Grado en Ingeniería Informática

Escuela Politécnica Superior de Jaén

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén
Departamento de Informática

Trabajo Fin de Grado

ANÁLISIS, GESTIÓN Y AUTOMATIZACIÓN DE RIEGO EN CULTIVOS AGRICOLAS

Alumno: **Antonio Jesús Martos Lara**

Tutor: Miguel Ángel García Cumbreras
Dpto: Departamento de informática
Co/Tutor: Salud María Jiménez Zafra

Julio, 2020

Alumno: Antonio Jesús Martos Lara

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Don Miguel Ángel García Cumbreras, tutor del Proyecto Fin de Carrera titulado: Olivapp, que presenta Antonio Jesús Martos Lara, autoriza su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, Julio de 2020

El alumno:

Antonio Jesús Martos Lara

Antonio Jesús Martos Lara
Firmado digitalmente por Antonio Jesús Martos Lara
Fecha: 2020.07.31 10:59:53 +02'00'

Los tutores:

Miguel Ángel García Cumbreras
Salud María Jiménez Zafra

GARCIA CUMBRERAS MIGUEL ANGEL - 52871272E
Firmado digitalmente por GARCIA CUMBRERAS MIGUEL ANGEL - 52871272E
Fecha: 2020.07.31 12:53:06 +02'00'

JIMENEZ ZAFRA SALUD MARIA - 77370897R
Firmado digitalmente por JIMENEZ ZAFRA SALUD MARIA - 77370897R
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-77370897R, givenName=SALUD MARIA, sn=JIMENEZ ZAFRA, cn=JIMENEZ ZAFRA SALUD MARIA - 77370897R
Fecha: 2020.07.31 12:31:24 +02'00'

INDICE:

INDICE ILUSTRACIONES:

INDICE TABLAS DE CONTENIDOS:

Capítulo 1: Introducción

1.1 Introducción

El objetivo principal de este TFG es la automatización de riego en cultivos agrícolas, teniendo en cuenta el monitoreo de variables ambientales y control remoto de dicho riego.

Este es un proyecto enfocado al sector primario, más concretamente a la agricultura. Una de las actividades hasta ahora menos actualizadas.

El campo se moderniza y los últimos avances tecnológicos en el sector agrario sirven para hacer más fácil el día a día al agricultor y mejorar su productividad. Hasta hace unos años estos avances tecnológicos se aplicaban a las diferentes herramientas, las cuales, facilitaban bastante el trabajo de nuestros agricultores mejorando y agilizando la recogida y labranza de sus campos. Sin embargo, uno de los retos de la tecnología en la actualidad en este sector es dar respuesta a las principales mejoras a las que se enfrenta la agricultura, como mejorar y aumentar la productividad y reducir el impacto ambiental.

En los últimos años han surgido muchas empresas que poco a poco están perfeccionando este ámbito previamente mencionado. Para ello, se están usando diferentes técnicas ya funcionales, las cuales harán posible tomar decisiones a tiempo real, mejorar la eficiencia y la productividad de las explotaciones y mejorar el aprovechamiento de los recursos.

Las diferentes tecnologías que se están empezando a implantar en las diferentes explotaciones son:

- **Sistemas de navegación GPS**: Se utiliza para los tractores o vehículos agrícolas, permitiendo optimizar labores agrícolas como serian la siembra, abonado o recolección de la cosecha, los cuales están controlados por un software para reducir costes y mejorar la eficiencia al evitar dejar zonas sin tratar. Optimizar el camino hacia la/s explotación/es y aumentar la seguridad de los agricultores en su trabajo diario.

Ilustración 1: Sistema de geolocalización

- **Drones**: Gracias a la irrupción de los drones los agricultores pueden tener imágenes aéreas de sus explotaciones para obtener una gestión de estas mucho más eficientes. Esta técnica tiene múltiples aplicaciones, entre ellas: estimación del rendimiento del cultivo, detección de plagas, optimización del uso de fertilizantes y analizar el estrés hídrico (cuando la demanda de agua es más alta que la cantidad disponible). Acciones que antes tenían que realizarse a pie con las deficiencias que conlleva.

Ilustración 2: Drone en explotación agrícola

- **Técnica láser:** Esta técnica se utiliza para saber el momento óptimo para su recolección. El láser crea un patrón de fruto y se compara con un patrón estándar de referencia, esto hará que los frutos se recolecten en el momento justo de maduración, lo que mejorara la calidad del producto.

Ilustración 3: Técnica laser, comprobación de maduración de fruta

- **Aplicaciones móviles y sensores:** La instalación de sensores en los cultivos permite al agricultor conocer a tiempo real el estado de su explotación. Se puede controlar el nivel de humedad, el estrés hídrico, la temperatura, los restos de nitrogenados… Así en base a esta medición y análisis de datos, los agricultores pueden tomar mejores decisiones, tanto preventivas como operativas. Una de las ventajas a destacar, es que el agricultor puede llevar a cabo estas gestiones desde su propia casa a través de distintas aplicaciones móviles.

Ilustración 4: Imagen de sistema inteligente de sensor temperatura con aplicación móvil

Como vemos el ámbito tecnológico para control a tiempo real en la agricultura se está implantando, y tiene muchas ventajas para el control de las explotaciones. Pero realmente este tipo de técnicas solo se están aplicando a grandes explotaciones por su elevado coste, ya que son tecnologías muy innovadoras.

1.2 Motivación

El interés por la automatización de tareas cada día es más importante, ya que una buena supervisión, gestión y control de tareas automáticas pueden ahorrarnos mucho trabajo y dinero, con lo cual podemos aumentar el rendimiento y productividad de una explotación.

Otro factor muy importante a la hora de impulsar este nuevo proyecto es el ahorro de agua, ya que grandes fincas mal gestionadas a la hora del riego pueden acarrear un gasto de agua excesivo, innecesario y sobre todo perjudicial en ciertos cultivos que no requieren altas tasas de humedad en tierra.

Usabilidad en la gestión de tareas con alta carga de trabajo, mediante aplicaciones multiplataforma, entornos fáciles e intuitivos para los usuarios y hacerles dicha gestión mucho más confortable y amena.

1.3 Objetivos concretos del TFG

Como se ha comentado previamente, el objetivo principal de este proyecto final de carrera está enfocado a la automatización de la acción de riego en una finca agrícola, con posibilidad de consultas y control del usuario en todo momento. Los objetivos detallados son los siguientes:

1. Aplicación de riego automatizado, el cual evita la dependencia de control del estado de riego por parte de los agricultores en una finca, facilitándoles así la gestión de esta tarea y evitando la presencia física y desplazamiento a esta.

2. Aplicación de un riego programado periódico por el usuario final.

3. Aplicación del riego manual, el usuario podrá activar y desactivar el riego de dicha explotación de manera remota.

4. Solucionar la dificultad del control y actuación eficiente del riego en explotaciones agrarias.

5. Consultar los diferentes riegos aplicados a nuestra explotación, su duración y la cantidad de agua utilizada aproximadamente en cada uno de ellos, clasificándolos en riegos automáticos, programados o manuales.

6. Consultar el estado del riego en dicha explotación en cualquier momento y desde cualquier lugar, siempre y cuando contemos con conexión a internet.

7. Consultar el estado del terreno de nuestra explotación, mediante los sensores que tendríamos instalados, estos sensores recopilan valores y se pueden consultar a tiempo real.

1.4 Especificaciones

Este proyecto se va a encargar de gestionar lógicamente y eficientemente, una de las actividades del sector agrario. Esta actividad es el riego de cultivos.

Contaremos con una aplicación gráfica multiplataforma, en la cual podremos ver los registros de los diferentes sensores y podremos controlar manual o automáticamente el riego para optimizarlo.

Nuestro controlador abrirá las válvulas de riego para aplicar un riego automatizado, el cual actuara mediante un algoritmo previamente programado, el cual aplicará el riego cuando valores como la humedad, temperatura o precipitaciones estén por debajo de un umbral.

Además, un aspecto fundamental de este proyecto es que el usuario pueda monitorizar los diferentes datos recogidos y almacenados en nuestro sistema. Se ha buscado darle más funcionalidad a la parte del usuario final para que también pueda programar riegos periódicos y forzar el encendido o apagado del sistema.

1.5 Alcance del sistema

Este proyecto tiene como fin que el usuario final pueda tener las estadísticas respecto a la aplicación de riego en una explotación de cultivo, a la vez que pueda gestionar dicho riego manualmente y tener en todo momento controlado el estado físico de su finca mediante datos meteorológicos cuantitativos.

Para ello el controlador debe recoger constantemente datos meteorológicos para alimentar nuestro sistema, el cual será el encargado de generar la actuación de riego mediante dichos datos recogidos y futuros datos meteorológicos recogidos de agencias meteorológicas.

La idea de este proyecto es que cualquier agricultor con una mínima inversión tanto del hardware como del software, pueda instalar este sistema en explotaciones medianas y pequeñas, ya que como hemos visto al principio de este capítulo las diferentes tecnologías que se están aplicando hoy en día son sistemas bastante complejos y caros para pequeñas explotaciones.

Toda gestión y consulta realizada se hará a través de una aplicación multiplataforma que mostraremos en capítulos posteriores, la cual nos informará y nos permitirá actuar directamente sobre nuestro sistema de riego desde cualquier ubicación, siempre y cuando contemos con conexión a internet.

1.6 Organización del resto de la memoria

- Capítulo 2: Veremos el análisis de nuestro proyecto, aspectos principales para empezar nuestra aplicación.
    - Analizaremos los usuarios finales los cuales acabaran usando dicha aplicación y veremos la funcionalidad completa de estos
    - Realizaremos la planificación del proyecto para organizar de forma correcta el transcurso del desarrollo.
    - Veremos los requisitos funcionales y no funcionales así como la metodología utilizada para el desarrollo de nuestro sistema.
    - Conoceremos la estimación de costes para la finalización y paso a producción del sistema.
- Capítulo 3: Este capítulo realizaremos el análisis y diseño del sistema.
    - Funcionalidad de cada uno de los diferentes tipos de usuarios, así como también veremos las actividades que podrán realizar en nuestro sistema.
    - Veremos la funcionalidad y el diseño de cada una de las vistas o componentes de nuestro sistema.
- Capítulo 4: Ya una vez sabemos la funcionalidad y el diseño de nuestra aplicación pasamos a analizar las diferentes herramientas y tecnologías que hay en el mercado para ver cuáles serían las que mejor se adaptarían a nuestros requisitos.

    - Analizaremos la funcionalidad incluido, así como pseudo-código de cada una de las tecnologías seleccionadas.
- Capítulo 5: Pruebas y validación. En este capítulo vamos a probar todas las funcionalidades de nuestro sistema, así como probar unitariamente los métodos de implementación.
- Capítulo 6: Conclusiones y trabajo futuro. En esta sección veremos las conclusiones obtenidas tras el desarrollo de nuestro proyecto. También veremos el posible trabajo futuro a realizar.
- Anexo I: Guía de instalación.
- Anexo II: Manual de usuario.

Capítulo 2: Análisis del proyecto

2.1 Usuarios finales

Los usuarios finales de este proyecto son los agricultores que dispongan de explotaciones de cualquier cultivo. Dicha explotación deberá de contar con agua y riego.

Uno de los principales problemas a los que nos enfrentamos en este proyecto es que, tras una serie de entrevistas a posibles usuarios finales, nos han dejado claro que sería muy conveniente establecer una usabilidad simple y básica para la utilización de esta herramienta agrícola.

También nos informaron que tienen bajos niveles de conocimientos tecnológicos, lo cual nos hace limitar en parte la funcionabilidad de dicha aplicación. Esta automatización de la labor de riego dependiente hace que dichos administradores de fincas cuenten con más facilidad y menos carga de trabajo para controlar el estado y gestionar dicha tarea.

2.2 Consulta de funcionalidades a los usuarios finales

Para analizar a posibles usuarios finales de nuestra aplicación se ha realizado un formulario de Google para analizar las respuestas de estos usuarios. Este formulario ha sido rellenado por ciertos miembros de la Cooperativa olivarera “Virgen de la Fuensanta”, la cual se ubica en la localidad de Fuensanta de Martos un pueblo de la provincia de Jaén para asegurar que llegue a agricultores que dispongan de pequeña/s y mediana/s explotación/es los cuales podrían ser futuros clientes.

Dicho formulario ha sido realizado desde la plataforma “Forms” de Google. El link del formulario es el siguiente: https://forms.gle/Y8VG3MRkDCQ9zYqq9

2.3 Resultados del cuestionario:

Una vez realizada la encuesta a 27 usuarios diferentes, hemos obtenido una serie de resultados los cuales analizaremos individualmente por cuestión.

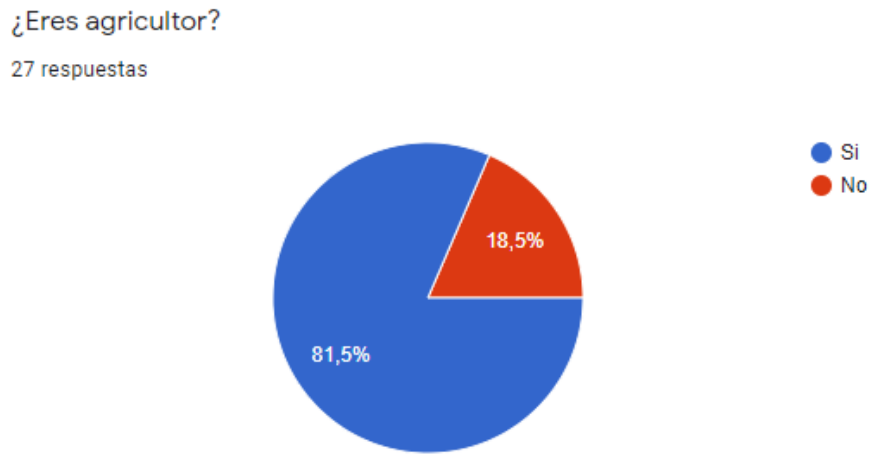


Ilustración 5: Resultado cuestión formulario

Como podemos ver obtenemos que el 81,5% de los entrevistados son agricultores (22 de los 27).

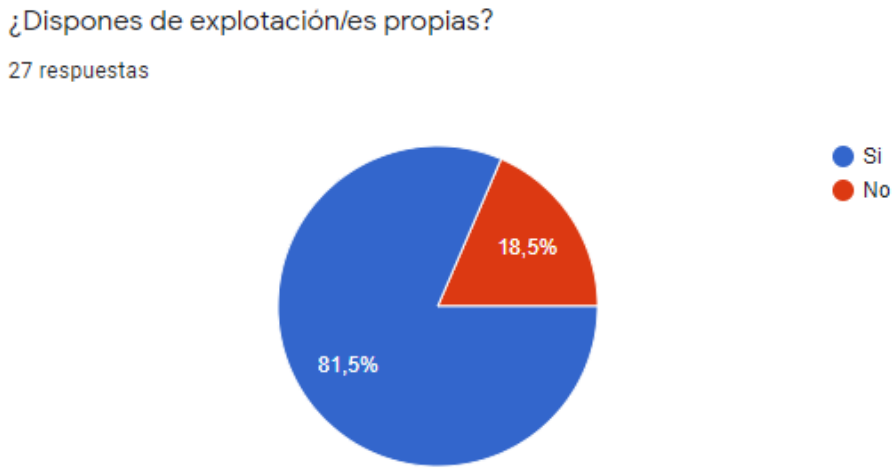


Ilustración 6: Resultado cuestión 2 formulario

En esta segunda cuestión podemos observar que el 100% de los agricultores cuentan con explotaciones propias, lo cual podrían ser futuros clientes.

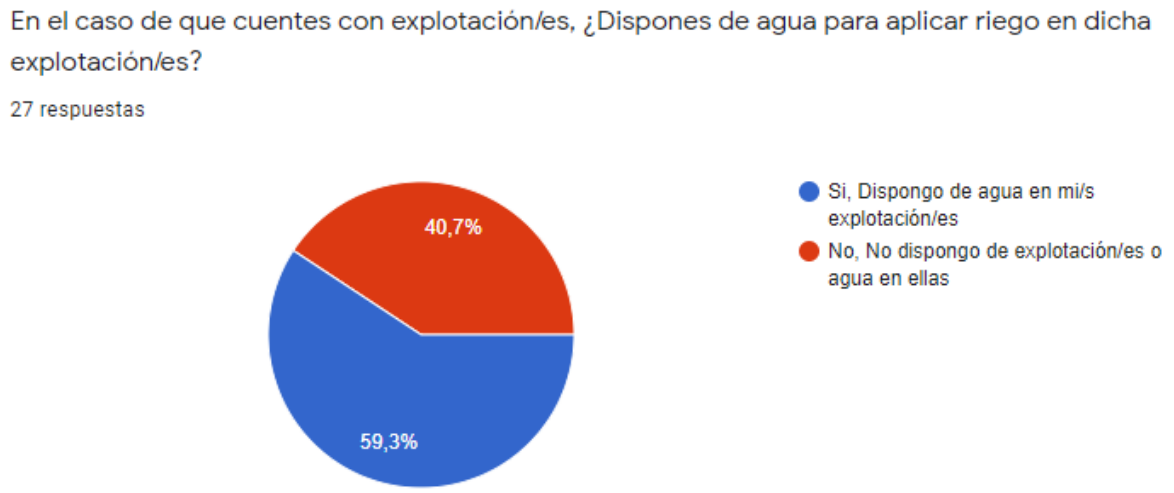


Ilustración 7: Resultados cuestión 3 del formulario

Aquí nos encontramos con una pequeña limitación, la disposición de agua en nuestra explotación. Hay muchas explotaciones las cuales no es posible disponer de agua para riego. El 59,3% de las explotaciones disponen de agua.

