="stylesheet">
```

```
</head>

<body id="panelPrincipal" class="fixed-sn light-blue-skin">

  <!--Main Navigation-->
  <header>
    <!--NavBar-->
    <nav class="navbar fixed-top navbar-expand-md navbar-dark default-color scrolling-
navbar">
      <a class="navbar-brand" href="#"><strong><i class="fas fa-carrot fa-2x ml-2
orange-text"></i></strong></a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
        aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
          <li class="nav-item">
            <a class="nav-link" href="index.html">Modulo</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="tareas.html">Tareas</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="plantas.html">Plantas</a>
          </li>
        </ul>
        <ul class="navbar-nav ml-auto nav-flex-icons">
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" href="sistema.html">
              <i class="fas fa-microchip"></i>
              <span class="clearfix d-none d-sm-inline-block">Sistema</span>
            </a>
          </li>
          <li class="nav-item">
            <a id="botonAlertas" class="nav-link waves-effect waves-light" data-
toggle="modal"
              data-target="#modalShowAlertas">
              <i class="far fa-bell"></i>
              <span class="clearfix d-none d-sm-inline-block">Alertas</span>
            </a>
          </li>
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" data-toggle="modal" data-
target="#modalConfirmSignOut">
              <i class="fas fa-sign-out-alt fa-border"></i>
            </a>
          </li>
        </ul>
      </div>
    </nav>
    <!--/.NavBar-->
  </header>
  <!--/ .Main Navigation-->
  <!--Main layout-->
  <main>
    <div class="container-fluid" style="margin-top:100px;">

      <!--Section: Modals-->
      <section>
        <!-- Modal. Alertas -->
        <div class="modal fade top" id="modalShowAlertas" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloAlertas">Alertas</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <!-- List group links -->
                <div class="list-group">
```

```
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Horas
                    de luz
                    <span id="valHorasDeLuz" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Humedad
                    <span id="valHumedad" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Moisture
                    <span id="valMoisture" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Temperatura
                    <span id="valTemperatura" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                </div>
                <!-- List group links -->
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-sm btn-info" data-
dismiss="modal">Aceptar</button>
              </div>
            </div>
          </div>
        </div>
        <!--/ .Modal. Alertas -->
        <!--Modal: modalSignOut-->
        <div class="modal fade top" id="modalConfirmSignOut" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <!--Content-->
            <div class="modal-content text-center">
              <!--Header-->
              <div class="modal-header d-flex justify-content-center">
                <p class="heading">Salir de la app?</p>
              </div>

              <!--Body-->
              <div class="modal-body">
                <i class="fas fa-bell fa-2x animated rotateIn"></i>
              </div>

              <!--Footer-->
              <div class="modal-footer justify-content-center">
                <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                <button type="button" id="botonSignOut" class="btn btn-sm btn-
primary">Salir</button>
              </div>
            </div>
            <!--/.Content-->
          </div>
        </div>
        <!--/ Modal: modalSignOut-->
      </section>
      <!--/ Section: Modals-->

      <!--Section: Table-->
      <section class="mb-5">
        <!-- Encabezado. Tabla. -->
        <div class="d-sm-flex align-items-center justify-content-between mb-4">
          <h1 class="h3 mb-0 text-gray-800">Listado de Recolecciones</h1>
        </div>
        <!--/ .Encabezado. Tabla. -->
        <!--Tabla de Recolecciones-->
        <div class="table-responsive">
```

```
          <table id="dtTablaRecolecciones" class="table table-hover table-bordered"
width="100%">
            <thead class="default-color white-text">
              <tr>
                <th>ID</th>
                <th class="th-sm"> - Planta.</th>
                <th class="th-sm"> - Kg recolectados.</th>
                <th class="th-sm"> - Momento de recolección.</th>
              </tr>
            </thead>
            <tbody>
            </tbody>
            <tfoot class="default-color white-text">
              <tr>
                <th>ID</th>
                <th class="th-sm"> - Planta.</th>
                <th class="th-sm"> - Kg recolectados.</th>
                <th class="th-sm"> - Momento de recolección.</th>
              </tr>
            </tfoot>
          </table>
        </div>
        <!--/ Tabla de Recolecciones-->
      </section>
      <!--Section: Table-->

    </div>
  </main>
  <!--Main layout-->
  <!-- Footer -->
  <footer class="page-footer font-small default-color">

    <!-- Copyright -->
    <div class="footer-copyright text-center py-3">© 2019 Copyright:
      <a href="https://huertotool.firebaseapp.com/"> My Little Orchard</a>
    </div>
    <!-- Copyright -->

  </footer>
  <!-- Footer -->

</body>

<!-- SCRIPTS -->
<!-- JQuery -->
<script type="text/javascript" src="assets/js/jquery-3.3.1.min.js"></script>
<!-- Bootstrap tooltips -->
<script type="text/javascript" src="assets/js/popper.min.js"></script>
<!-- Bootstrap core JavaScript -->
<script type="text/javascript" src="assets/js/bootstrap.min.js"></script>
<!-- MDB core JavaScript -->
<script type="text/javascript" src="assets/js/mdb.js"></script>
<!-- JQUERY Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/jquery.dataTables.min.js"></script>
<!-- MDBootstrap Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/datatables2.min.js"></script>
<!-- DataTables Select JS -->
<script type="text/javascript" src="assets/vendor/DataTables/datatables-
select2.min.js"></script>
<!-- Load Firebase and Firestore -->
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-auth.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-firestore.js"></script>
<!-- Nuestra app firebase -->
<script type="text/javascript" src="./assets/js/huerto-fb.js"></script>

<script>

  window.addEventListener('load', function () {

    initApp();

  });
```

```
  $(document).ready(function () {

    // Modal Sign-out
    $('#botonSignOut').on('click', function () {

      clickSignoutButton();
    });

    // Modal de alertas
    $('#botonAlertas').on('click', function () {

      getAlertas();
    });

    // Actualizamos la vista
    getListadoRecolecciones();

  });

</script>

</html>
```

Archivo: riegos.html

```
<!DOCTYPE html>
<html lang="en">

<head>

  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta http-equiv="x-ua-compatible" content="ie=edge">

  <title>App Control. Riegos.</title>

  <!-- Font Awesome -->
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css">
  <!-- Bootstrap core CSS -->
  <link href="assets/css/bootstrap.min.css" rel="stylesheet">
  <!-- Material Design Bootstrap -->
  <link href="assets/css/mdb.min.css" rel="stylesheet">
  <!-- SB Admin styles -->
  <link href="assets/css/sb-admin-2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables  -->
  <link href="assets/vendor/DataTables/datatables2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables Select  -->
  <link href="assets/vendor/DataTables/datatables-select2.min.css" rel="stylesheet">
  <!-- Huerto tool styles -->
  <link href="assets/css/style.css" rel="stylesheet">

</head>

<body id="panelPrincipal" class="fixed-sn light-blue-skin">

  <!--Main Navigation-->
  <header>
    <!--NavBar-->
    <nav class="navbar fixed-top navbar-expand-md navbar-dark default-color scrolling-
navbar">
      <a class="navbar-brand" href="#"><strong><i class="fas fa-carrot fa-2x ml-2
orange-text"></i></strong></a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
        aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
          <li class="nav-item">
            <a class="nav-link" href="index.html">Modulo</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="tareas.html">Tareas</a>
```