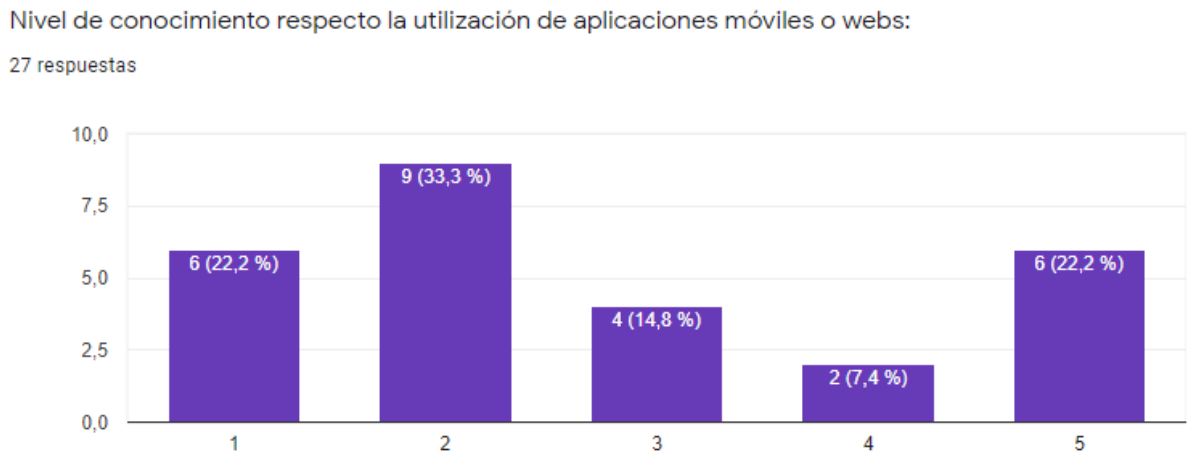


Ilustración 8: Resultados cuestión 4 formulario

Gracias a esta cuestión podremos determinar el nivel de conocimiento respecto el uso de una aplicación web o móvil. Como podemos ver el 55,5% de estos usuarios tienen bajo o ningún conocimiento a la hora de utilización de estas tecnologías.

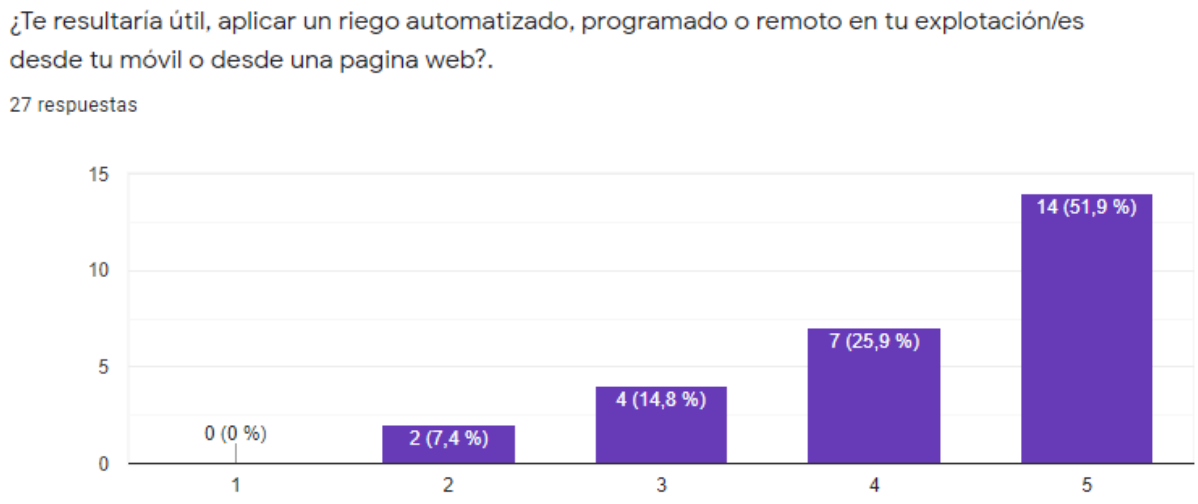


Ilustración 9: Resultados cuestión 5 formulario

Esta cuestión va dirigida a la utilidad funcional que ven los usuarios finales respecto la implantación de nuestro sistema. Más del 50% de los encuestados ven muy útil dicha implantación. Gracias a esta cuestión podemos ver un amplio marco de negocio en nuestro proyecto.

Ilustración 10: Resultados cuestión 6 del formulario

Estos resultados nos muestran que el 74,1% de los encuestados ven que el riego automático puede aumentar su producción considerablemente.

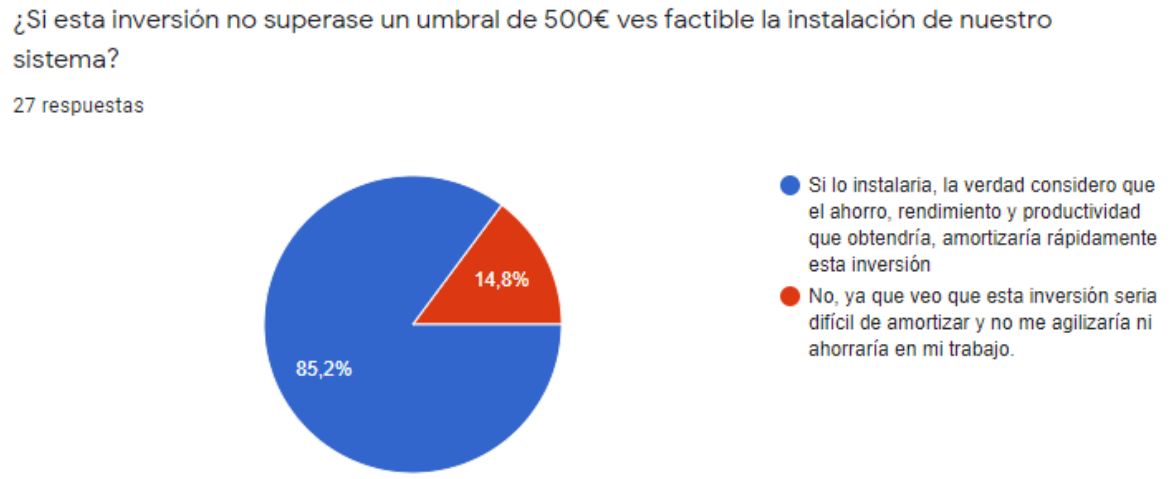


Ilustración 11: Resultados cuestión 7 del formulario

Tras establecer un umbral máximo de 500€ la mayoría de usuarios consideran que amortizarían este sistema bastante rápido por la mejora de producción.

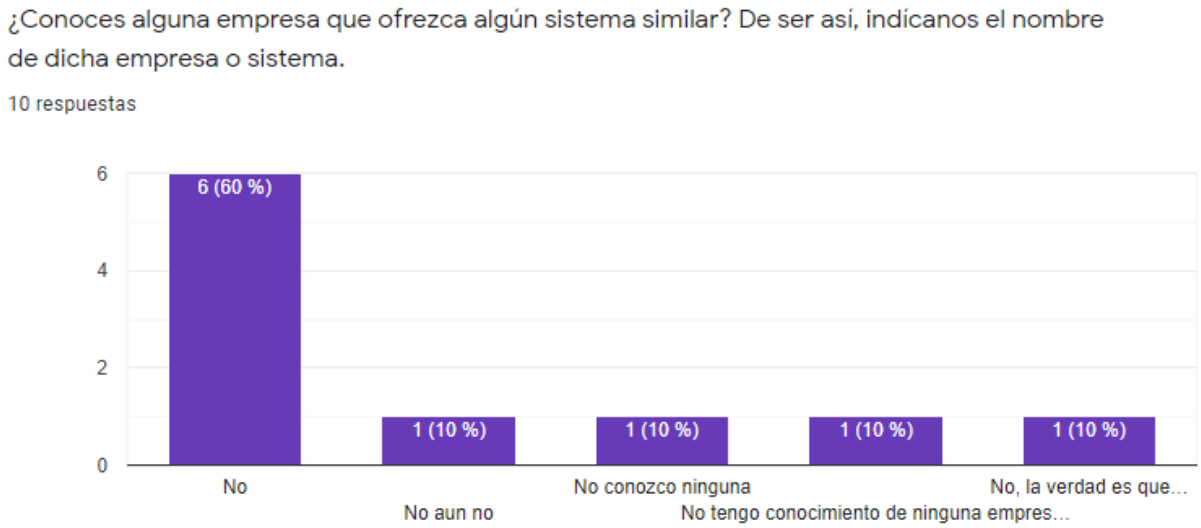


Ilustración 12: Resultados cuestión 8 del formulario

Esta última cuestión nos refleja que ninguno de los usuarios encuestados conoce o saben de la existencia de un sistema similar al nuestro. Esto nos facilita la posibilidad de la implantación de nuestro sistema por la falta de conocimiento de posibles competidores o sistemas ya implementados.

2.4 Estimación del tamaño y esfuerzo

Este proyecto al ser un Trabajo de Fin de Grado no tiene restricciones económicas, pero si temporales. La complejidad y número total de acciones se establecerán respecto al tiempo disponible.

- Contaremos en primer lugar con una aplicación multiplataforma que se encargará de la interacción con nuestro sistema.

- La aplicación contará con un panel de control o dashboard informativo. Por otra parte, también podremos interactuar controlando el riego de nuestra finca.

- Usaremos una plataforma online en la cual podremos almacenar a tiempo real los datos recogidos por nuestro sistema.

- Por otro lado, tenemos nuestro controlador-actuador que se encarga de establecer una conexión a nuestro servidor o plataforma online. Recoge datos meteorológicos mediante sensores climáticos y es el encargado de gestionar la acción de riego, activando o parando este.

2.4 Planificación Temporal

Teniendo en cuenta los objetivos fijados para este TFG, y después de ver las funcionalidades generales determinaremos los tiempos de planificación con la siguiente tabla. Esta incluye todas las tareas globales a realizar, llevando a cabo una estimación temporal de cada tarea respecto a su dificultad.

TAREA	ESTIMACIÓN EN SEMANAS
Búsqueda de información y formación	2 semanas = 50 horas
Análisis y planificación del proyecto	3 semanas = 75 horas
Diseño	3 semanas = 75 horas
Implementación código	6 semanas = 150 horas
Pruebas y evaluación	2 semanas = 50 horas
Documentación	2 semanas = 50 horas
TOTAL	**18 semanas = 450 horas**

Tabla 1: Estimación de tiempos por tareas

2.4.1 Diagrama de Gantt

Un diagrama de Gantt es una herramienta útil para planificar proyectos. Al proporcionarte una vista general de las tareas programadas, todas las partes implicadas sabrán qué tareas tienen que completarse y en qué fecha.

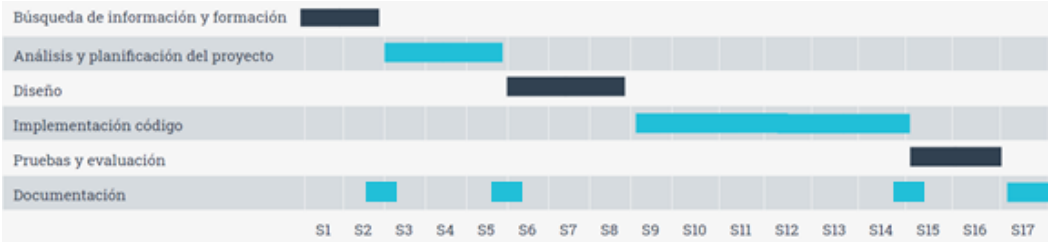


Ilustración 13: Diagrama de gantt de taréas

2.5 Estimación de costes

En esta sección, vamos a realizar una estimación de costes totales del proyecto dividiéndolo en: software, hardware, servidor/conexión y gastos de personal.

2.5.1 Software

- Desglosaremos en horas cada uno de los componentes o funciones en cada uno de los entornos.

Implementación - FrontEnd	
Bloques implementados	**Horas aproximadas**
Componente Navegación	10
Componente Menú	10
Componente Footer	10
Componente Contenido (3 vistas)	50
Modelo Login	35
Modelo Registro	20
Servicio Conexión	25
Servicio Autentificación	25
Total - Horas dedicadas	**185**

Implementación - BackEnd	
Conexión del sistema a internet	15
Recogida de valores de sensores	15
Escritura y lectura de valores en BD	16
Actuación en bomba de riego	10
Creación algoritmo de riego	6
Total – Horas dedicadas	**62**
TOTAL:	**247**

Tabla 2: Horas dedicadas en Front-en y en Back-end

2.5.2 Hardware

Aquí reflejaremos el gasto económico en los diferentes componentes de nuestro actuador. También indicaremos el precio para una posible ampliación. En posteriores capítulos explicaré más en profundidad el estudio de una posible ampliación.

Hardware - Sistema	
Controlador-Actuador, sensores y sistema de retroalimentación energética	**80€ Aproximadamente**

Tabla 3: Coste Hardware

2.5.3 Servidor y Almacenamiento

Contaremos con una serie de gastos de los servidores que almacenarán todos los datos de nuestro sistema.

Hosting y Dominio	
Componente	**Precio Aproximado**
Hosting web almacenamiento 50GB	60€
Dominio web “.com”	15€
Total:	**75€ Anualmente**

Tabla 4:Coste servidor y almacenamiento

2.5.4 Gastos Personal

Contaremos con diferentes perfiles para el desarrollo de este proyecto. Para ello analizaremos los gastos estimados que nos produciría la producción de dicho proyecto, utilizando de referencia las diferentes jerarquías, especializaciones y precios de la siguiente plataforma: https://www.linkedin.com/

Personal	
Analista en tecnología de información TIC	16€/hora * 8 horas * 10 días = 1280€
Desarrollador Junior con 1 año de experiencia en las tecnologías a desarrollar	9€/hora * 8 horas * 30 días = 2160€
Total:	**3.440 €**

Tabla 5: Coste por personal

2.5.5 Desarrollo Hardware – Controlador/actuador

Otros de los gastos a tener en cuenta son los de implementación y configuración de los sensores del dispositivo hardware. Estos serán los encargados de registrar y almacenar en nuestra BBDD los valores de los distintos sensores.

Desarrollo hardware	
Instalación y configuración sensores, desarrollador IOT junior	8€/hora * 10 horas
Implementación código	9€/hora * 8 horas * 8 días
Total:	**576€**

Tabla 6:Coste por desarrollo

2.5.6 Gastos Extra:

Hay más gastos que no se han reflejado en las estimaciones de costes anteriores. Serían servicios como la conexión a internet para poder conectar nuestro sistema en cualquier ubicación.

Gastos Extra	
Componente	**Precio Aproximado**
Tarifa 5GB GSM 4G	8€ mensuales / 96€ anuales

Tabla 7: Coste por gastos extra

2.6 Requisitos funcionales:

Un requisito funcional define una función o acción del sistema de software o sus componentes. Los requisitos funcionales son:

- El usuario podrá consultar a través de la aplicación el estado del riego para conocer en todo momento la tarea de riego a tiempo real.

- El usuario consultará las estadísticas de aplicaciones de la tarea de riego, control de caudal y gasto de agua. También se podrán consultar diferentes estadísticas climatológicas.

- El usuario realizará acciones para activar o detener dicha tarea de riego. De esta manera, podríamos controlar manualmente dicho riego sin alterar la automatización del sistema.

- El usuario podrá programar un riego periódicamente a través de nuestra aplicación.

- El usuario tendrá la posibilidad de consultar el estado de los sensores mostrando una advertencia en caso de avería o falta de actividad.

2.7 Requisitos no funcionales:

Un requisito no funcional es aquel que especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de los comportamientos específicos de este. Los requisitos no funcionales son:

2.7.1 Interfaz

La interfaz es considerada un factor muy importante para nuestro sistema. Por este motivo, tenemos que realizar la creación de una interfaz organizada, uniforme e intuitiva.