```
          </li>
          <li class="nav-item">
            <a class="nav-link" href="plantas.html">Plantas</a>
          </li>
        </ul>
        <ul class="navbar-nav ml-auto nav-flex-icons">
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" href="sistema.html">
              <i class="fas fa-microchip"></i>
              <span class="clearfix d-none d-sm-inline-block">Sistema</span>
            </a>
          </li>
          <li class="nav-item">
            <a id="botonAlertas" class="nav-link waves-effect waves-light" data-
toggle="modal"
              data-target="#modalShowAlertas">
              <i class="far fa-bell"></i>
              <span class="clearfix d-none d-sm-inline-block">Alertas</span>
            </a>
          </li>
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" data-toggle="modal" data-
target="#modalConfirmSignOut">
              <i class="fas fa-sign-out-alt fa-border"></i>
            </a>
          </li>
        </ul>
      </div>
    </nav>
    <!--/.NavBar-->
  </header>
  <!--/ .Main Navigation-->
  <!--Main layout-->
  <main>
    <div class="container-fluid" style="margin-top:100px;">

      <!--Section: Modals-->
      <section>
        <!-- Modal. Alertas -->
        <div class="modal fade top" id="modalShowAlertas" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloAlertas">Alertas</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <!-- List group links -->
                <div class="list-group">
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Horas
                    de luz
                    <span id="valHorasDeLuz" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Humedad
                    <span id="valHumedad" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Moisture
                    <span id="valMoisture" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Temperatura
                    <span id="valTemperatura" class="badge badge-info pull-right ml-3">-
</span>
```

```
                </a>
              </div>
              <!-- List group links -->
            </div>
            <div class="modal-footer">
              <button type="button" class="btn btn-sm btn-info" data-
dismiss="modal">Aceptar</button>
            </div>
          </div>
        </div>
      </div>
      <!--/ .Modal. Alertas -->
      <!--Modal: modalSignOut-->
      <div class="modal fade top" id="modalConfirmSignOut" tabindex="-1" role="dialog"
aria-hidden="true">
        <div class="modal-dialog modal-frame modal-top" role="document">
          <!--Content-->
          <div class="modal-content text-center">
            <!--Header-->
            <div class="modal-header d-flex justify-content-center">
              <p class="heading">Salir de la app?</p>
            </div>

            <!--Body-->
            <div class="modal-body">
              <i class="fas fa-bell fa-2x animated rotateIn"></i>
            </div>

            <!--Footer-->
            <div class="modal-footer justify-content-center">
              <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
              <button type="button" id="botonSignOut" class="btn btn-sm btn-
primary">Salir</button>
            </div>
          </div>
          <!--/.Content-->
        </div>
      </div>
      <!--/ Modal: modalSignOut-->
    </section>
    <!--/ Section: Modals-->

    <!--Section: Table-->
    <section class="mb-5">
      <!-- Encabezado. Tabla. -->
      <div class="d-sm-flex align-items-center justify-content-between mb-4">
        <h1 class="h3 mb-0 text-gray-800">Listado de riegos</h1>
      </div>
      <!--/ .Encabezado. Tabla. -->
      <!--Tabla de riegos-->
      <div class="table-responsive">
        <table id="dtTablaRiegos" class="table table-hover table-bordered"
width="100%">
          <thead class="default-color white-text">
            <tr>
              <th class="th-sm"> - Encharcado
              </th>
              <th class="th-sm"> - Hora
              </th>
              <th class="th-sm"> - Moist. Final
              </th>
              <th class="th-sm"> - Moist. Inicial
              </th>
              <th class="th-sm"> - Momento
              </th>
              <th class="th-sm"> - Duración
              </th>
            </tr>
          </thead>
          <tbody>
          </tbody>
          <tfoot class="default-color white-text">
            <tr>
              <th class="th-sm"> - Encharcado
              </th>
```

```
                <th class="th-sm"> - Hora
                </th>
                <th class="th-sm"> - Moist. Final
                </th>
                <th class="th-sm"> - Moist. Inicial
                </th>
                <th class="th-sm"> - Momento
                </th>
                <th class="th-sm"> - Duración
                </th>
              </tr>
            </tfoot>
          </table>
        </div>
        <!--/ Tabla de riegos-->
      </section>
      <!--Section: Table-->

    </div>
  </main>
  <!--Main layout-->
  <!-- Footer -->
  <footer class="page-footer font-small default-color">

    <!-- Copyright -->
    <div class="footer-copyright text-center py-3">© 2019 Copyright:
      <a href="https://huertotool.firebaseapp.com/"> My Little Orchard</a>
    </div>
    <!-- Copyright -->

  </footer>
  <!-- Footer -->

</body>

<!-- SCRIPTS -->
<!-- JQuery -->
<script type="text/javascript" src="assets/js/jquery-3.3.1.min.js"></script>
<!-- Bootstrap tooltips -->
<script type="text/javascript" src="assets/js/popper.min.js"></script>
<!-- Bootstrap core JavaScript -->
<script type="text/javascript" src="assets/js/bootstrap.min.js"></script>
<!-- MDB core JavaScript -->
<script type="text/javascript" src="assets/js/mdb.js"></script>
<!-- JQUERY Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/jquery.dataTables.min.js"></script>
<!-- MDBootstrap Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/datatables2.min.js"></script>
<!-- DataTables Select JS -->
<script type="text/javascript" src="assets/vendor/DataTables/datatables-
select2.min.js"></script>
<!-- Load Firebase and Firestore -->
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-auth.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-firestore.js"></script>
<!-- Nuestra app firebase -->
<script type="text/javascript" src="./assets/js/huerto-fb.js"></script>

<script>

  window.addEventListener('load', function () {

    initApp();

  });

  $(document).ready(function () {

    // Modal Sign-out
    $('#botonSignOut').on('click', function () {

      clickSignoutButton();
    });
```

```
    // Modal de alertas
    $('#botonAlertas').on('click', function () {

      getAlertas();
    });

    // Actualizamos la vista
    getListadoRiegos();

  });

</script>

</html>
```

Archivo: siembras.html

```
<!DOCTYPE html>
<html lang="en">

<head>

  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta http-equiv="x-ua-compatible" content="ie=edge">

  <title>App Control. Siembras.</title>

  <!-- Font Awesome -->
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css">
  <!-- Bootstrap core CSS -->
  <link href="assets/css/bootstrap.min.css" rel="stylesheet">
  <!-- Material Design Bootstrap -->
  <link href="assets/css/mdb.min.css" rel="stylesheet">
  <!-- SB Admin styles -->
  <link href="assets/css/sb-admin-2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables  -->
  <link href="assets/vendor/DataTables/datatables2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables Select  -->
  <link href="assets/vendor/DataTables/datatables-select2.min.css" rel="stylesheet">
  <!-- Huerto tool styles -->
  <link href="assets/css/style.css" rel="stylesheet">

</head>

<body id="panelPrincipal" class="fixed-sn light-blue-skin">

  <!--Barra de menu-->
  <header>
    <!--NavBar-->
    <nav class="navbar fixed-top navbar-expand-md navbar-dark default-color scrolling-
navbar">
      <a class="navbar-brand" href="#"><strong><i class="fas fa-carrot fa-2x ml-2
orange-text"></i></strong></a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
        aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
          <li class="nav-item">
            <a class="nav-link" href="index.html">Modulo</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="tareas.html">Tareas</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="plantas.html">Plantas</a>
          </li>
        </ul>
```

```
        <ul class="navbar-nav ml-auto nav-flex-icons">
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" href="sistema.html">
              <i class="fas fa-microchip"></i>
              <span class="clearfix d-none d-sm-inline-block">Sistema</span>
            </a>
          </li>
          <li class="nav-item">
            <a id="botonAlertas" class="nav-link waves-effect waves-light" data-
toggle="modal"
               data-target="#modalShowAlertas">
              <i class="far fa-bell"></i>
              <span class="clearfix d-none d-sm-inline-block">Alertas</span>
            </a>
          </li>
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" data-toggle="modal" data-
target="#modalConfirmSignOut">
              <i class="fas fa-sign-out-alt fa-border"></i>
            </a>
          </li>
        </ul>
      </div>
    </nav>
    <!--/.NavBar-->
  </header>
  <!--/ .Barra de menu-->

  <!--Main layout-->
  <main>
    <div class="container-fluid" style="margin-top:100px;">

      <!--Section: Modals-->
      <section>
        <!-- Modal. Alertas -->
        <div class="modal fade top" id="modalShowAlertas" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloAlertas">Alertas</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <!-- List group links -->
                <div class="list-group">
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Horas
                    de luz
                    <span id="valHorasDeLuz" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Humedad
                    <span id="valHumedad" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Moisture
                    <span id="valMoisture" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Temperatura
                    <span id="valTemperatura" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                </div>
                <!-- List group links -->
              </div>
```

```
              <div class="modal-footer">
                <button type="button" class="btn btn-sm btn-info" data-
dismiss="modal">Aceptar</button>
              </div>
            </div>
          </div>
        </div>
        <!--/ .Modal. Alertas -->
        <!--Modal: modalSignOut-->
        <div class="modal fade top" id="modalConfirmSignOut" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <!--Content-->
            <div class="modal-content text-center">
              <!--Header-->
              <div class="modal-header d-flex justify-content-center">
                <p class="heading">Salir de la app?</p>
              </div>

              <!--Body-->
              <div class="modal-body">
                <i class="fas fa-bell fa-2x animated rotateIn"></i>
              </div>

              <!--Footer-->
              <div class="modal-footer justify-content-center">
                <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                <button type="button" id="botonSignOut" class="btn btn-sm btn-
primary">Salir</button>
              </div>
            </div>
            <!--/.Content-->
          </div>
        </div>
        <!--/ Modal: modalSignOut-->
        <!-- Modal. Nueva recoleccion -->
        <div class="modal fade top" id="modalNuevaRecoleccion" tabindex="-1"
role="dialog" aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloNuevaRecoleccion">Nueva
recolección</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <form class="border border-light">

                  <label for="inputKgRecoleccion">Cantidad</label>
                  <input type="text" id="inputKgRecoleccion" class="form-control mb-1"
placeholder="Kg. Decimal.">

                </form>
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                <button type="button" class="btn btn-sm btn-primary"
id="guardarRecoleccion">Guardar</button>
              </div>
            </div>
          </div>
        </div>
        <!--/ .Modal. Nueva recoleccion -->
      </section>
      <!--/ Section: Modals-->

      <!--Section: Table-->
      <section class="mb-5">
        <!-- Encabezado. Tabla. -->
        <div class="d-sm-flex align-items-center justify-content-between mb-4">
          <h1 class="h3 mb-0 text-gray-800">Listado de Siembras</h1>
```

```
          <div class="d-sm-flex align-items-left justify-content-between">
            <button type="button" class="btn btn-success btn-sm waves-effect waves-
light"
              id="recolectarPlanta">Recolectar</button>
          </div>
        </div>
        <!--/ .Encabezado. Tabla. -->
        <!--Tabla de Siembras-->
        <div class="table-responsive">
          <table id="dtTablaSiembras" class="table table-hover table-bordered"
width="100%">
            <thead class="default-color white-text">
              <tr>
                <th></th>
                <th>ID</th>
                <th class="th-sm"> - Planta.</th>
                <th class="th-sm"> - Superficie.</th>
                <th class="th-sm"> - Momento de siembra.</th>
              </tr>
            </thead>
            <tbody>
            </tbody>
            <tfoot class="default-color white-text">
              <tr>
                <th></th>
                <th>ID</th>
                <th class="th-sm"> - Planta.</th>
                <th class="th-sm"> - Superficie.</th>
                <th class="th-sm"> - Momento de siembra.</th>
              </tr>
            </tfoot>
          </table>
        </div>
        <!--/ Tabla de Siembras-->
      </section>
      <!--Section: Table-->

    </div>
  </main>
  <!--Main layout-->

  <!-- Footer -->
  <footer class="page-footer font-small default-color">

    <!-- Copyright -->
    <div class="footer-copyright text-center py-3">© 2019 Copyright:
      <a href="https://huertotool.firebaseapp.com/"> My Little Orchard</a>
    </div>
    <!-- Copyright -->

  </footer>
  <!-- Footer -->

</body>

<!-- SCRIPTS -->
<!-- JQuery -->
<script type="text/javascript" src="assets/js/jquery-3.3.1.min.js"></script>
<!-- Bootstrap tooltips -->
<script type="text/javascript" src="assets/js/popper.min.js"></script>
<!-- Bootstrap core JavaScript -->
<script type="text/javascript" src="assets/js/bootstrap.min.js"></script>
<!-- MDB core JavaScript -->
<script type="text/javascript" src="assets/js/mdb.js"></script>
<!-- JQUERY Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/jquery.dataTables.min.js"></script>
<!-- MDBootstrap Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/datatables2.min.js"></script>
<!-- DataTables Select JS -->
<script type="text/javascript" src="assets/vendor/DataTables/datatables-
select2.min.js"></script>

<!-- Load Firebase and Firestore -->
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-app.js"></script>
```

```
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-auth.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-firestore.js"></script>
<!-- Nuestra app firebase -->
<script type="text/javascript" src="./assets/js/huerto-fb.js"></script>

<script>

  var tableSiembras;

  window.addEventListener('load', function () {

    initApp();

  });

  $(document).ready(function () {

    // Modal Sign-out
    $('#botonSignOut').on('click', function () {

      clickSignoutButton();
    });

    // Modal de alertas
    $('#botonAlertas').on('click', function () {

      getAlertas();
    });

    // Modal nueva recoleccion
    $("#recolectarPlanta").on('click', function () {

      if (!tableSiembras.rows('.selected').any()) {

        alert("Seleccione una siembra.");

      } else {

        $("#modalNuevaRecoleccion").modal('show');
      }

    });

    // Recolectar planta
    $('#guardarRecoleccion').on('click', function () {

      var datosRecoleccion = {

        "idPlanta": "",
        "Kg": "",
        "momento": ""
      };

      // Obtenemos los valores de recoleccion
      const momento = firebase.firestore.Timestamp.fromDate(new Date());
      var row = tableSiembras.row('.selected');
      var data = row.data();