La interfaz de nuestra aplicación contará con una serie de colores básicos que no proporcionen confusión. Así usuarios con limitaciones visuales respecto el color podrán utilizarla sin problema. Esta interfaz contará con texto estructurado, imágenes, gráficos y botones correctamente definidos.

2.7.2 Accesibilidad

Se podrá acceder mediante conexión a internet y usando cualquiera de las siguientes plataformas:

- Android
- IOS
- Página Web

La única limitación de accesibilidad sería para personas con discapacidad visual total. Por limitaciones de tiempo y desarrollo se obviará la implementación de soluciones para esta discapacidad.

2.7.3 Disponibilidad

Principalmente contamos con una disponibilidad completa de 24 horas al día durante 365 días al año. Teniendo en cuenta nuestra infraestructura de almacenamiento y tratamiento de datos se realizará a través de servidores que nos garantizando así la disponibilidad total.

2.7.4 Privacidad

La aplicación puede almacenar información personal para controlar futuros comportamientos del usuario. Al ser una aplicación privada solo la usarán usuarios registrados para la gestión agrícola de la tarea de riego sin limitación en la recogida de información de la explotación.

El método de ingreso a nuestra aplicación sería mediante email y contraseña evitando el registro de nuevos usuarios. Para dar de alta un nuevo usuario, configurar el perfil completo o establecer el tipo de cultivo junto con los datos específicos de su explotación/es, deberá de hacerlo el administrador del sistema

2.8 Metodología

Desde un principio nuestro proyecto se planteó mediante una metodología en cascada. La metodología en cascada es un modelo lineal para diseñar software, que emplea un proceso de diseño secuencial. El desarrollo fluye secuencialmente desde el punto inicial hasta el punto final en varias etapas diferentes:

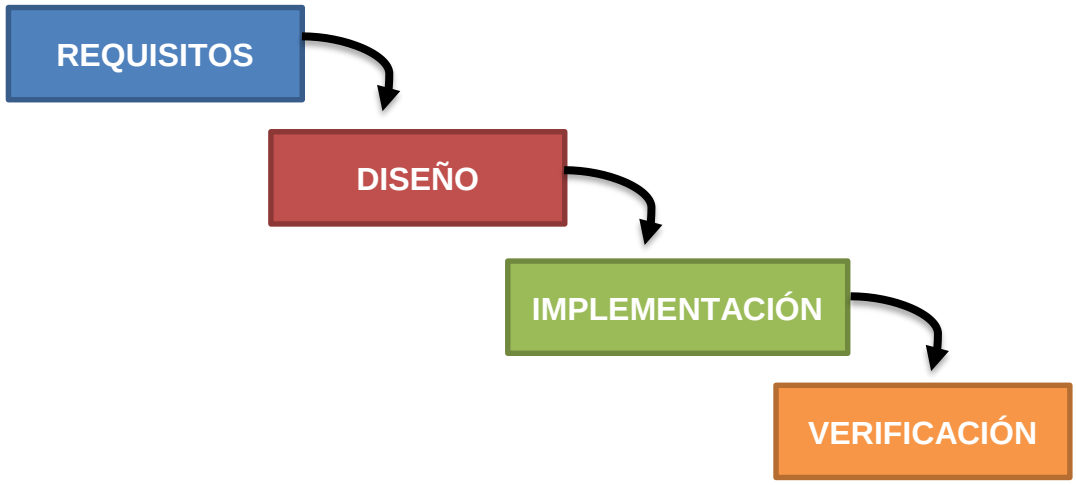


Ilustración 14: Metodología en cascada

2.8.1 Ventajas:

- ✓ Inicio con el desarrollo software con bastante rapidez. Teniendo en cuenta que el tiempo del que disponemos para el desarrollo del proyecto es bastante ajustado para la funcionabilidad y escalabilidad del sistema.
- ✓ Estimar calendarios y presupuestos con mayor precisión. Gracias a esto hemos podido organizarnos mejor a la hora de prescindir de ciertas funciones y agregar otras funcionabilidades nuevas.
- ✓ Lograr un nivel de satisfacción del cliente más elevado que en otras metodologías. Desde el comienzo, el diseño o prototipo está interactuando con el cliente.

2.8.2 Desventajas:

- × Alterar el diseño del proyecto en cualquier etapa es muy complicado. Por ello, vamos interactuando directamente con el cliente final en cada una de dichas etapas.
- × La corrección de errores puede ser muy tediosa y longeva por la gran cantidad de código desarrollado y testeado. Para intentar solventar esta deficiencia hemos utilizado Gitlab. Para gestionar y ordenar el código de nuestro proyecto, cada pequeño desarrollo o refactorización de código, se actualiza mediante un punto de actualización (Commit), perfectamente definidos.

Capítulo 3: Análisis y diseño del sistema

Una vez realizado el análisis del proyecto pasamos en este capítulo al análisis del sistema pudiendo ver su funcionalidad. A parte, veremos los diferentes perfiles de usuarios que intervienen en nuestro sistema. Por último, analizaremos las diferentes partes de nuestra aplicación y sus funcionalidades.

3.1 Actores

Un actor representa un tipo de usuario. Lo primero que haremos será definir los diferentes actores que interactúan con nuestro sistema. En nuestro caso distinguiremos entre dos tipos de actores:

- **Administrador**

  Este usuario se encargará de la administración y gestión de nuestro sistema. Será el encargado de todas las gestiones con nuestra base de datos y de la gestión de usuarios.

  También es el encargado de registrar a cada uno de los usuarios que utilicen nuestro sistema, así como añadir o modificar los datos personales como de su explotación.

- **Usuario Final:**

  Este usuario podrá monitorizar los datos almacenados a tiempo real por nuestro sistema. También podrá controlar el riego de su explotación y consultar el contenido de nuestra base de datos.

3.2 Diagramas de casos de uso

Un diagrama de casos de uso es una descripción de manera gráfica de las diferentes actividades que un actor debe desarrollar para llevar a cabo un proceso. Este diagrama sirve para identificar las diferentes funcionalidades de un actor en un mismo escenario dado.

A continuación, mostraré un diagrama frontera de casos de uso, donde veremos los diferentes actores que intervienen en nuestro sistema.

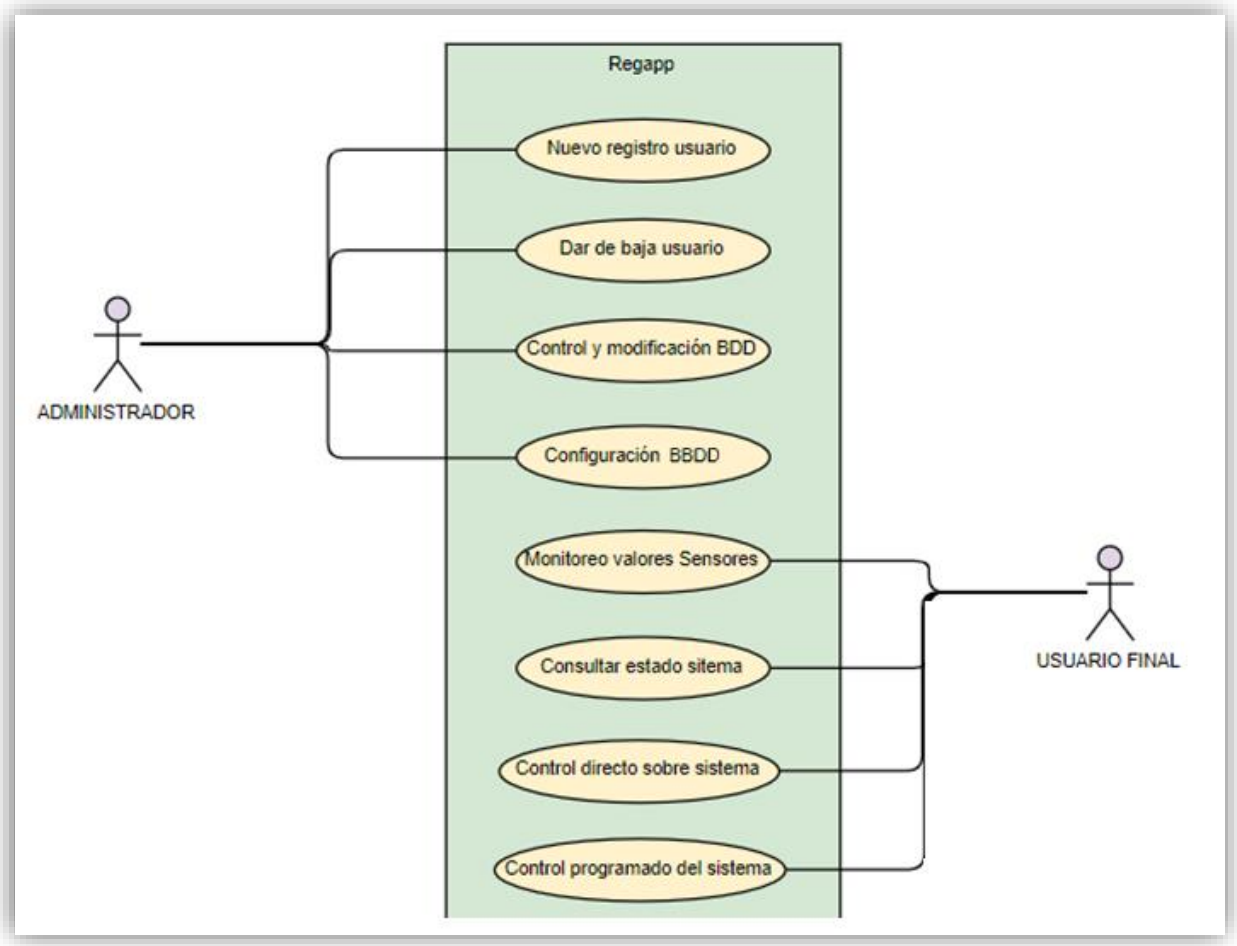


Ilustración 15: Diagrama frontera

3.3 Diagramas de casos de uso por actores

Aquí desglosaremos las acciones de cada uno de los actores que intervienen en nuestro sistema.

3.3.1 Administrador - Acceso BBDD

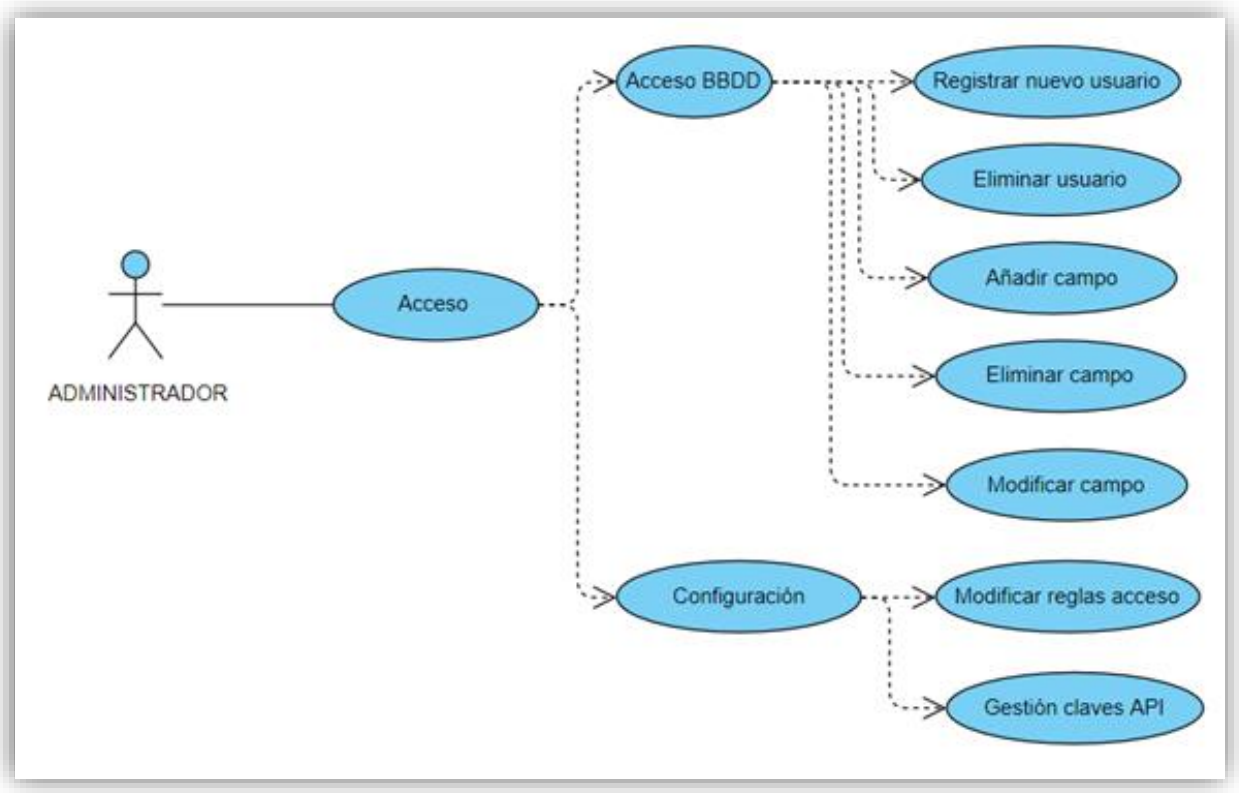


Ilustración 16: Actor adminsitrador, diagrama de casos de uso

- Actor principal: Administrador del sistema.
- Precondiciones: Nuestra base de datos debe estar completamente disponible. Tenemos que disponer de conexión a internet y contar con una cuenta de acceso.
- Pasos:
    1. Accede a BBDD
    2. Crea BBDD
    3. Modifica BBDD
    4. Configurar BBDD para las diferentes plataformas que utilizara el cliente final.
- Postcondiciones: Ninguna, el Administrador podrá alterar la BBDD o configurarla en función de las modificaciones o restricciones del usuario final.

3.3.2 Usuario final – Funcionabilidad de la aplicación

En esta sección, mostraremos los diagramas de casos de uso de los diferentes componentes con los que contará nuestra aplicación.

Estos componentes son las diferentes vistas que mostraremos. Gracias a estos diagramas podremos ver las diferentes actividades de nuestro actor (usuario final).

3.3.2.1 Inicio de sesión

En esta parte del proyecto, mostraremos un formulario de Login el cual el usuario final tendrá que rellenar y superar para acceder a los demás componentes.

El inicio de sesión se mantiene en la aplicación durante un cierto tiempo o durante un número de accesos a esta.

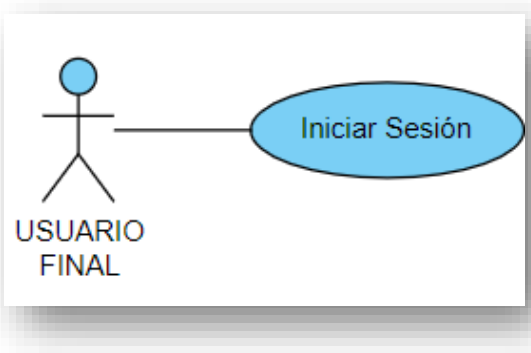


Ilustración 17: Actor usuario final en caso de uso Inicio de sesión

- Actor principal: Usuario Final.
- Precondiciones: Nuestra base de datos debe estar completamente disponible. Tenemos que disponer de conexión a internet y contar con una cuenta de acceso.
- Pasos:
    1. Introducir Email
    2. Introducir Contraseña
    3. Iniciar Sesión

- Condiciones:
  - Existir el usuario referente al email y contraseña en nuestra base de datos.
  - Disponibilidad de servicio internet y a nuestra base de datos.

3.3.2.2 Inicio

Es el componente principal que mostrará un monitoreo gráfico donde se reflejen los siguientes valores.

- Último Registro: muestra el último valor introducido en la BBDD.
- Última Semana: muestra una gráfica la cual refleja los últimos 7 días.
- AEMET: muestra la previsión meteorológica para los próximos 4 días.
- Últimos 7 Riegos Aplicados: muestra una tabla donde se indica la fecha, duración, litros aproximados y tipo de riego.

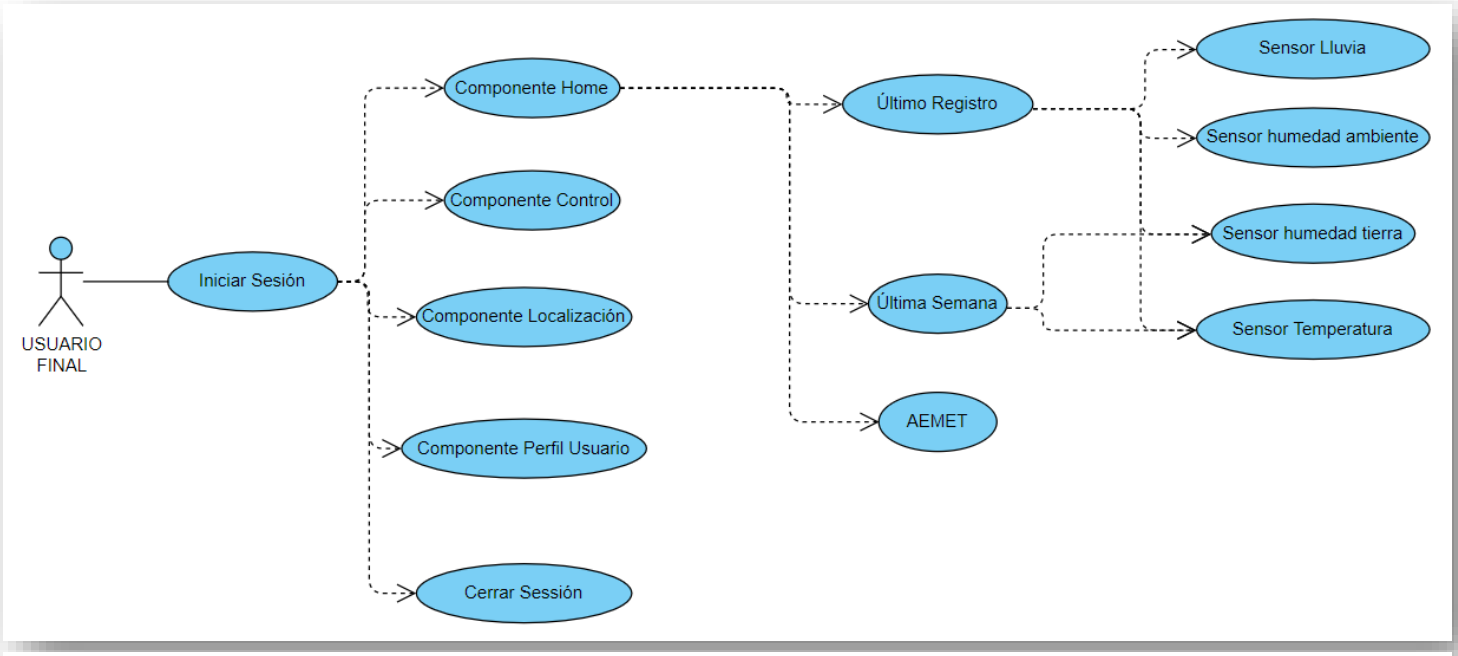


Ilustración 18: Diagrama de casos del componente inicio

- Actor principal: Usuario Final.
- Precondiciones: Base de datos, actuador-controlador y sus sensores deben estar funcionales.
  - Existir el usuario referente al email y contraseña en la base de datos.

- Disponibilidad de servicio internet y conexión a en la base de datos.

3.3.2.3 Control

Este componente será el encargado de monitorizarnos el estado en el que se encuentra nuestro sistema Arduino y sus sensores asociados. Sus posibles estados serian: Encendido/Apagado/Avería.

También tendrá una funcionalidad de control que podremos utilizar para activar, desactivar o programar el riego de nuestra finca.

- Estado Sistema: muestra información del estado de arduino y los diferentes sensores asociados.
- Riego inteligente: Activar o desactivar automatización del riego mediante nuestro algoritmo.
- Acciones directas: Activar o desactivar el riego en la finca en tiempo real.
- Riego programado: Podremos activar o desactivar un riego periódico, el cual tenemos la posibilidad de modificar.

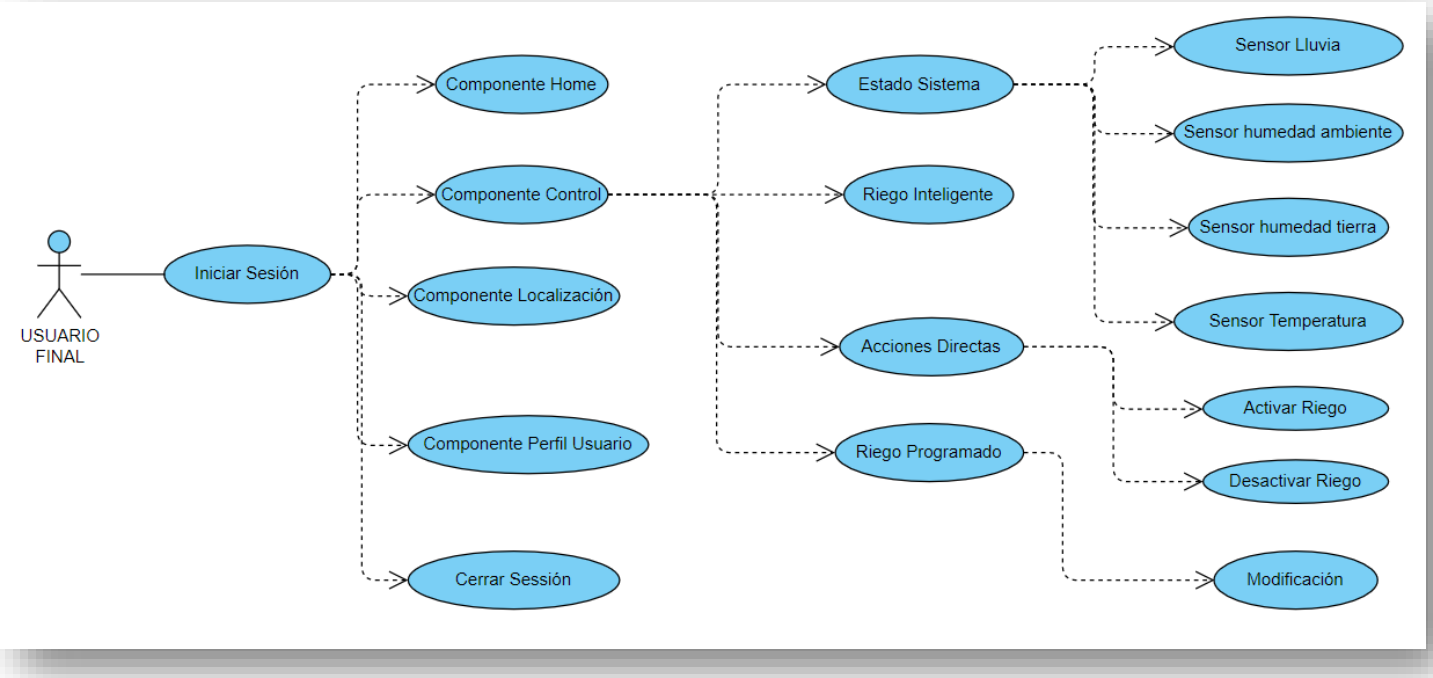


Ilustración 19:Diagrama de caso de uso del componente control

- Actor principal: usuario final.
- Precondiciones: base de datos, actuador-controlador y sus sensores deben estar funcionales.
    - Existir el usuario referente al email y contraseña en la base de datos.
    - Disponibilidad de servicio internet y conexión a en la base de datos.

3.3.2.4 Localización

En este componente mostraremos información de nuestra explotación, su ubicación y esquema de riego.

- API Google Maps: Integración en nuestro componente, visualizando la localización precisa de la finca.
- Slider Localización: Muestra una presentación de imágenes con información de la ubicación de la finca previamente almacenados en la BBDD.
- Slider Terreno: Muestra una presentación de imágenes con información del terreno de la finca previamente almacenados en la BBDD.“*Referencia Web*”, Mediante botones abrimos una nueva pestaña web en la cual podemos visitar Google Maps o Sigpac (aplicación del gobierno de registro de fincas por polígono y parcela).

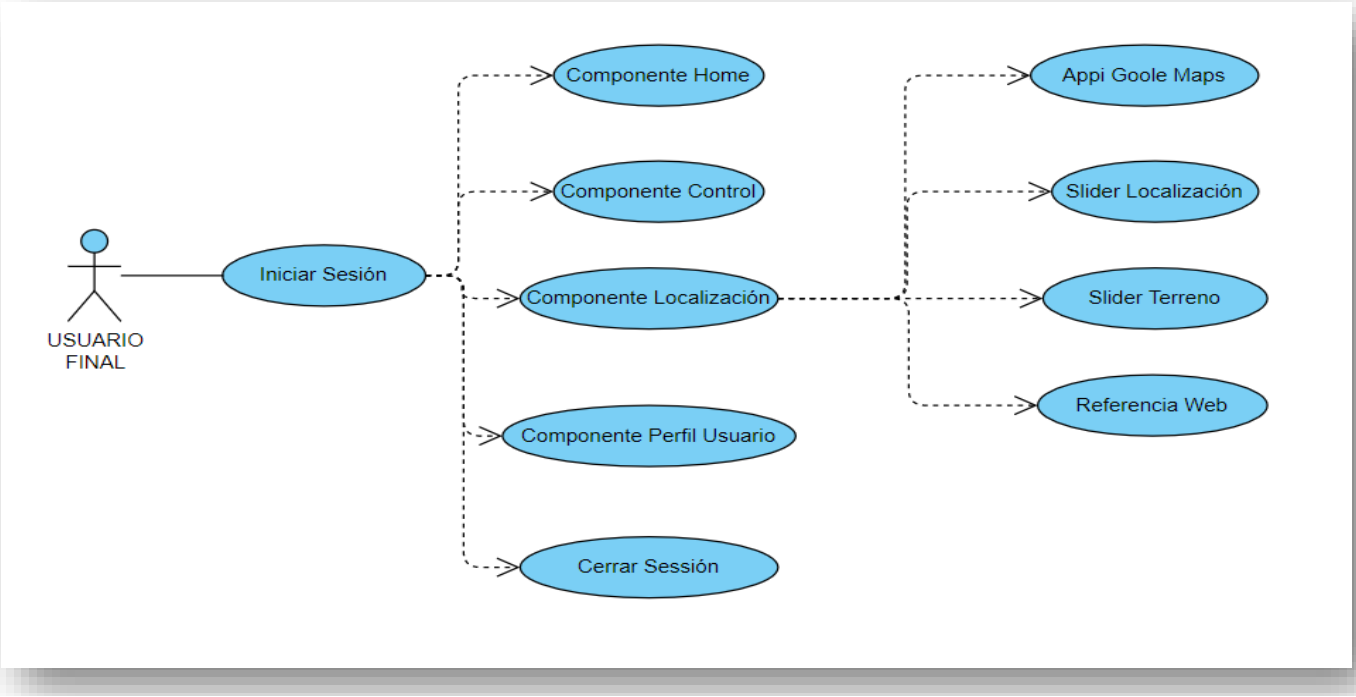


Ilustración 20: Diagrama de casos de uso del componente Localización

- Actor principal: usuario final.
- Precondiciones: base de datos, actuador-controlador y sus sensores deben estar funcionales.
    - Existir el usuario referente al email y contraseña en la base de datos.
    - Disponibilidad de servicio internet y conexión a en la base de datos.
    - Disponibilidad de uso de páginas externas (Google y Sigpac).

3.3.2.5 Perfil Usuario

En este componente podremos consultar, modificar y añadir información personal del cliente a través de un formulario.

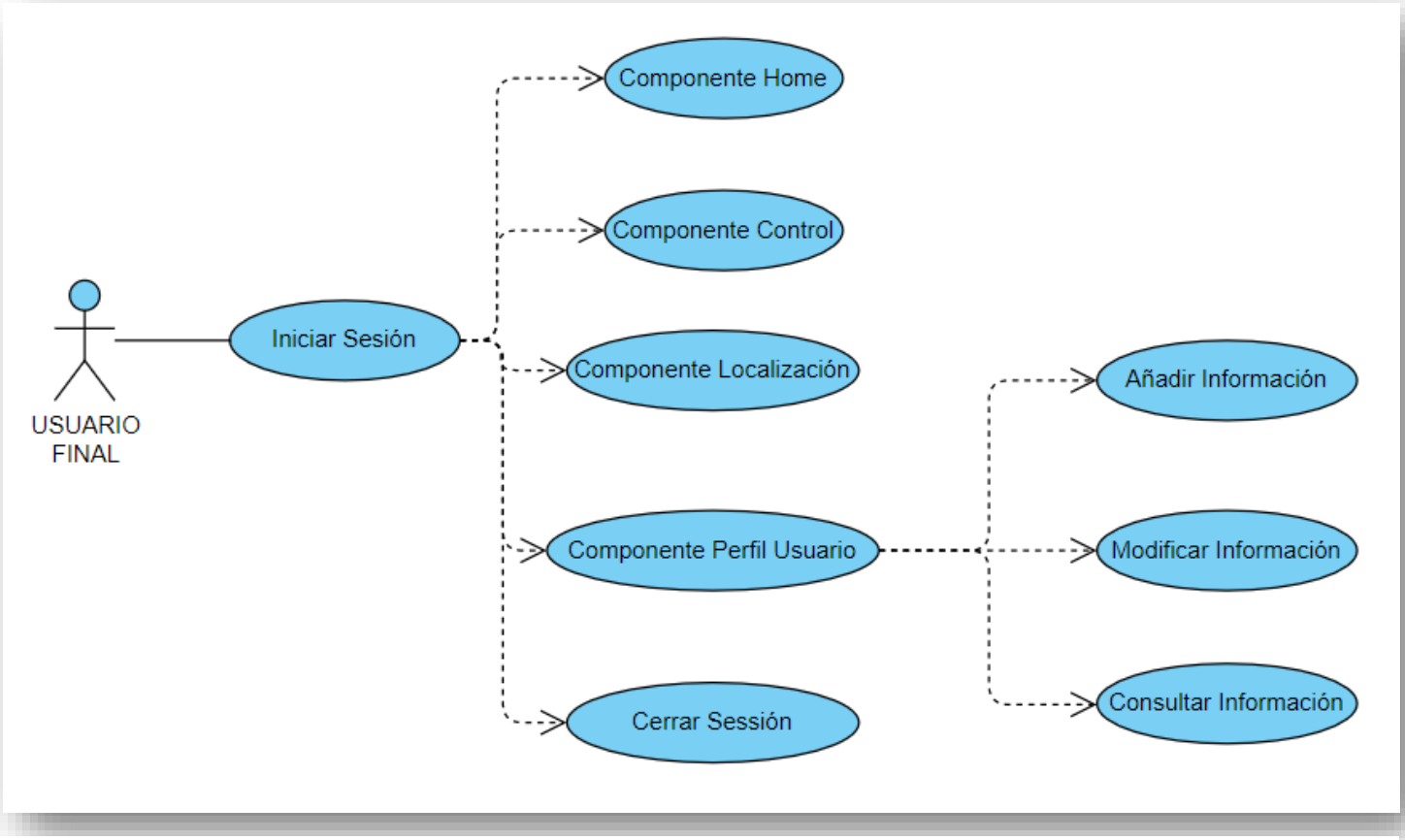


Ilustración 21: Diagrama de casos de uso del componente perfil de usuairo

- Actor principal: Usuario Final.
- Precondiciones: Base de datos, actuador-controlador y sus sensores deben estar funcionales.

- Existir el usuario referente al email y contraseña en la base de datos.
- Disponibilidad de servicio internet y conexión a en la base de datos.

3.3.2.6 Cerrar sesión

En esta funcionabilidad de la aplicación se cierra la sesión en la BBDD. Obligando al usuario a volver a iniciar sesión en caso de querer seguir usando la aplicación.

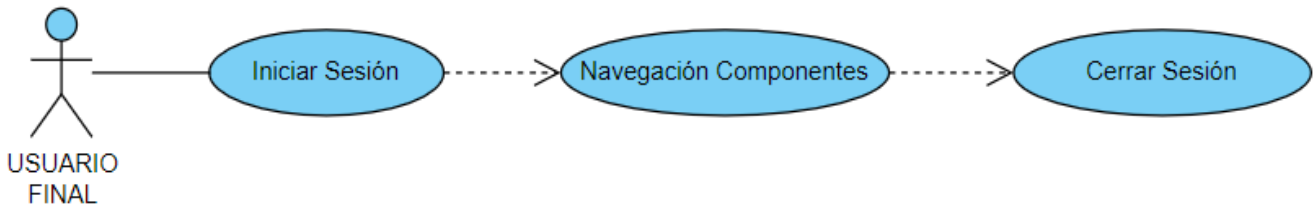


Ilustración 22: Diagrama casos de uso del usuario final en cerrar sesión

- Actor principal: Usuario Final.
- Precondiciones: El servidor Firebase debe estar funcional, tenemos que disponer de conexión a internet, y contar con una cuenta de acceso.
- Condiciones:
  - Usuario Registrado.
  - Inicio de sesión activo.
  - Disponibilidad de servicio internet y conexión a base de datos.