      // Asignamos los valores al objeto
      datosRecoleccion.idPlanta = data.idPlanta;
      datosRecoleccion.Kg = $('#inputKgRecoleccion').val();
      datosRecoleccion.momento = momento;

      if (validarRecoleccion(datosRecoleccion.Kg)) {

        recolectarPlantaEnFirebase(datosRecoleccion)
          .then(function (idRecoleccion) {

            // Despues de recolectar suprimimos la siembra
            return borrarSiembraEnFirebase(data.id);
          })
          .then(function () {

            //Actualizamos la vista
            $('#modalNuevaRecoleccion').modal('hide');
```

```
            // Eliminamos la fila del dataSet.
            row.remove();

            // Actualizamos la vista sin repaginar.
            tableSiembras.draw(false);
          })
          .catch(function (err) {

            // Se ha producido un error en la cadena de acciones asincronas
            alert('Error al recolectar planta.', err);
          });

      }
      else {

        alert("Revise los datos del formulario.")
      }

    });

    // Actualizamos la vista
    getListadoSiembras();

  });

</script>

</html>
```

Archivo: signin.html

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta http-equiv="x-ua-compatible" content="ie=edge">

  <title>App Control. Sign-In.</title>

  <!-- Font Awesome -->
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css">
  <!-- Bootstrap core CSS -->
  <link href="assets/css/bootstrap.min.css" rel="stylesheet">
  <!-- Material Design Bootstrap -->
  <link href="assets/css/mdb.min.css" rel="stylesheet">
  <!-- SB Admin styles -->
  <link href="assets/css/sb-admin-2.min.css" rel="stylesheet">
  <!-- Huerto tool styles -->
  <link href="assets/css/style.css" rel="stylesheet">

  <!-- Load Firebase and Firestore -->
  <!-- The core Firebase JS SDK is always required and must be listed first -->
  <script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-app.js"></script>
  <script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-auth.js"></script>
  <script src="https://www.gstatic.com/firebasejs/ui/4.5.1/firebase-ui-
auth__es.js"></script>
  <link type="text/css" rel="stylesheet"
href="https://www.gstatic.com/firebasejs/ui/4.5.1/firebase-ui-auth.css" />

  <!-- Nuestra app firebase -->
  <script type="text/javascript" src="./assets/js/huerto-fb.js"></script>

  <!-- Vamos a utilizar FirebaseUI web. -->
  <script type="text/javascript">

    // FirebaseUI config.
    var uiConfig = {
      signInSuccessUrl: 'index.html',
      signInOptions: [
        // Seleccionamos la forma de autenticarse en la app.
```

```
        firebase.auth.EmailAuthProvider.PROVIDER_ID
      ]

    };

    // Inicializamos FirebaseUI Widget.
    var ui = new firebaseui.auth.AuthUI(firebase.auth());
    // El metodo start espera hasta que el DOM esta cargado.
    ui.start('#firebaseui-auth-container', uiConfig);

  </script>

</head>

<body class="fixed-sn light-blue-skin" style="display:block">

  <!--Main Navigation-->
  <header>
    <!--NavBar top-->
    <nav class="navbar fixed-top navbar-expand-md navbar-dark default-color scrolling-
navbar">
      <a class="navbar-brand" href="#"><strong><i class="fas fa-carrot fa-2x ml-2
orange-text"></i></strong></a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
        aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
    </nav>
    <!--/.NavBar top-->
  </header>

  <!--Main layout-->
  <main>

    <div class="container-fluid" style="margin-top:100px;">

      <!--Section: Main panel-->
      <section class="mb-5">
        <!-- Panel de configuración.  -->
        <div class="row justify-content-center">
          <div class="col-sm-9 col-md-6 col-xl-4">
            <!-- Form. Login -->
            <div id="firebaseui-auth-container"></div>
            <!--/ .Form. Login -->
          </div>
        </div>
        <!--/ .Panel de configuración. -->
      </section>
      <!--Section: Main panel-->

    </div>
  </main>
  <!--Main layout-->


  <!-- Footer -->
  <footer class="page-footer font-small default-color">

    <!-- Copyright -->
    <div class="footer-copyright text-center py-3">© 2019 Copyright:
      <a href="https://huertotool.firebaseapp.com/"> My Little Orchard</a>
    </div>
    <!-- Copyright -->

  </footer>
  <!-- Footer -->

</body>

<!-- SCRIPTS -->
<!-- JQuery -->
<script type="text/javascript" src="assets/js/jquery-3.3.1.min.js"></script>
<!-- Bootstrap tooltips -->
<script type="text/javascript" src="assets/js/popper.min.js"></script>
```

```
<!-- Bootstrap core JavaScript -->
<script type="text/javascript" src="assets/js/bootstrap.min.js"></script>
<!-- MDB core JavaScript -->
<script type="text/javascript" src="assets/js/mdb.js"></script>

</html>
```