3.5 Diseño

En esta sección veremos tanto el diseño a nivel de arquitectura de nuestro sistema como el diseño a nivel de aplicación. Ambos diseños nos aportarán información suficiente y necesaria para posteriormente seleccionar las tecnologías a utilizar.

3.5.1 Esquema general del sistema

Mostraremos el esquema general de nuestro sistema completo:

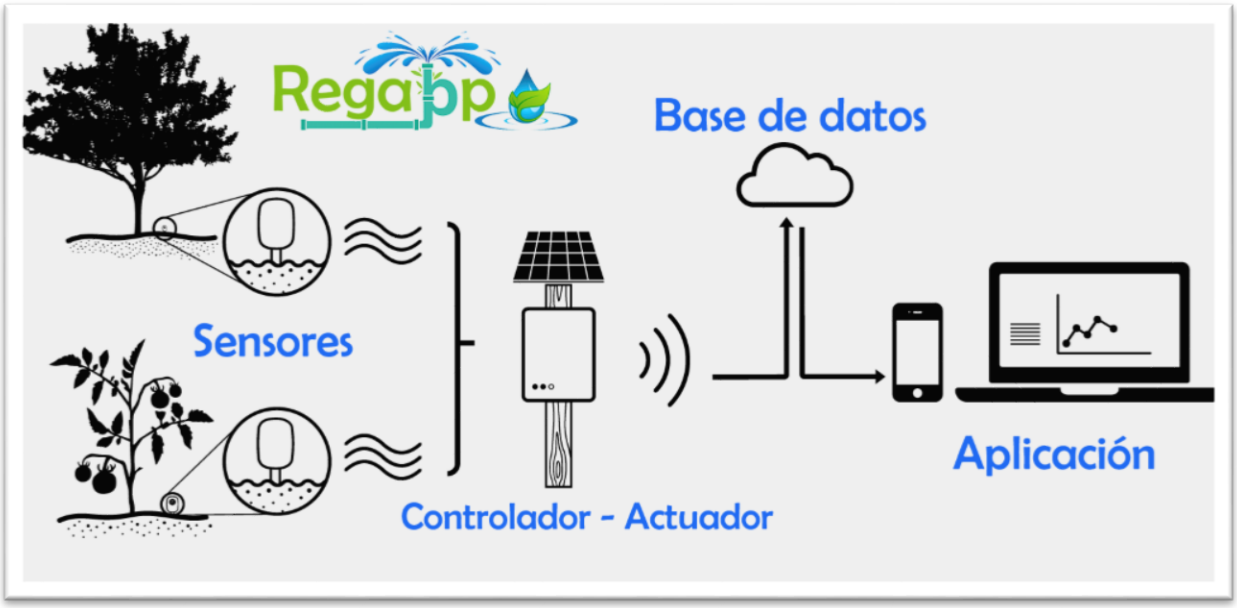


Ilustración 23: Esquema aplicación

Como podemos ver contamos con cuatro elementos indispensables para el correcto funcionamiento de nuestro sistema de riego. La disponibilidad de cada uno es indispensable. Por lo tanto, si algún elemento de nuestro sistema dejase de estar disponible por el factor que fuese, puede acarrear errores en el resto de nuestra aplicación.

3.5.2 Descripción de los elementos del sistema

Una vez hemos visto de forma general los diferentes elementos de nuestro sistema en el apartado anterior, especificaremos cada uno de estos elementos:

- **Sensores**: Se recogen valores que almacena y gestionará nuestro actuador. Contamos con 4 sensores:
  - Sensor de temperatura: Este sensor nos indicará la temperatura ambiente de una explotación en ºC.
  - Sensor de humedad superficial: Este sensor nos indicará la humedad ambiente de una explotación expresada en % de humedad.

- Sensor de humedad de la tierra: Este sensor nos indicará la humedad de la tierra a una profundidad de aproximadamente 10cm de una explotación expresada en % de humedad.
- Sensor de precipitaciones: Este sensor nos indicará diferentes valores. Por ejemplo: si está lloviendo nos indicará la intensidad y una aproximación de los l/m^2 mediante el grado de humedad y la duración.

- **Controlador-Actuador**: Es el encargado de la gestión de los valores registrados por los sensores y de almacenarlos a tiempo real en nuestra base de datos. Este será capaz de detectar problemas o incongruencias en los diferentes sensores, pudiendo almacenar en dicha base de datos códigos de error. Así nuestro usuario final mediante la aplicación podría ser informado de dichos errores o problemas.

  Nuestro controlador también estará cada cierto tiempo consultando valores de nuestra base de datos. Estos campos serán modificados por el usuario para activar, desactivar o programar este.

- **Base de datos**: Utilizaremos una BBDD a tiempo real en la nube, es decir, necesitaremos conexión a internet para realizar las diferentes inserciones y consultas. En la base de datos no es necesario contar con una estructura específica porque básicamente la utilizaremos para insertar los diferentes valores registrados por nuestros sensores en una fecha y hora concreta.

  A parte utilizaremos nuestra BBDD para la inserción de los diferentes riegos aplicados, diferenciando entre riegos manuales, programados y automáticos.

- **Aplicación:** Es la interfaz gráfica mediante la cual los usuarios finales podrán consultar y monitorizar los diferentes datos almacenados en nuestra BBDD. Esta interfaz aparte permitirá al usuario tomar el control del riego de su explotación. Esta aplicación deberá ser lo suficientemente intuitiva ya que como hemos visto nuestros usuarios finales pueden no estar familiarizados con este tipo de tecnologías.

3.5.3 Estructura de la base de datos

En este apartado veremos la estructura de nuestra BBDD, como hemos comentado anteriormente. Esta no dispone de una estructura concreta y no necesitamos mantener un orden especifico, es decir, vamos a dividirla en 4 grandes apartados diferenciados entre sí los datos que almacenan. Nuestra BBDD tendrá una estructura de árbol de directorios:

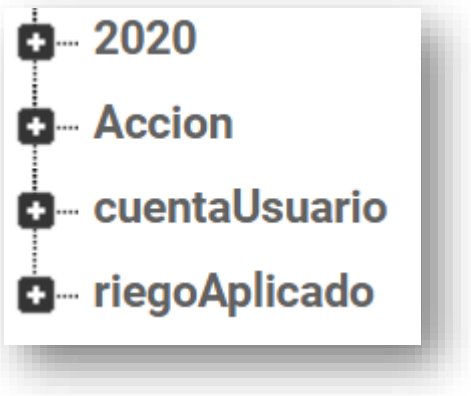


Ilustración 24: Estructura base de datos

- Registros de valores de los sensores: Como podemos ver en el siguiente ejemplo la estructura que utilizaremos seria: AÑO→ DIA/MES/AÑO→ HORA→ ATRIBUTOS Y VALORES

Ilustración 25: BBDD registro sensores

- Riegos aplicados: En esta estructura se almacenarán los diferentes riegos aplicados por los usuarios o el sistema, diferenciando entre riegos manuales y automáticos. La estructura que utilizamos sería: DIA/MES/AÑO → ATRIBUTOS Y VALORES. Veremos un ejemplo:

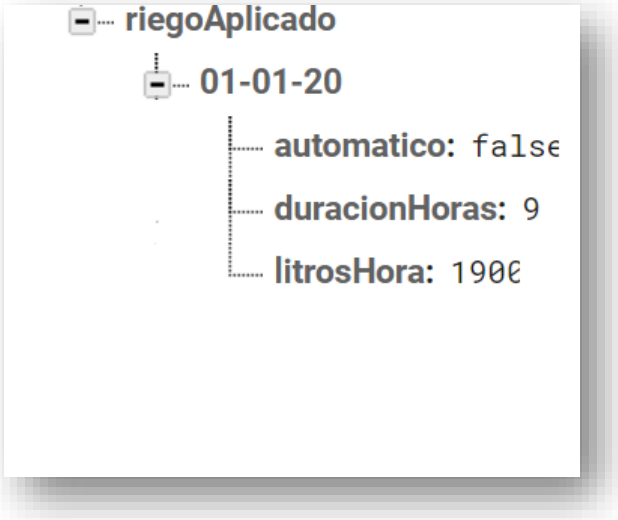


Ilustración 26: BBDD Riegos aplicados

- Cuenta de usuario: Aquí guardaremos los diferentes datos de perfil del usuario. Veamos un ejemplo:

Ilustración 27:BBDD cuenta usuario

- Acciones: En esta parte se almacenarán los diferentes valores booleanos que nos sirven para la información de estado de componentes hardware. A parte otros de estos atributos los utilizaremos para modificar ciertas funcionalidades del riego desde nuestra aplicación, mostramos un ejemplo:

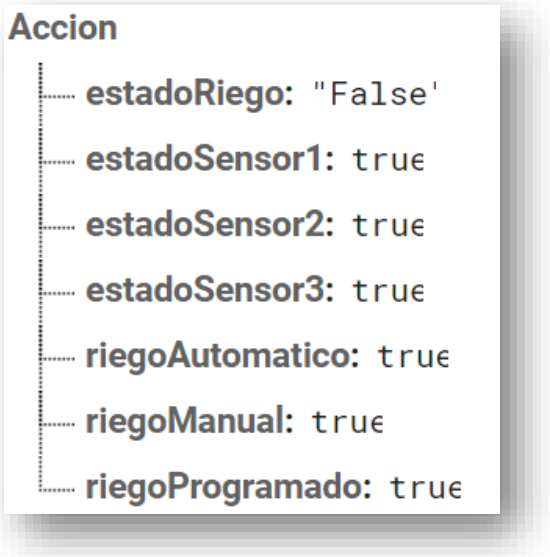


Ilustración 28: BBDD Acción control arduino

3.5.4 Diseño gráfico interfaz

Los sistemas interactivos se caracterizan por la importancia del diálogo con el usuario. La interfaz es por tanto una parte fundamental en el proceso de desarrollo y debe tenerse en cuenta desde el principio. Además, la interfaz determina en gran medida la percepción e impresión que el usuario posee de una aplicación. El usuario no está interesado en la estructura interna de la aplicación, sino en su fácil uso.

Una vez hecha la especificación, propuesto un diseño y desarrollado el código es muy difícil cambiar las características de la interacción y presentación de la información, salvo pequeños detalles. De esta forma, se obtienen interfaces muy dependientes de la estructura de los datos y sus funciones, sin tener en cuenta al usuario que va a utilizarlo.

Por lo tanto, debemos empezar con una idea clara de cómo queremos la interfaz y cómo serán las interacciones con el usuario para después desarrollar las especificaciones funcionales que sirvan de guía al diseño posterior.

Nos centraremos en los diferentes componentes para la realización del prototipado de la aplicación:

1. Login
2. Componente Inicio
3. Componente Control
4. Componente Localización
5. Componente Perfil Usuario

3.5.4.1 Sketch Aplicación

Con lápiz y papel jugaremos con ideas y bocetos. El objetivo es conseguir ideas de trabajo y explorar cualquier propuesta que se nos ocurra. Lo mejor es experimentar sin miedo a cometer errores o descargar ideas. Cuanto más tiempo pasemos jugando con las ideas y viendo cómo mejorarlas, más fácil será saber si una idea es buena o no.

Se documenta todo lo que necesitemos realizando anotaciones al margen. Estas anotaciones nos pueden servir para dotar correctamente la funcionalidad de cada vista. Ahora mostraremos los Sketch principales de los diferentes componentes.

LOGIN

Formulario de inicio de sesión para el usuario mediante email y contraseña. Cuenta con un botón para enviar dicho formulario que se ajusta al tamaño de la pantalla centrando el contenido.

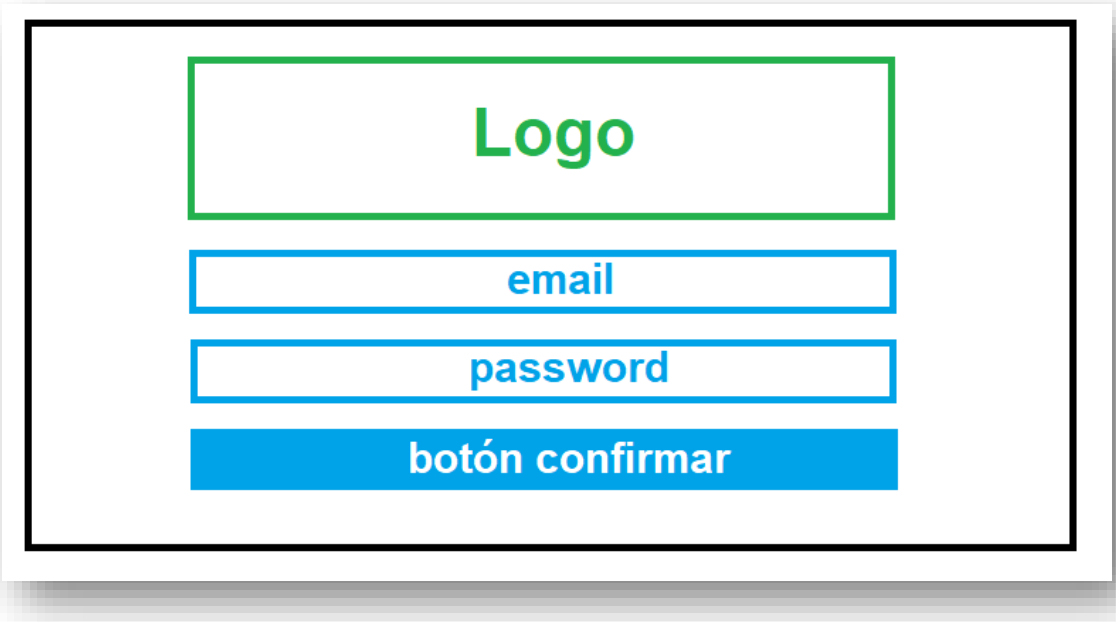


Ilustración 29: Sketch Login

INICIO

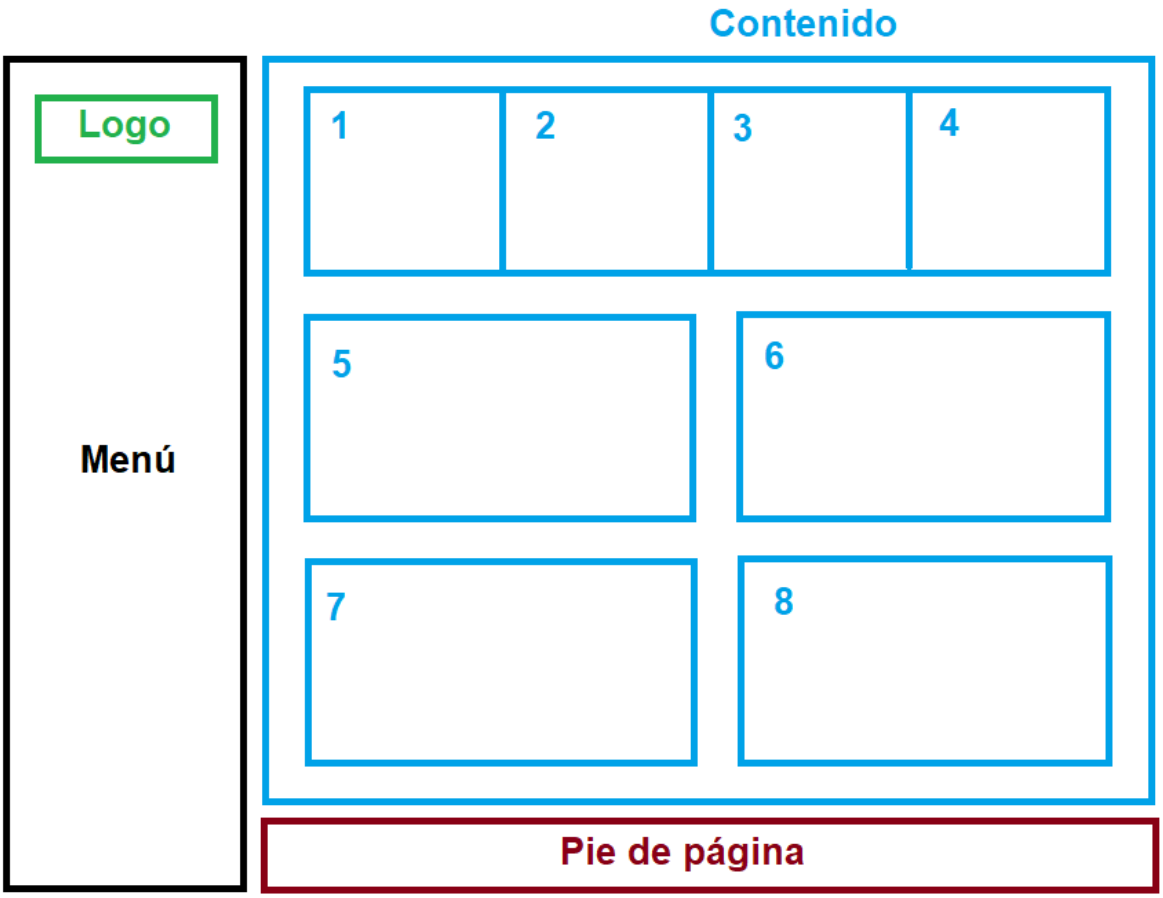


Ilustración 30: Sketch Inicio

- [1 , 4] → Últimos registros almacenados de cada uno de los sensores.
- [5 , 6] → Gráficas de valores temperatura y humedad de los últimos 7 días.
- [7] → Previsión meteorológica de los próximos 4 días a través de la API de AEMET (Agencia Estatal de Meteorología).
- [8] → Tabla con los últimos 7 riegos aplicados a nuestra explotación.

CONTROL

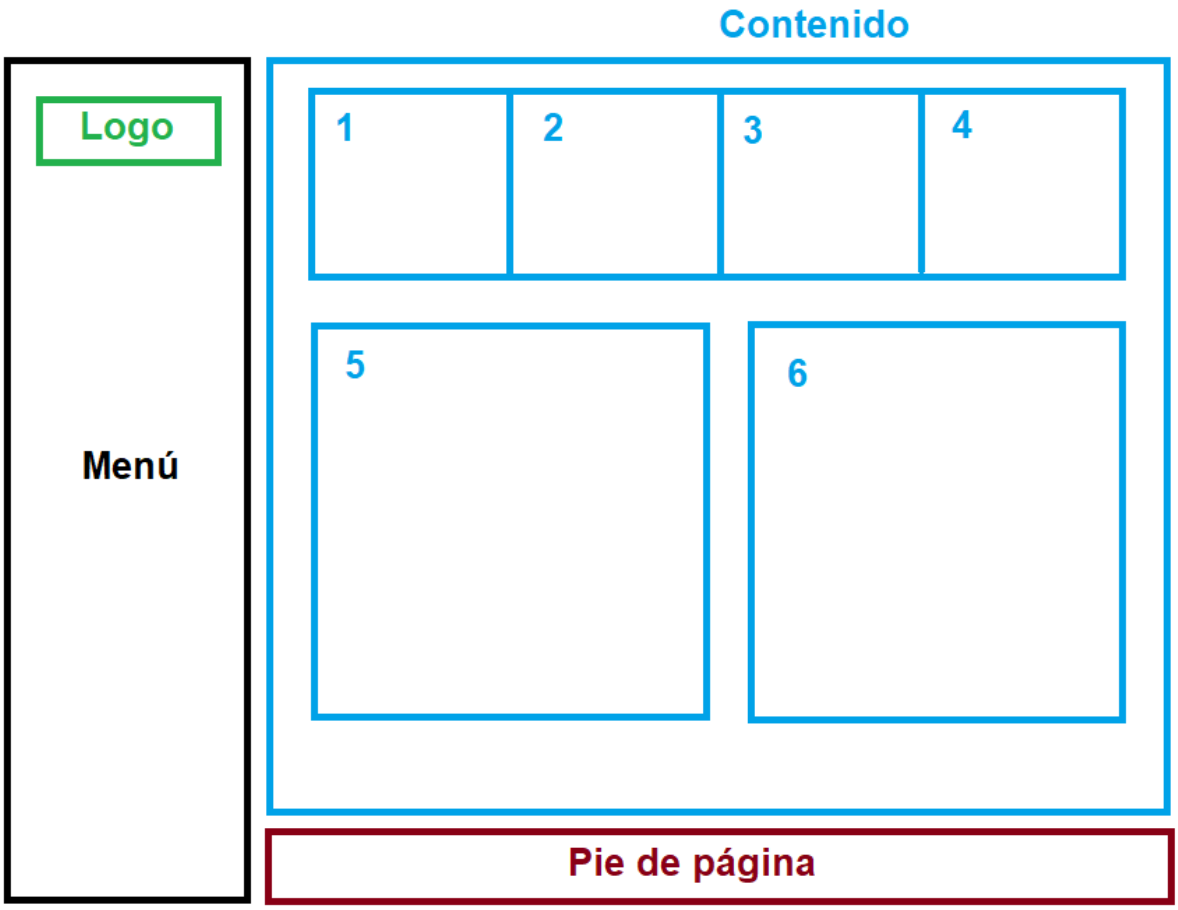


Ilustración 31: Sketch Control

- [1] → Estado Riego (Activado / Desactivado)
- [2] → Riego manual (Activado / Desactivado)
- [3] → Riego programado (Activado / Desactivado)
- [4] → Riego automático (Activado / Desactivado)
- [5] → Configuración de riego programado (Periodo de repetición, día de inicio y duración)
- [6] → Estado de sensores (disponible, error , desactivado)

LOCALIZACIÓN

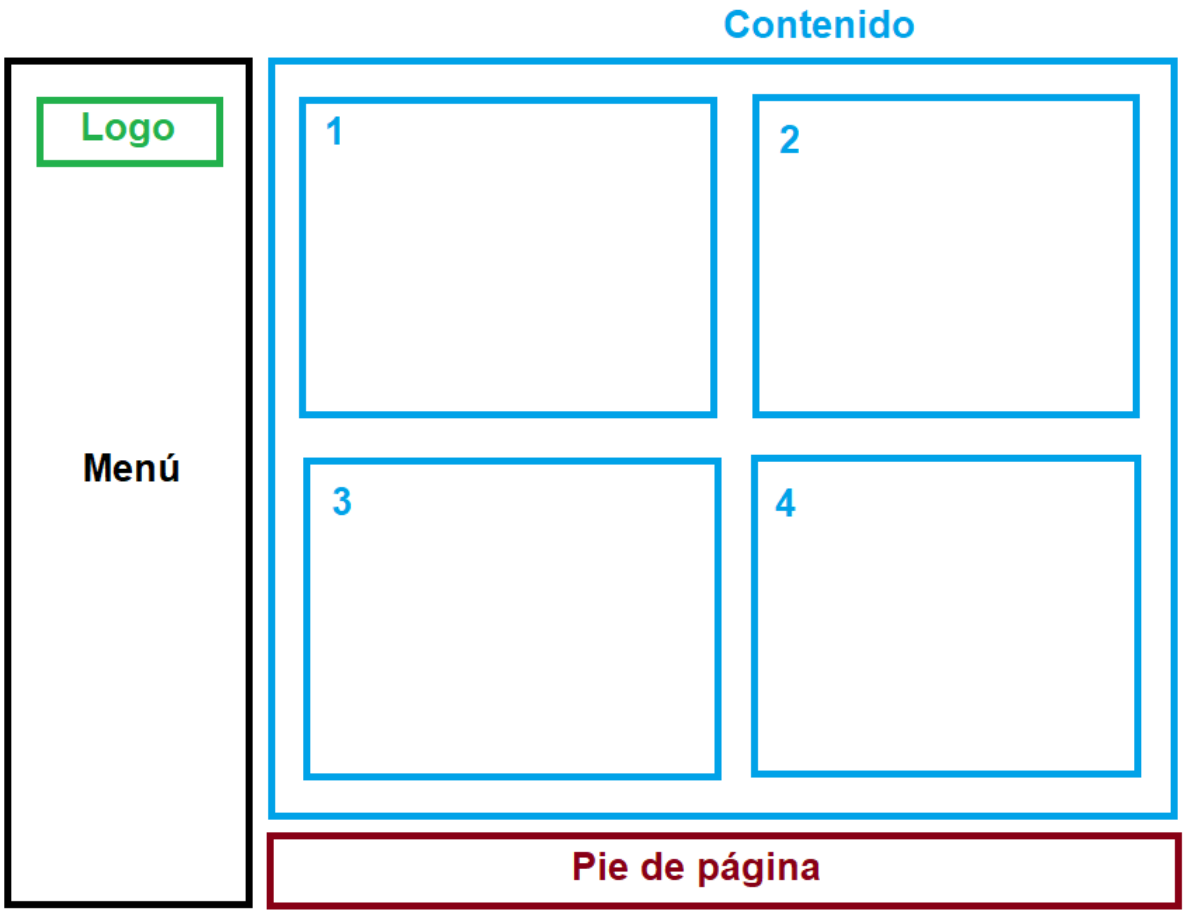


Ilustración 32: Sketch Localización

- [1] → Ubicación de la explotación (Google Maps API).
- [2] → Slider. Con información de la ubicación de la explotación.
- [3] → Enlaces a aplicaciones de mapas y parcelas (Aplicación Google Maps y Sigpac).
- [4] → Slider. Con información del tipo de tierra y esquema de instalación de riego.

PERFIL USUARIO

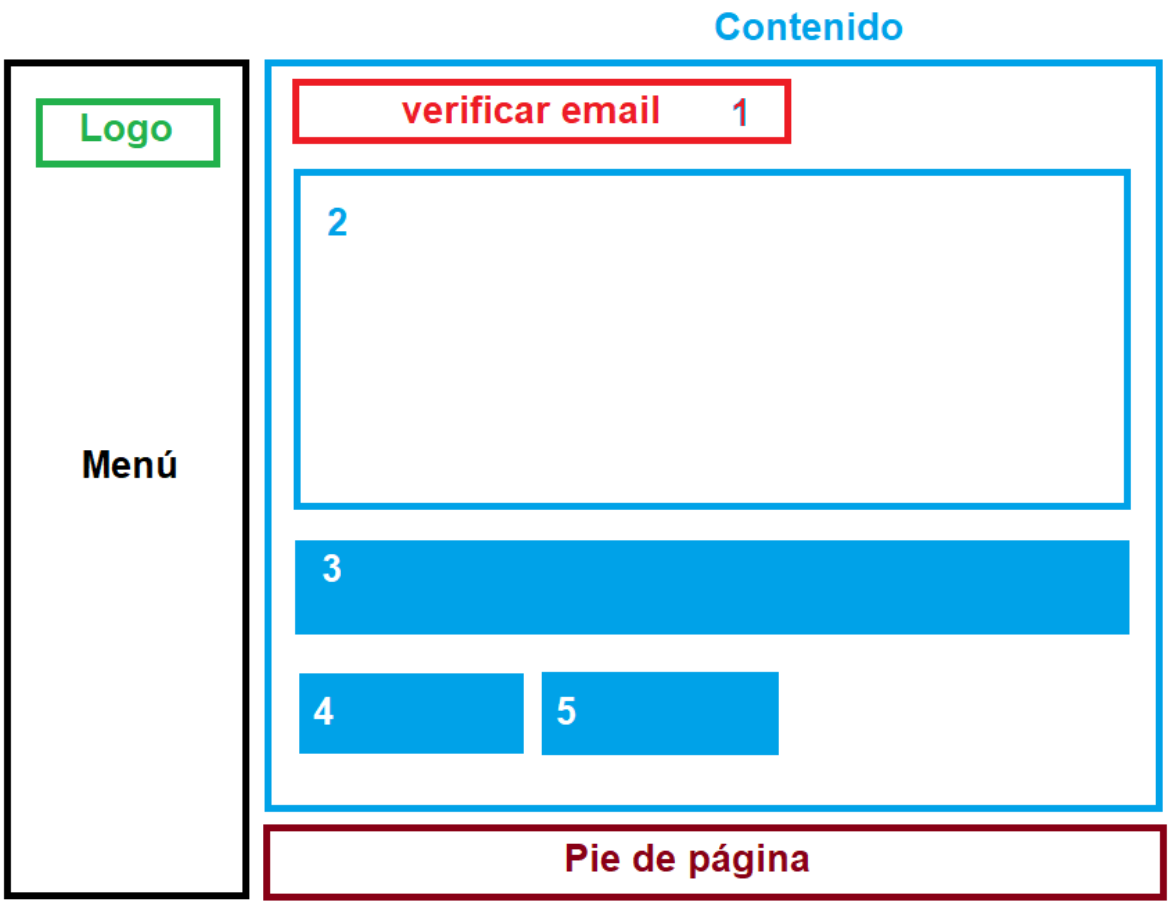


Ilustración 33: Sketch Perfil Usuario

- [1] → Botón para verificar email. Se enviará un correo electrónico para que este botón no aparezca más. El usuario tendrá que acceder a su bandeja de entrada y acceder al enlace del correo que hemos enviado automáticamente.
- [2] → Formulario editable con datos personales del usuario.
- [3] → Botón para cambiar contraseña. Nuestro sistema enviará un correo y el usuario podrá modificar la contraseña de su cuenta.
- [4] → Botón para confirmar los datos del formulario.
- [5] → Botón para restablecer los datos por defecto del formulario.

3.5.4.2 Definición de estilo

En este apartado nos centraremos principalmente en el Front-end de nuestra aplicación. Esta es una de las partes más delicadas e importantes de nuestro proyecto porque es la parte mediante la cual el usuario interaccionará con nuestro sistema.

Nuestro estilo será incorporado a través del framework Bootstrap y contará con un diseño responsive, se adaptará a cualquier tipo de dispositivo y plataforma. Gracias al modelo de rejillas que incorpora este framework hace que la disposición en pantallas horizontales de escritorio tenga similitud con las pantallas móviles o verticales.

Explicaremos la distribución de elementos en pantallas horizontales y verticales de forma gráfica mediante la siguiente ilustración, en la que se puede observar la diferencia entre la versión escritorio y la versión móvil:

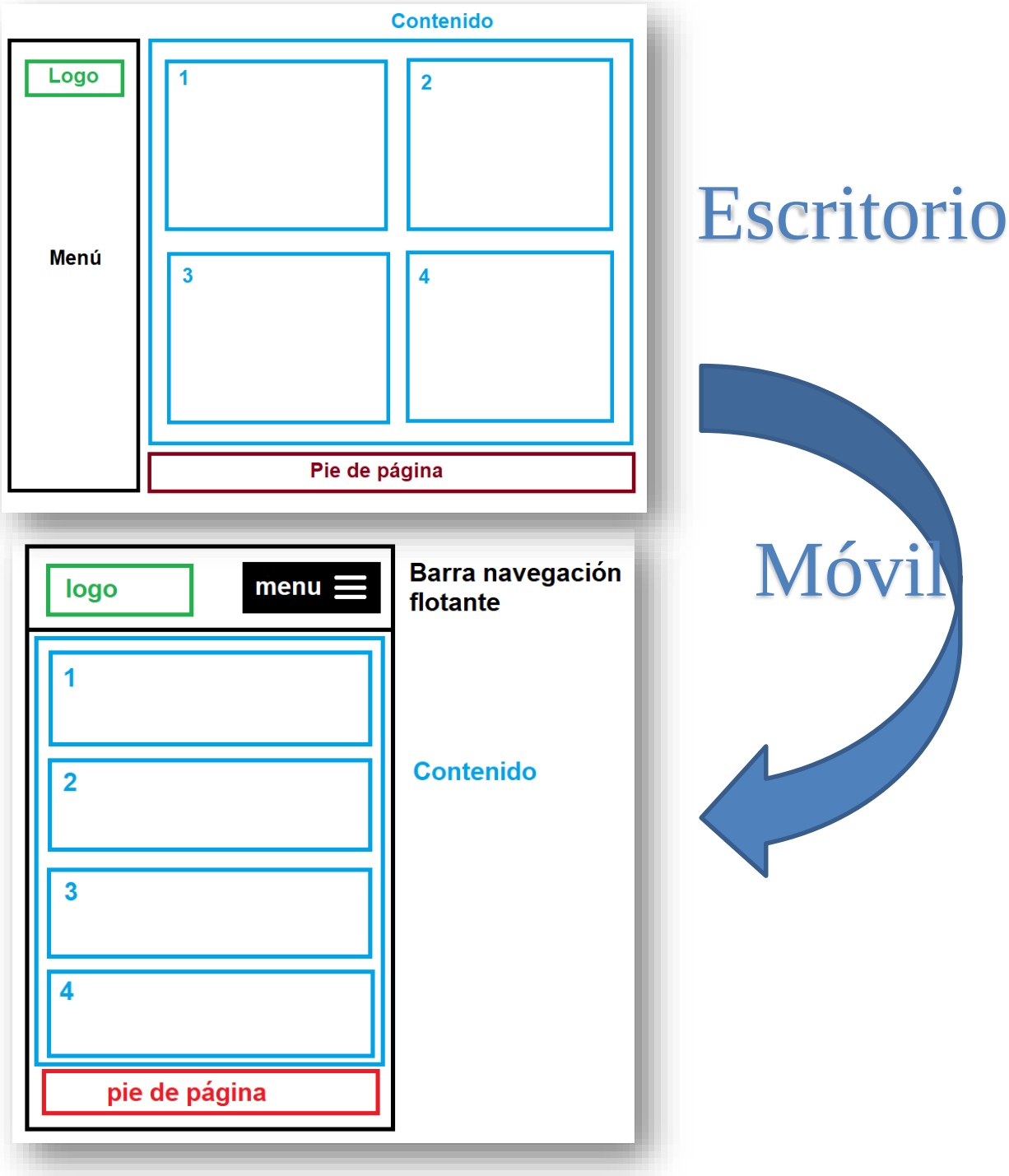


Ilustración 34: Transformación de aplicación escritorio a móvil

Nuestra aplicación en su totalidad se ha utilizado el tono azul pero hemos optado por utilizar un tono más específico como es: # 447ec9