Archivo: sistema.html

```
<!DOCTYPE html>
<html lang="en">

<head>

  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta http-equiv="x-ua-compatible" content="ie=edge">

  <title>App Control. Sistema.</title>

  <!-- Font Awesome -->
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css">
  <!-- Bootstrap core CSS -->
  <link href="assets/css/bootstrap.min.css" rel="stylesheet">
  <!-- Material Design Bootstrap -->
  <link href="assets/css/mdb.min.css" rel="stylesheet">
  <!-- SB Admin styles -->
  <link href="assets/css/sb-admin-2.min.css" rel="stylesheet">
  <!-- Huerto tool styles -->
  <link href="assets/css/style.css" rel="stylesheet">

</head>

<body id="panelPrincipal" class="fixed-sn light-blue-skin">

  <!--Main Navigation-->
  <header>
    <!--NavBar-->
    <nav class="navbar fixed-top navbar-expand-md navbar-dark default-color scrolling-
navbar">
      <a class="navbar-brand" href="#"><strong><i class="fas fa-carrot fa-2x ml-2
orange-text"></i></strong></a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
        aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
          <li class="nav-item">
            <a class="nav-link" href="index.html">Modulo</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="tareas.html">Tareas</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="plantas.html">Plantas</a>
          </li>
        </ul>
        <ul class="navbar-nav ml-auto nav-flex-icons">
          <li class="nav-item active">
            <a class="nav-link waves-effect waves-light" href="sistema.html">
              <i class="fas fa-microchip"></i>
              <span class="clearfix d-none d-sm-inline-block">Sistema</span>
            </a>
          </li>
          <li class="nav-item">
            <a id="botonAlertas" class="nav-link waves-effect waves-light" data-
toggle="modal"
              data-target="#modalShowAlertas">
              <i class="far fa-bell"></i>
              <span class="clearfix d-none d-sm-inline-block">Alertas</span>
            </a>
          </li>
```

```
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" data-toggle="modal" data-
target="#modalConfirmSignOut">
              <i class="fas fa-sign-out-alt fa-border"></i>
            </a>
          </li>
        </ul>
      </div>
    </nav>
    <!--/.NavBar-->
  </header>
  <!--/ .Main Navigation-->

  <!--Main layout-->
  <main>
    <div class="container-fluid" style="margin-top:100px;">

      <!--Section: Modals-->
      <section>
        <!-- Modal. Alertas -->
        <div class="modal fade top" id="modalShowAlertas" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloAlertas">Alertas</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <!-- List group links -->
                <div class="list-group">
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Horas
                    de luz
                    <span id="valHorasDeLuz" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Humedad
                    <span id="valHumedad" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Moisture
                    <span id="valMoisture" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Temperatura
                    <span id="valTemperatura" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                </div>
                <!-- List group links -->
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-sm btn-info" data-
dismiss="modal">Aceptar</button>
              </div>
            </div>
          </div>
        </div>
        <!--/ .Modal. Alertas -->
        <!--Modal: modalSignOut-->
        <div class="modal fade top" id="modalConfirmSignOut" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <!--Content-->
            <div class="modal-content text-center">
```

```
              <!--Header-->
              <div class="modal-header d-flex justify-content-center">
                <p class="heading">Salir de la app?</p>
              </div>

              <!--Body-->
              <div class="modal-body">
                <i class="fas fa-bell fa-2x animated rotateIn"></i>
              </div>

              <!--Footer-->
              <div class="modal-footer justify-content-center">
                <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                <button type="button" id="botonSignOut" class="btn btn-sm btn-
primary">Salir</button>
              </div>
            </div>
            <!--/.Content-->
          </div>
        </div>
        <!--/ Modal: modalSignOut-->
      </section>
      <!--Section: Modals-->

      <!--Section: Main panel-->
      <section class="mb-5">
        <!-- Encabezado. Sistema. -->
        <div class="d-sm-flex align-items-center justify-content-between mb-4">
          <h1 class="h3 mb-0 text-gray-800">Sistema</h1>
        </div>
        <!--/ .Encabezado. Sistema. -->
        <!-- Panel de Sistema.  -->
        <div class="row">

          <!-- Temperatura de tierra -->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Temp. Tierra</div>
                    <div id="valTempTierra" class="mb-0 font-weight-bold text-gray-
800">--</div>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Temperatura de tierra. -->

          <!-- Temperatura de aire -->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Temp. Aire</div>
                    <div id="valTempAire" class="mb-0 font-weight-bold text-gray-800">--
</div>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Temperatura de aire. -->

          <!-- Humedad de aire -->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Hum. Aire</div>
```

```
                    <div id="valHumAire" class="mb-0 font-weight-bold text-gray-800">--
</div>
                  </div>
                </div>
              </div>
            </div>
            <!--/ .Humedad de aire. -->

            <!-- Luz visible -->
            <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
              <div class="card border-left-info shadow h-100 py-2">
                <div class="card-body">
                  <div class="row no-gutters align-items-center">
                    <div class="col mr-2">
                      <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Luz. Visible</div>
                      <div id="valLuzVisible" class="mb-0 font-weight-bold text-gray-
800">--</div>
                    </div>
                  </div>
                </div>
              </div>
            </div>
            <!--/ .Luz visible. -->

            <!-- Luz UV -->
            <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
              <div class="card border-left-info shadow h-100 py-2">
                <div class="card-body">
                  <div class="row no-gutters align-items-center">
                    <div class="col mr-2">
                      <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Luz. UV</div>
                      <div id="valLuzUV" class="mb-0 font-weight-bold text-gray-800">--
</div>
                    </div>
                  </div>
                </div>
              </div>
            </div>
            <!--/ .Luz UV. -->

            <!-- Moisture -->
            <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
              <div class="card border-left-info shadow h-100 py-2">
                <div class="card-body">
                  <div class="row no-gutters align-items-center">
                    <div class="col mr-2">
                      <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Moisture</div>
                      <div id="valMoist" class="mb-0 font-weight-bold text-gray-800">--
</div>
                    </div>
                  </div>
                </div>
              </div>
            </div>
            <!--/ .Moisture -->

          </div>
          <div class="row">

            <!-- Regado hoy-->
            <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
              <div class="card border-left-info shadow h-100 py-2">