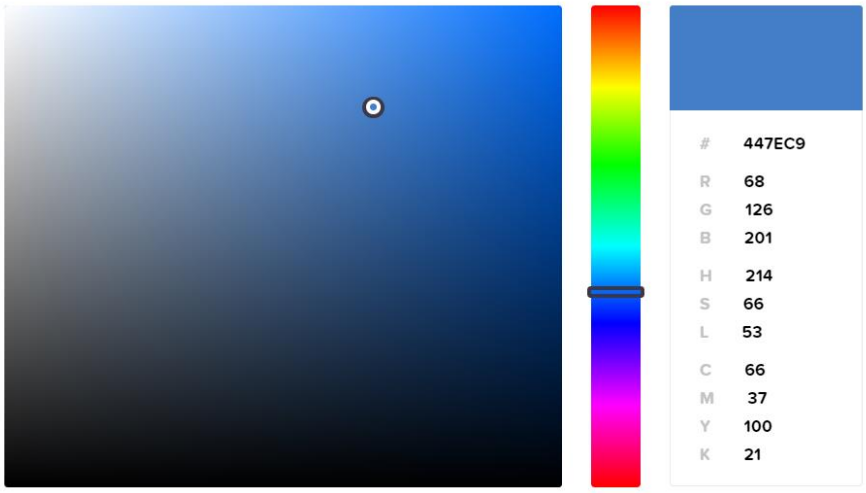


Ilustración 35: Color principal de la aplicación

El segundo tono más utilizado, como se podrá ver en ciertos detalles, los botones están en un tono verde específico: # 7dbc00

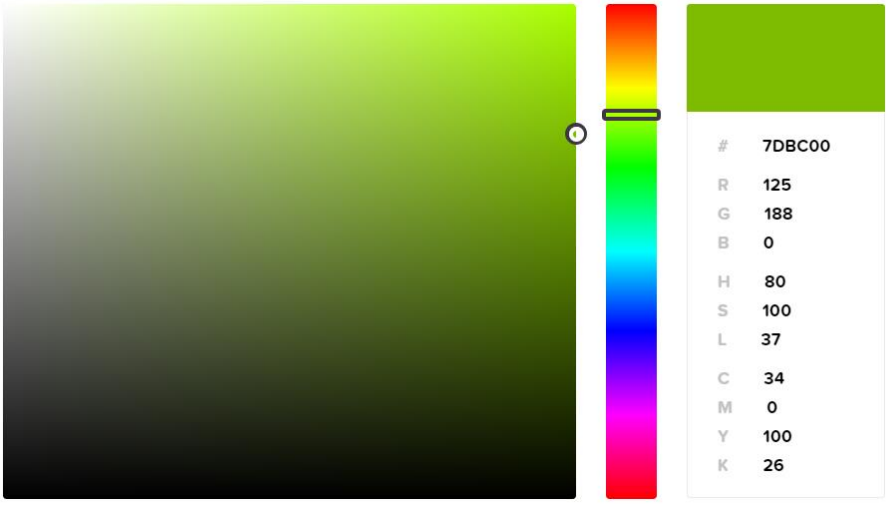


Ilustración 36: Color secundario de la aplicación

3.5.4.3 Storyboard Aplicación

Un storyboard es un conjunto de viñetas relacionadas entre sí. Estas se representan de forma gráfica y representan las diferentes vistas o componentes de nuestro proyecto. Esto nos servirá para mostrar el diseño final que tendrá nuestra aplicación.

Ahora mostraremos el storyboard de nuestra aplicación. Para ello hemos usado la herramienta Marvel que nos permite diseñar a alto nivel diseños avanzados los cuales representan la vista final.

Marvel es una herramienta de realización de prototipos profesional para cualquier plataforma. Adjunto el enlace del storyboard para poder verlo y navegar de forma dinámica.

https://marvelapp.com/66fdg57

1. **Vista Login**: será la primera vista que vera el usuario final. Será necesario que acceda con el email y contraseña de su cuenta, siendo necesaria para poder acceder a nuestra aplicación.

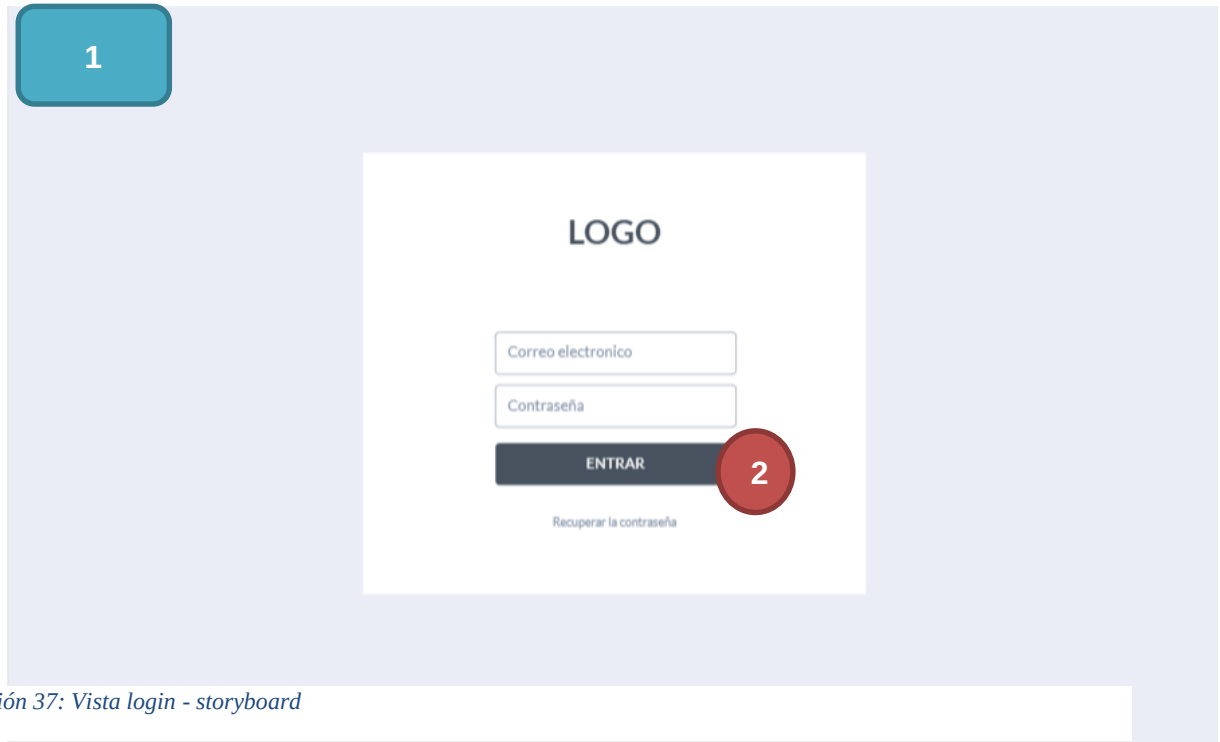


Ilustración 37: Vista login - storyboard

2. **Vista Inicio:** Será la vista por defecto que se cargará cuando un usuario registrado entre a nuestra aplicación.

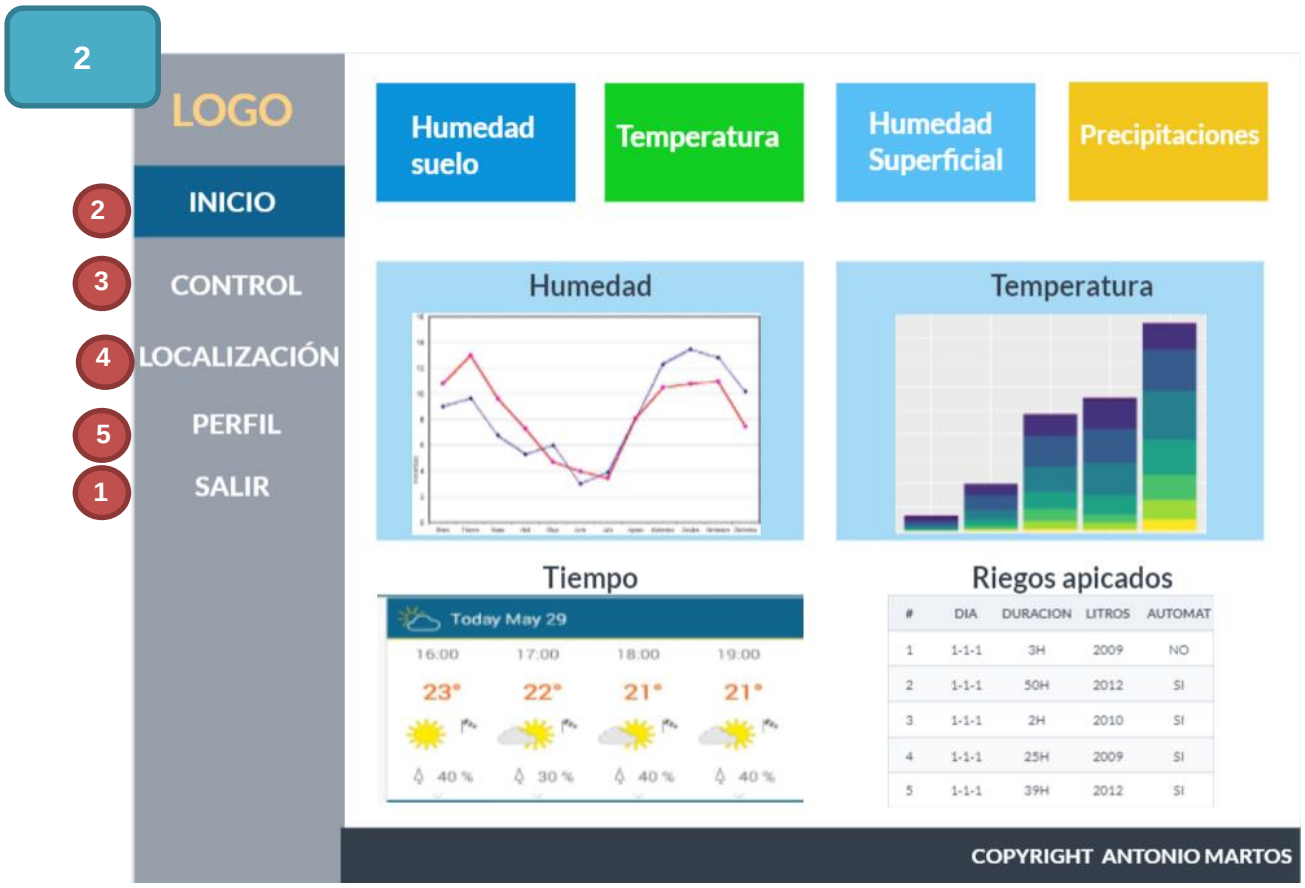


Ilustración 38: Vista Inicio storyboard

3. **Vista Control:** Será la vista mediante la cual el usuario podrá controlar y programar el riego de una manera directa y a tiempo real.

Ilustración 39: Vista Control storyboard

4. **Vista localización**: En esta vista mostraremos la ubicación de la explotación de nuestro usuario y los diferentes datos registrados de esta.

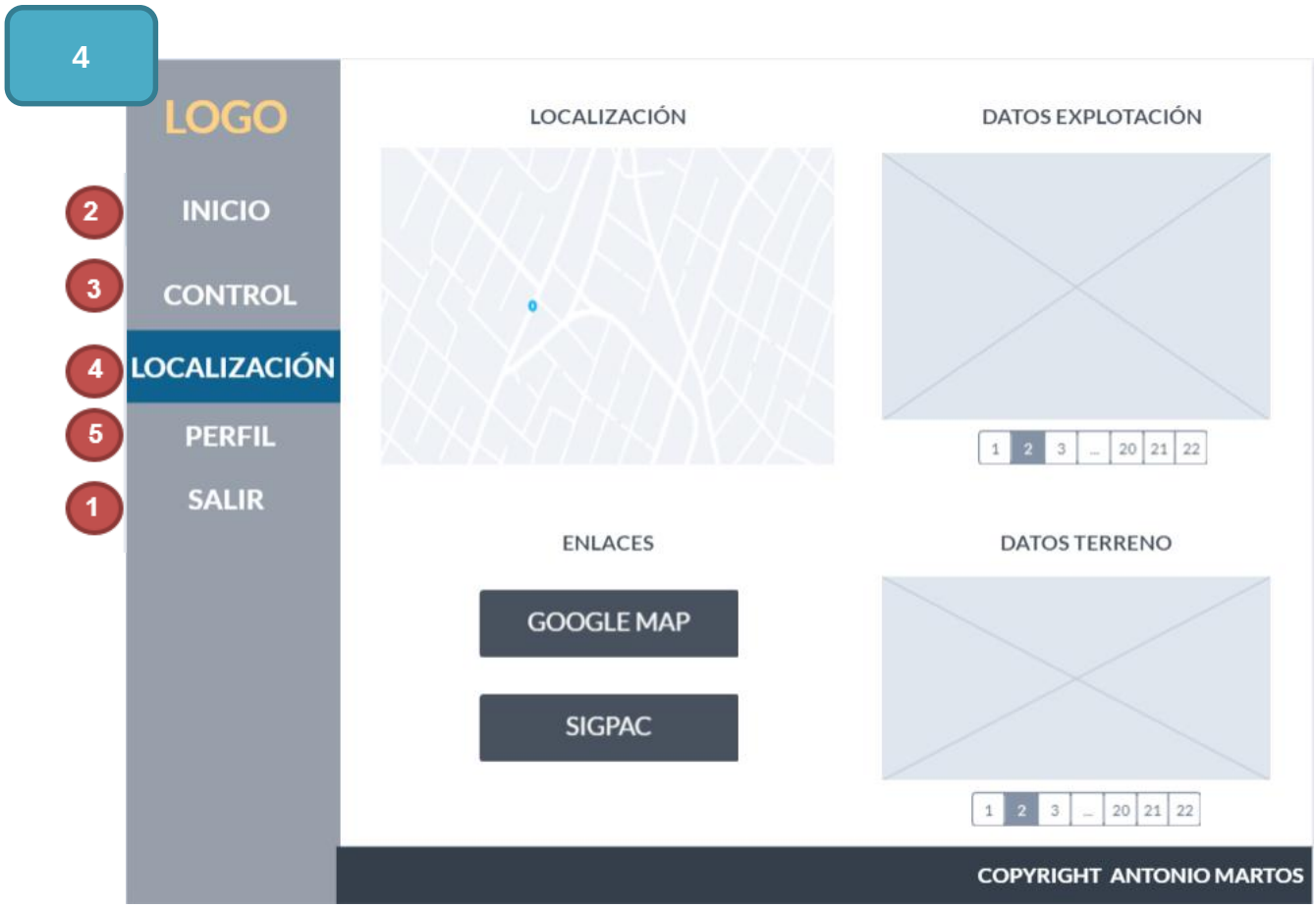


Ilustración 40: Vista Localización storyboard

5. **Vista Cuenta:** En esta vista podremos consultar y modificar los datos personales de nuestro usuario. También podremos cambiar la contraseña de este y aseguraríamos que el usuario haya verificado su email mediante un correo que enviaremos.