                <div class="card-body">
                  <div class="row no-gutters align-items-center">
                    <div class="col mr-2">
                      <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Regado hoy</div>
                      <div id="valRegadoHoy" class="mb-0 font-weight-bold text-gray-800">-
-</div>
                    </div>
                  </div>
                </div>
```

```
            </div>
          </div>
          <!--/ .Regado hoy. -->

          <!-- In -Out -->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">In - Out</div>
                    <div id="valInOut" class="mb-0 font-weight-bold text-gray-800">--
</div>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!--/ .In -Out. -->

          <!-- Tiempo de riego -->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Tiempo. Riego</div>
                    <div id="valTiempoDeRiego" class="mb-0 font-weight-bold text-gray-
800">--</div>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Tiempo de riego. -->

          <!-- Hora de riego -->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Hora de riego</div>
                    <div id="valHoraDeRiego" class="mb-0 font-weight-bold text-gray-
800">--</div>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Hora de riego. -->

          <!-- Moisture inicial -->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Moist. Inicial</div>
                    <div id="valMoistureInicial" class="mb-0 font-weight-bold text-gray-
800">--</div>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Moisture inicial. -->

          <!-- Moisture final -->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
```

```
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Moist. Final</div>
                    <div id="valMoistureFinal" class="mb-0 font-weight-bold text-gray-
800">--</div>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Moisture final -->

        </div>
        <div class="row">

          <!-- Horas de luz-->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Horas de luz</div>
                    <div id="valHorasLuz" class="mb-0 font-weight-bold text-gray-800">--
</div>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Horas de luz. -->

          <!-- Estado sistema -->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Estado sistema</div>
                    <div id="valEstadoSistema" class="mb-0 font-weight-bold text-gray-
800">--</div>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Estado sistema. -->

          <!-- Km -->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Km</div>
                    <div id="valKm" class="mb-0 font-weight-bold text-gray-800">--</div>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Km. -->

          <!-- Fc -->
          <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Fc</div>
                    <div id="valFc" class="mb-0 font-weight-bold text-gray-800">--</div>
                  </div>
                </div>
```

```
            </div>
          </div>
        </div>
        <!--/ .Fc. -->

        <!-- On - Off -->
        <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
          <div class="card border-left-info shadow h-100 py-2">
            <div class="card-body">
              <div class="row no-gutters align-items-center">
                <div class="col mr-2">
                  <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">On - Off</div>
                  <div id="valOnOff" class="mb-0 font-weight-bold text-gray-800">--
</div>
                </div>
              </div>
            </div>
          </div>
        </div>
        <!--/ .On - Off. -->

        <!-- State of charge -->
        <div class="col-sm-6 col-md-4 col-lg-2 mb-4">
          <div class="card border-left-info shadow h-100 py-2">
            <div class="card-body">
              <div class="row no-gutters align-items-center">
                <div class="col mr-2">
                  <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">State of charge</div>
                  <div id="valStateOfCharge" class="mb-0 font-weight-bold text-gray-
800">--</div>
                </div>
              </div>
            </div>
          </div>
        </div>
        <!--/ .State of charge -->

      </div>
      <!--/ .Panel de Sistema. -->
      <!-- Configuración. -->
      <div class="d-sm-flex align-items-center justify-content-between mb-4">
        <h1 class="h3 mb-0 text-gray-800">Configurar</h1>
      </div>
      <!--/ .Configuración. -->
      <div class="row">

        <!-- Encender - apagar -->
        <div class="col-sm-6 mb-4">
          <div class="card border-left-info shadow h-100 py-2">
            <div class="card-body">
              <div class="row no-gutters align-items-center">
                <div class="col mr-2">
                  <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Encendido del sistema</div>
                  <div class="custom-control custom-switch mt-2">
                    <span class="h5 mb-0 font-weight-bold text-gray-800 d-flex
justify-content-left">
                      <span class="h5 mb-0 mr-4 font-weight-bold text-gray-
800">Off</span>
                      <input type="checkbox" class="custom-control-input"
id="setOnOff">
                      <label class="custom-control-label h5 mb-0 ml-4 font-weight-bold
text-gray-800"
                        for="setOnOff">On</label>
                    </span>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
        <!--/ .Encender - apagar. -->

        <!-- Interior - exterior -->
        <div class="col-sm-6 mb-4">
```

```
            <div class="card border-left-info shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Interior - Exterior</div>
                    <div class="custom-control custom-switch mt-2">
                      <span class="h5 mb-0 font-weight-bold text-gray-800 d-flex
justify-content-left">
                        <span class="h5 mb-0 mr-4 font-weight-bold text-gray-
800">In</span>
                        <input type="checkbox" class="custom-control-input"
id="setInOut">
                        <label class="custom-control-label h5 mb-0 ml-4 font-weight-bold
text-gray-800"
                          for="setInOut">Out</label>
                      </span>
                    </div>
                  </div>
                </div>
              </div>
            </div>
            <!--/ .Interior - exterior. -->

          </div>
          <div class="row">

            <!-- Tiempo de reposo -->
            <div class="col-sm-4 mb-4">
              <div class="card border-left-info shadow h-100 py-2">
                <div class="card-body">
                  <div class="row no-gutters align-items-center">
                    <div class="col mr-2">
                      <label for="setRestTime" class="text-xs font-weight-bold text-info
text-uppercase mb-1">Tiempo de
                        reposo</label>
                      <input type="range" class="custom-range" min="20" max="60"
id="setRestTime">
                    </div>
                  </div>
                </div>
              </div>
            </div>
            <!--/ .Tiempo de reposo. -->

            <!-- Tiempo de publicación -->
            <div class="col-sm-4 mb-4">
              <div class="card border-left-info shadow h-100 py-2">
                <div class="card-body">
                  <div class="row no-gutters align-items-center">
                    <div class="col mr-2">
                      <label for="setPublishTime" class="text-xs font-weight-bold text-
info text-uppercase mb-1">Tiempo de
                        publicación</label>
                      <input type="range" class="custom-range" min="1" max="60"
id="setPublishTime">
                    </div>
                  </div>
                </div>
              </div>
            </div>
            <!--/ .Tiempo de publicación. -->

            <!-- Reset riego -->
            <div class="col-sm-4 mb-4">
              <div class="card border-left-info shadow h-100 py-2">
                <div class="card-body">
                  <div class="row no-gutters align-items-center">
                    <div class="col mr-2">
                      <div class="text-xs font-weight-bold text-info text-uppercase mb-
1">Reset riego</div>
                      <div class="h5 mb-0 font-weight-bold text-gray-800">Ok</div>
                    </div>
                    <div class="col-auto">
```

```
                    <button type="button" class="btn btn-primary btn-sm waves-effect
waves-light"
                      id="resetRiego">Reset</button>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!--/ .Reset riego. -->

        </div>
      </section>
      <!--Section: Main panel-->

    </div>
  </main>
  <!--Main layout-->

  <!-- Footer -->
  <footer class="page-footer font-small default-color">

    <!-- Copyright -->
    <div class="footer-copyright text-center py-3">© 2019 Copyright:
      <a href="https://huertotool.firebaseapp.com/"> My Little Orchard</a>
    </div>
    <!-- Copyright -->

  </footer>
  <!-- Footer -->

</body>

<!-- SCRIPTS -->
<!-- JQuery -->
<script type="text/javascript" src="assets/js/jquery-3.3.1.min.js"></script>
<!-- Bootstrap tooltips -->
<script type="text/javascript" src="assets/js/popper.min.js"></script>
<!-- Bootstrap core JavaScript -->
<script type="text/javascript" src="assets/js/bootstrap.min.js"></script>
<!-- MDB core JavaScript -->
<script type="text/javascript" src="assets/js/mdb.js"></script>
<!-- Particle JS API -->
<script type="text/javascript" src="https://cdn.jsdelivr.net/npm/particle-api-
js@8/dist/particle.min.js"></script>
<!-- Particle HUERTO API -->
<script type="text/javascript" src="assets/js/huerto-particle.js"></script>

<!-- Load Firebase and Firestore -->
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-auth.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-firestore.js"></script>
<!-- Nuestra app firebase -->
<script type="text/javascript" src="./assets/js/huerto-fb.js"></script>

<script>

  window.addEventListener('load', function () {

    initApp();

  });

  $(document).ready(function () {

    var tempTierra;
    var tempAire;
    var humedadAire;
    var luzVisible;
    var UVIndex;
    var moisture;
    var regadoHoy;
    var inOut;
    var probLluvia;
    var tiempoDeRiego;
    var horaDeRiego;
    var moistureInicial;
```

```
var moistureFinal;
var horasLuz;
var estadoSistema;
var Km;
var Fc;
var onOff;
var stateOfCharge;
var tipoError;

// Modal Sign-out
$('#botonSignOut').on('click', function () {

  clickSignoutButton();
});

// Modal de alertas
$('#botonAlertas').on('click', function () {

  getAlertas();
});

// Encender - apagar
$('#setOnOff').on('change', function () {

  var onOff = $("#setOnOff").prop('checked') ? "On" : "Off";
  alert("Nuevo estado del sistema: " + onOff);
});

// Interior - exterior
$('#setInOut').on('change', function () {

  var inOut = $("#setInOut").prop('checked') ? "Out" : "In";
  alert("Nuevo estado del sistema: " + inOut);
});

// Tiempo de reposo
$('#setRestTime').on('change', function () {

  alert("Nuevo tiempo de reposo: " + $("#setRestTime").val());
});

// Tiempo de publicacion
$('#setPublishTime').on('change', function () {

  alert("Nuevo tiempo de publicación: " + $("#setPublishTime").val());
});

// Restablecer riego
$('#resetRiego').on('click', function () {

  alert("Va a restablecerse la configuración de riego.");
});

// Actualizamos la vista

//Temperatura de tierra
tempTierra = obtenerVariable("Temp_Tierra").then(function (valor) {
  if (valor != "Err") {

    $("#valTempTierra").text(valor.toFixed(2));
  } else {

    $("#valTempTierra").text(valor);
  }

  return valor;
});

// Temperatura de aire
tempAire = obtenerVariable("Temp_Aire").then(function (valor) {

  if (valor != "Err") {

    $("#valTempAire").text(valor.toFixed(2));
  } else {
```

```
        $("#valTempAire").text(valor);
      }

      return valor;
    });

    // Humedad de aire
    humedadAire = obtenerVariable("Humedad_Aire").then(function (valor) {

      if (valor != "Err") {

        $("#valHumAire").text(valor.toFixed(2));
      } else {

        $("#valHumAire").text(valor);
      }

      return valor;
    });

    // Luz visible
    luzVisible = obtenerVariable("Luz_Visible").then(function (valor) {

      if (valor != "Err") {

        $("#valLuzVisible").text(valor.toFixed(2));
      } else {

        $("#valLuzVisible").text(valor);
      }

      return valor;
    });

    // UV Index
    UVIndex = obtenerVariable("Luz_UV").then(function (valor) {

      if (valor != "Err") {

        $("#valLuzUV").text(valor.toFixed(2));
      } else {

        $("#valLuzUV").text(valor);
      }

      return valor;
    });

    // Moisture
    moisture = obtenerVariable("Moisture").then(function (valor) {

      if (valor != "Err") {

        $("#valMoist").text(valor.toFixed(2));
      } else {

        $("#valMoist").text(valor);
      }

      return valor;
    });

    // Regado hoy
    regadoHoy = obtenerVariable("Regado_Hoy").then(function (valor) {

      $("#valRegadoHoy").text(valor);
      return valor;
    });

    // Probabilidad de lluvia
    regadoHoy = obtenerVariable("Prob_Lluvia").then(function (valor) {

      $("#valProbLluvia").text(valor);
      return valor;
    });

    // In-OUT
```

```
inOut = obtenerVariable("In_Out").then(function (valor) {

  $("#valInOut").text(valor);
  if (valor == "In") {

    $("#setInOut").prop('checked', false);
  } else {

    $("#setInOut").prop('checked', true);
  }
  return valor;
});

// Tiepo de riego
tiempoDeRiego = obtenerVariable("Tiempo_Riego").then(function (valor) {

  if (valor != "Err") {

    $("#valTiempoDeRiego").text(valor.toFixed(2));
  } else {

    $("#valTiempoDeRiego").text(valor);
  }

  return valor;
});

// Hora de riego
horaDeRiego = obtenerVariable("Hora_Riego").then(function (valor) {

  $("#valHoraDeRiego").text(valor);
  return valor;
});

// Moisture inicial
moistureInicial = obtenerVariable("Moisture_Ini").then(function (valor) {

  if (valor != "Err") {

    $("#valMoistureInicial").text(valor.toFixed(2));
  } else {

    $("#valMoistureInicial").text(valor);
  }

  return valor;
});

// Moisture final
moistureFinal = obtenerVariable("Moisture_Fin").then(function (valor) {

  if (valor != "Err") {

    $("#valMoistureFinal").text(valor.toFixed(2));
  } else {

    $("#valMoistureFinal").text(valor);
  }

  return valor;
});

// Horas de luz
horasLuz = obtenerVariable("Horas_Luz").then(function (valor) {

  $("#valHorasLuz").text(valor);
  return valor;
});

// Estado del sistema
estadoSistema = obtenerVariable("Estado_AF").then(function (valor) {

  $("#valEstadoSistema").text(valor);
  return valor;
});
```

```
    // Cte del modulo
    Km = obtenerVariable("Km").then(function (valor) {

      if (valor != "Err") {

        $("#valKm").text(valor.toFixed(2));
      } else {

        $("#valKm").text(valor);
      }

      return valor;
    });

    // Factor de correccion
    Fc = obtenerVariable("Fc").then(function (valor) {

      if (valor != "Err") {

        $("#valFc").text(valor.toFixed(2));
      } else {

        $("#valFc").text(valor);
      }

      return valor;
    });

    // Voltaje de carga
    onOff = obtenerVariable("On_Off").then(function (valor) {
      $("#valOnOff").text(valor);
      if (valor == "Off") {

        $("#setOnOff").prop('checked', false);
      } else {

        $("#setOnOff").prop('checked', true);
      }
      return valor;
    });

    // Nivel e carga
    stateOfCharge = obtenerVariable("Carga").then(function (valor) {

      if (valor != "Err") {

        $("#valStateOfCharge").text(valor.toFixed(2));
      } else {

        $("#valStateOfCharge").text(valor);
      }

      return valor;
    });

  });


</script>

</html>
```

Archivo: tareas.html

```
<!DOCTYPE html>
<html lang="en">

<head>

  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta http-equiv="x-ua-compatible" content="ie=edge">
```

```
  <title>App Control. Tareas.</title>

  <!-- Font Awesome -->
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css">
  <!-- Bootstrap core CSS -->
  <link href="assets/css/bootstrap.min.css" rel="stylesheet">
  <!-- Material Design Bootstrap -->
  <link href="assets/css/mdb.min.css" rel="stylesheet">
  <!-- SB Admin styles -->
  <link href="assets/css/sb-admin-2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables  -->
  <link href="assets/vendor/DataTables/datatables2.min.css" rel="stylesheet">
  <!-- MDBootstrap Datatables Select  -->
  <link href="assets/vendor/DataTables/datatables-select2.min.css" rel="stylesheet">
  <!-- Huerto tool styles -->
  <link href="assets/css/style.css" rel="stylesheet">

</head>

<body id="panelPrincipal" class="fixed-sn light-blue-skin">

  <!--Barra de menu-->
  <header>
    <!--NavBar-->
    <nav class="navbar fixed-top navbar-expand-md navbar-dark default-color scrolling-
navbar">
      <a class="navbar-brand" href="#"><strong><i class="fas fa-carrot fa-2x ml-2
orange-text"></i></strong></a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
        aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
          <li class="nav-item">
            <a class="nav-link" href="index.html">Modulo</a>
          </li>
          <li class="nav-item active">
            <a class="nav-link" href="tareas.html">Tareas</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="plantas.html">Plantas</a>
          </li>
        </ul>
        <ul class="navbar-nav ml-auto nav-flex-icons">
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" href="sistema.html">
              <i class="fas fa-microchip"></i>
              <span class="clearfix d-none d-sm-inline-block">Sistema</span>
            </a>
          </li>
          <li class="nav-item">
            <a id="botonAlertas" class="nav-link waves-effect waves-light" data-
toggle="modal"
              data-target="#modalShowAlertas">
              <i class="far fa-bell"></i>
              <span class="clearfix d-none d-sm-inline-block">Alertas</span>
            </a>
          </li>
          <li class="nav-item">
            <a class="nav-link waves-effect waves-light" data-toggle="modal" data-
target="#modalConfirmSignOut">
              <i class="fas fa-sign-out-alt fa-border"></i>
            </a>
          </li>
        </ul>
      </div>
    </nav>
    <!--/.NavBar-->
  </header>
  <!--/ .Barra de menu-->

  <!--Main layout-->
  <main>
```

```
    <div class="container-fluid" style="margin-top:100px;">

      <!--Section: Modals-->
      <section>
        <!-- Modal. Alertas -->
        <div class="modal fade top" id="modalShowAlertas" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="tituloAlertas">Alertas</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>
              <div class="modal-body">
                <!-- List group links -->
                <div class="list-group">
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Horas
                    de luz
                    <span id="valHorasDeLuz" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Humedad
                    <span id="valHumedad" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Moisture
                    <span id="valMoisture" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                  <a
                    class="list-group-item d-flex justify-content-start align-items-
center list-group-item-action waves-effect">Temperatura
                    <span id="valTemperatura" class="badge badge-info pull-right ml-3">-
</span>
                  </a>
                </div>
                <!-- List group links -->
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-sm btn-info" data-
dismiss="modal">Aceptar</button>
              </div>
            </div>
          </div>
        </div>
        <!--/ .Modal. Alertas -->
        <!--Modal: modalSignOut-->
        <div class="modal fade top" id="modalConfirmSignOut" tabindex="-1" role="dialog"
aria-hidden="true">
          <div class="modal-dialog modal-frame modal-top" role="document">
            <!--Content-->
            <div class="modal-content text-center">
              <!--Header-->
              <div class="modal-header d-flex justify-content-center">
                <p class="heading">Salir de la app?</p>
              </div>

              <!--Body-->
              <div class="modal-body">
                <i class="fas fa-bell fa-2x animated rotateIn"></i>
              </div>

              <!--Footer-->
              <div class="modal-footer justify-content-center">
                <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
                <button type="button" id="botonSignOut" class="btn btn-sm btn-
primary">Salir</button>
```

```
            </div>
          </div>
          <!--/.Content-->
        </div>
      </div>
      <!--/ Modal: modalSignOut-->
      <!-- Modal. Nueva tarea -->
      <div class="modal fade top" id="modalNuevaTarea" tabindex="-1" role="dialog"
aria-hidden="true">
        <div class="modal-dialog modal-frame modal-top" role="document">
          <div class="modal-content">
            <div class="modal-header">
              <h5 class="modal-title" id="tituloNuevaTarea">Nueva tarea</h5>
              <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                <span aria-hidden="true">&times;</span>
              </button>
            </div>
            <div class="modal-body">
              <form class="border border-light">

                <label for="inputPrioridadTarea">Prioridad</label>
                <select class="browser-default custom-select mb-1"
id="inputPrioridadTarea">
                  <option value="0" selected="">Baja</option>
                  <option value="1">Media</option>
                  <option value="2">Alta</option>
                  <option value="3">Urgente</option>
                </select>

                <label for="inputDescTarea">Descripción</label>
                <input type="text" maxlength="100" id="inputDescTarea" class="form-
control mb-1"
                  placeholder="Trabajos a realizar">

              </form>
            </div>
            <div class="modal-footer">
              <button type="button" class="btn btn-sm" data-
dismiss="modal">Descartar</button>
              <button type="button" class="btn btn-sm btn-primary"
id="guardarTarea">Guardar</button>
            </div>
          </div>
        </div>
      </div>
      <!--/ .Modal. Nueva tarea -->
    </section>
    <!--/ Section: Modals-->

    <!--Section: Table-->
    <section class="mb-5">
      <!-- Encabezado. Tabla. -->
      <div class="d-sm-flex align-items-center justify-content-between mb-4">
        <h1 class="h3 mb-0 text-gray-800">Listado de Tareas</h1>
        <div class="d-sm-flex align-items-left justify-content-between">
          <button type="button" class="btn btn-primary btn-sm waves-effect waves-
light" id="addTarea"
            data-toggle="modal" data-target="#modalNuevaTarea">Nueva</button>
          <button type="button" class="btn btn-danger btn-sm waves-effect waves-light"
            id="borrarTarea">Eliminar</button>
        </div>
      </div>
      <!--/ .Encabezado. Tabla. -->
      <!--Tabla de tareas-->
      <div class="table-responsive">
        <table id="dtTablaTareas" class="table table-hover table-bordered"
width="100%">
          <thead class="default-color white-text">
            <tr>
              <th></th>
              <th class="th-sm">ID
              </th>
              <th class="th-sm"> - Prioridad
              </th>
              <th class="th-sm"> - Descripción
```

```
                </th>
              </tr>
            </thead>
            <tbody>
            </tbody>
            <tfoot class="default-color white-text">
              <tr>
                <th></th>
                <th>ID</th>
                <th> - Prioridad</th>
                <th> - Descripción</th>
              </tr>
            </tfoot>
          </table>
        </div>
        <!--/ Tabla de tareas-->
      </section>
      <!--Section: Table-->

    </div>
  </main>
  <!--Main layout-->

  <!-- Footer -->
  <footer class="page-footer font-small default-color">

    <!-- Copyright -->
    <div class="footer-copyright text-center py-3">© 2019 Copyright:
      <a href="https://huertotool.firebaseapp.com/"> My Little Orchard</a>
    </div>
    <!-- Copyright -->

  </footer>
  <!-- Footer -->

</body>

<!-- SCRIPTS -->
<!-- JQuery -->
<script type="text/javascript" src="assets/js/jquery-3.3.1.min.js"></script>
<!-- Bootstrap tooltips -->
<script type="text/javascript" src="assets/js/popper.min.js"></script>
<!-- Bootstrap core JavaScript -->
<script type="text/javascript" src="assets/js/bootstrap.min.js"></script>
<!-- MDB core JavaScript -->
<script type="text/javascript" src="assets/js/mdb.js"></script>
<!-- JQUERY Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/jquery.dataTables.min.js"></script>
<!-- MDBootstrap Datatables  -->
<script type="text/javascript"
src="assets/vendor/DataTables/datatables2.min.js"></script>
<!-- DataTables Select JS -->
<script type="text/javascript" src="assets/vendor/DataTables/datatables-
select2.min.js"></script>

<!-- Load Firebase and Firestore -->
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-auth.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.15.5/firebase-firestore.js"></script>
<!-- Nuestra app firebase -->
<script type="text/javascript" src="./assets/js/huerto-fb.js"></script>

<script>

  var tableTareas;

  window.addEventListener('load', function () {

    initApp();

  });

  $(document).ready(function () {

    // Modal Sign-out
```

```
$('#botonSignOut').on('click', function () {

  clickSignoutButton();
});

// Modal de alertas
$('#botonAlertas').on('click', function () {

  getAlertas();
});

// Guardar nueva tarea
$('#guardarTarea').on('click', function () {

  var datosTarea = {

    "criticidad": 0,
    "descripcion": "Descripción de la tarea."
  };
  var laPrioridad;

  datosTarea.criticidad = $('#inputPrioridadTarea').val();
  datosTarea.descripcion = $('#inputDescTarea').val();

  // Debe añadir la tarea a firebase
  guardarTareaEnFirebase(datosTarea).then(function (idNuevaTarea) {

    switch (datosTarea.criticidad) {

      case "0":
        laPrioridad = "Baja";
        break;

      case "1":
        laPrioridad = "Media";
        break;

      case "2":
        laPrioridad = "Alta";
        break;

      case "3":
        laPrioridad = "Urgente";
        break;
    }

    // Actualizamos la tabla
    tableTareas.row.add({

      select: "",
      id: idNuevaTarea,
      Prioridad: laPrioridad,
      Descripción: datosTarea.descripcion
    });

    // Actualizamos la vista.
    tableTareas.draw();

    // Cerramos el formulario
    $('#modalNuevaTarea').modal('hide');
  }).catch(function (err) {

    // Se ha producido un error en la cadena de acciones asincronas
    alert('Error al crear nueva tarea.', err);
  });

});

// Eliminar tarea
$('#borrarTarea').on('click', function () {

  var row = tableTareas.row('.selected');
  var data = row.data();

  // Utilizamos el ID para eliminar la fila en firestore
  var ID = data.id;
```

```
      // Debe eliminar la tarea en firebase
      borrarTareaEnFirebase(ID).then(function () {

        // Eliminamos la fila del dataSet.
        row.remove();

        // Actualizamos la vista sin repaginar.
        tableTareas.draw(false);

      }).catch(function (err) {

        // Se ha producido un error en la cadena de acciones asincronas
        alert('Error al borrar tarea.', err);
      });

    });

    // Actualizamos la vista
    getListadoTareas();

  });

</script>

</html>
```