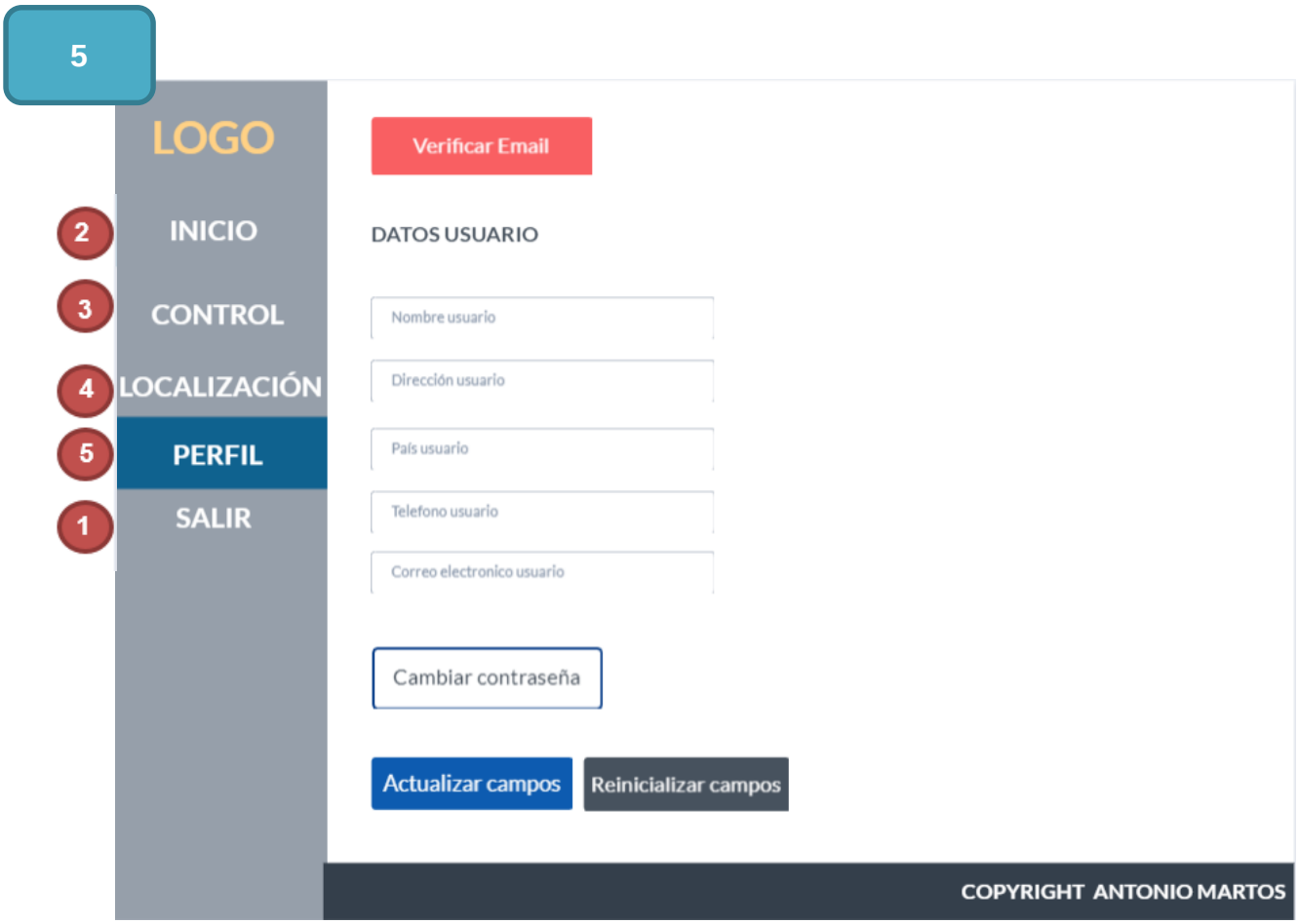


Ilustración 41: Vista Perfil storyboard

Capítulo 4: Tecnologías y desarrollo

En este capítulo describiremos el diseño y la arquitectura tecnología de nuestro proyecto.

4.1 Esquema general

Este proyecto para la automatización está formado en su totalidad por la conexión entre sí de tres plataformas como podremos ver en el siguiente esquema.

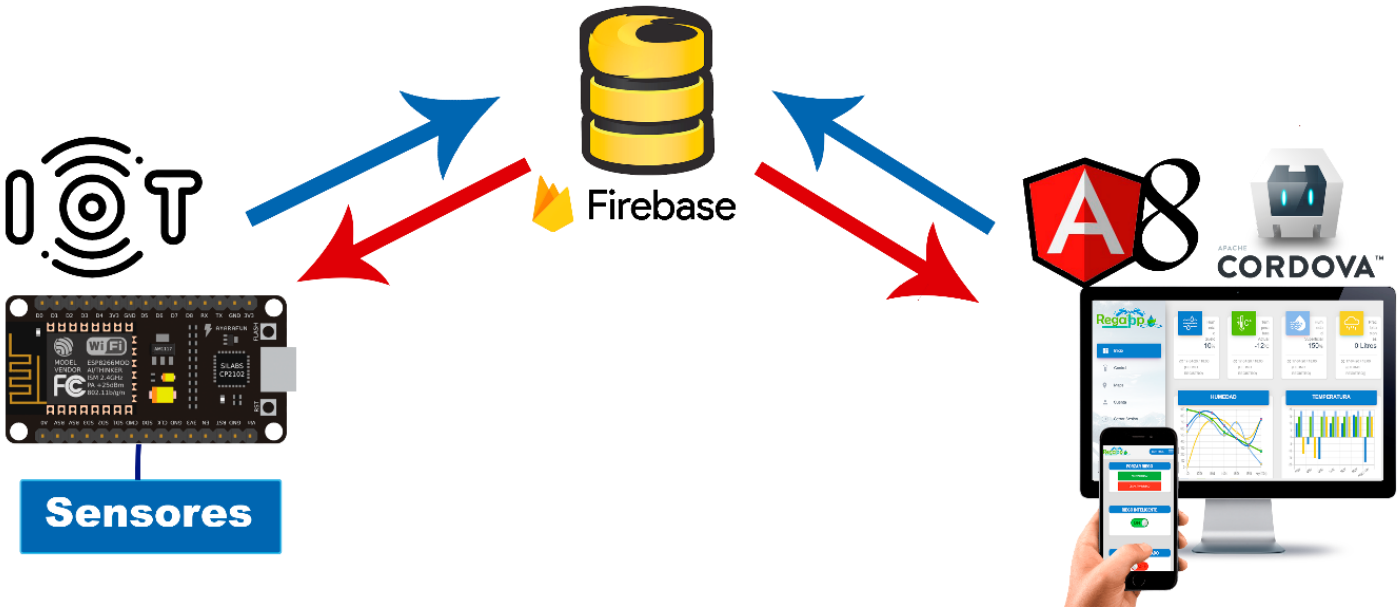


Ilustración 42: Esquema de tecnologías aplicadas

4.2 Plataforma IOT

En primer lugar, nos encontramos con la plataforma IOT, una plataforma indispensable en este proyecto pero que podría funcionar sin ninguna otra plataforma.

En nuestro caso vamos a utilizar Arduino, un controlador-actuador, el cual utilizaremos para recoger valores de sensores para leer y escribir en nuestra base de datos. Esta plataforma podrá activar nuestro riego de diferentes formas que veremos en secciones posteriores. Para programar esta plataforma utilizaremos su IDE nativo, mediante el cual desarrollaremos la implementación en el lenguaje C y C++.

Hemos elegido esta plataforma específica por su bajo coste respecto otras plataformas como podría ser Rapberry. A parte, esta plataforma cuenta con una gran comunidad de desarrolladores y muchas librerías disponibles.

4.2.1 Funcionalidad

- Conectaremos nuestro microcontrolador a la red vía Wifi o GSM.
- Comprobamos a tiempo real si el usuario fuerza el activado o desactivado del riego en nuestra finca mediante la BBDD RealTime.
- Auto chequeo de anomalías en valores obtenidos por los diferentes sensores, pudiendo detectar e informar al usuario final de dicho problema.
- Si todo el chequeo es correcto, procedemos a recoger los valores de dichos sensores.
- Cada 6 horas se añadirán estos registros a nuestra BBDD de Firebase.
- Ejecutaremos nuestro algoritmo el cual determinará si nuestra finca necesita regarse y la duración. Esto se hace teniendo en cuenta los valores de los sensores, los riegos previamente aplicados y la previsión meteorológica de los futuros 4 días.
- Por último, comprobamos si el usuario tiene activado el riego programado, de ser así, procederíamos a aplicar el riego en la finca a las horas y los días previamente indicados.

4.2.2 Implementación y métodos

Veremos los diferentes métodos implementados en nuestro dispositivo IOT Arduino. Con estos métodos tenemos control total de este dispositivo y controlaremos su correcto funcionamiento.

Métodos	Descripción
Boolean EstablecerConexiónWifi(String SSID, String PASS)	Este método nos devuelve TRUE si se ha establecido la conexión wifi correctamente.
Boolean RiegoForzado()	Devuelve TRUE en el caso de que el usuario active el riego de manera manual.
Void ProgramarRiego(Date Inicio ,Int Repeticion, int Duracion)	Configuración del riego programado respecto los parámetros pasados por el usuario.
Boolean RiegoProgramado()	Devuelve TRUE en caso de estar activado y programado el riego programado.
Boolean RiegoInteligente()	Devuelve TRUE en el caso de que este activado el riego inteligente.
Int[] RecogidaValoresSensores()	Devuelve un array de enteros los cuales son los valores de los diferentes sensores.
Int[] ComprobacionValoresSensores(int[] valores)	Devuelve un array de enteros (0 or 1 or -1). • -1, Sensor averiado. • 0, Sensor con valores anómalos. • 1, Sensor funcional.
Boolean EnvioValoresSensores(int[] valores)	Devuelve TRUE si no hay problema alguno en almacenar en la BBDD nuestros valores.
Int Regar()	• -1, no se aplica ningún tipo de riego • 1, se aplica riego manual • 2, se aplica riego programado • 3, se aplica riego inteligente Registra el riego aplicado, duración, aproximación de litros y la fecha de dicho riego en nuestra BBDD.
Void ActivarRiego()	Abre las válvulas de riego.
Void DesactivarRiego()	Cierra las válvulas de riego.

Tabla 8: Métodos implementados en Arduino

4.2.3 Seudocódigo

```
EstablecerConexion_wifi_or_GSM()
modo=false
duracion=0
LOOP{

    SI ACTIVAR RIEGO{
        modo=true;
    }

    SI DESACTIVAR RIEGO{
        modo=false;
    }

    ComprobarAnomaliasSensores();
    RecogerValores_Sensores();

    CADA 6 HORAS{
        AñadirValores_BBDD_Firebase();
    }

    SI ALGORITMO RIEGO INTELIGENTE{
        ActivamosRiego();
        modo=true;

        SI EXCEDE DURACION{
            modo=false;
        }
    }

    SI RIEGO PROGRAMADO{
        ActivamosRiego();
        modo=true

        SI EXCEDE DURACION{
            modo=false;
        }
    }
```

Ilustración 43: Pseudocódigo implementación Arduino

4.2.4 Esquema

Mostraremos la conexión de nuestro NODEMCU y los diferentes sensores utilizados en nuestro sistema Arduino.

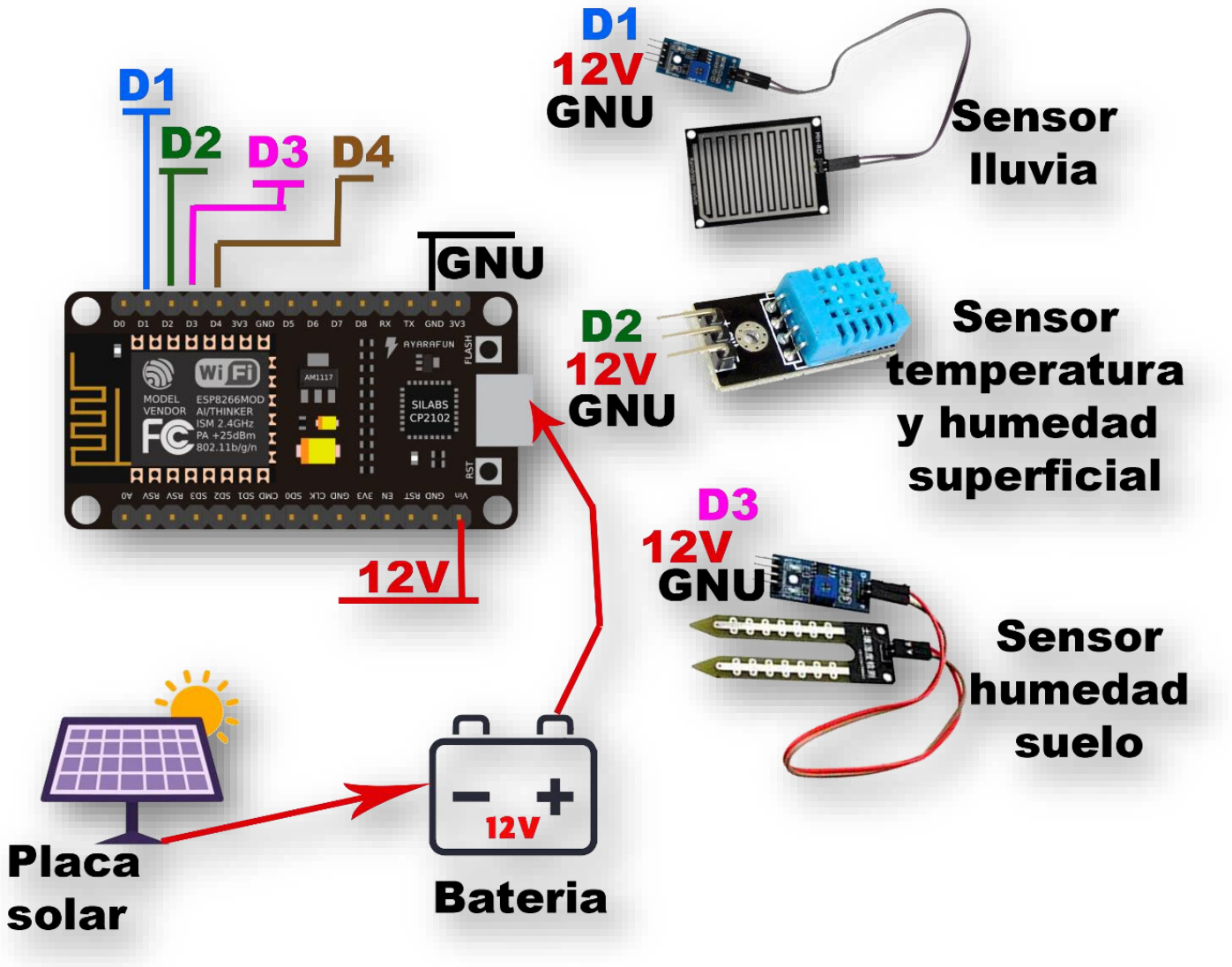


Ilustración 44: Componentes Arduino

4.3 Plataforma Firebase

Es la plataforma que se encuentra de intermediaria. Su función principal es el almacenamiento y lectura de datos.

Firebase es una plataforma móvil creada por Google, cuya principal función es facilitar la creación y gestión de una base de datos. Dicha base de datos cuenta con una alta capacidad y bajo tiempo de respuesta, lo cual la hace muy rápida y sencilla.

Dispone de almacenamiento para plataformas webs. Esta plataforma nos da la posibilidad de utilizar dos tipos de bases de datos:

- FireStore: tiene una estructura visible en la consola más sencilla y fácil de escalar cuando se tiene grandes jerarquías con el manejo de subcolecciones.
- RealTime: almacena los datos en JSON pero se organiza o se representa en la plataforma como un árbol. Esta forma es compleja cuando crecen las anidaciones, las jerarquías o la información.

4.3.1 ¿Por qué elegimos Firebase?

En nuestro caso hemos elegido Firebase porque es una de las plataformas más populares para IOT, proporcionando un almacenamiento en tiempo real (RealTime) idóneo para nuestro proyecto. A la vez es un acceso muy rápido ya que los datos no se encuentran estructurados, sino que podemos mostrar solamente los últimos cambios realizados.

Además de todo esto Firebase al ser una plataforma de Google, no escatima en funcionabilidades añadidas. A parte del almacenamiento utilizamos otra función como podría ser la autentificación multiplataforma que nos ofrece. También contamos con el monitoreo mediante Google Analytics, el cual ofrece esta misma plataforma. Pero esta información solo podrá ser consultada por el Administrador del sistema.

Nos encontramos otras plataformas que ofrecen servicios similares y tienen una alta popularidad entre los diferentes desarrolladores de sistemas IOT.

- Kumulos
- OneSignal
- ThinkSpeak
- TruePush

4.3.1.1 Coste

Empezamos nuestro estudio evaluando el coste de estas diferentes plataformas, intentando agregar el mínimo de gastos extras a nuestro proyecto siempre y cuando este ahorro de gastos no influya en nuestros objetivos ni en su funcionalidad.

Con lo cual en esta primera etapa descartamos tres plataformas por su coste. Nos quedamos con dos plataformas que cumplen todos y cada uno de nuestros requisitos sin añadir ningún tipo de gasto extra a nuestro proyecto, ofreciendo un servicio gratuito y limitado.

4.3.1.2 Limitaciones Firebase - ThinkSpeak

Ambas plataformas nos ofrecen un almacenamiento y aparte una serie de funcionalidades adicionales, como monitoreo de registros y lectura y escritura en BBDD en tiempo real. Su API está disponible para nuestro sistema IOT Arduino (C / C++) y para nuestra plataforma Web/Móvil.

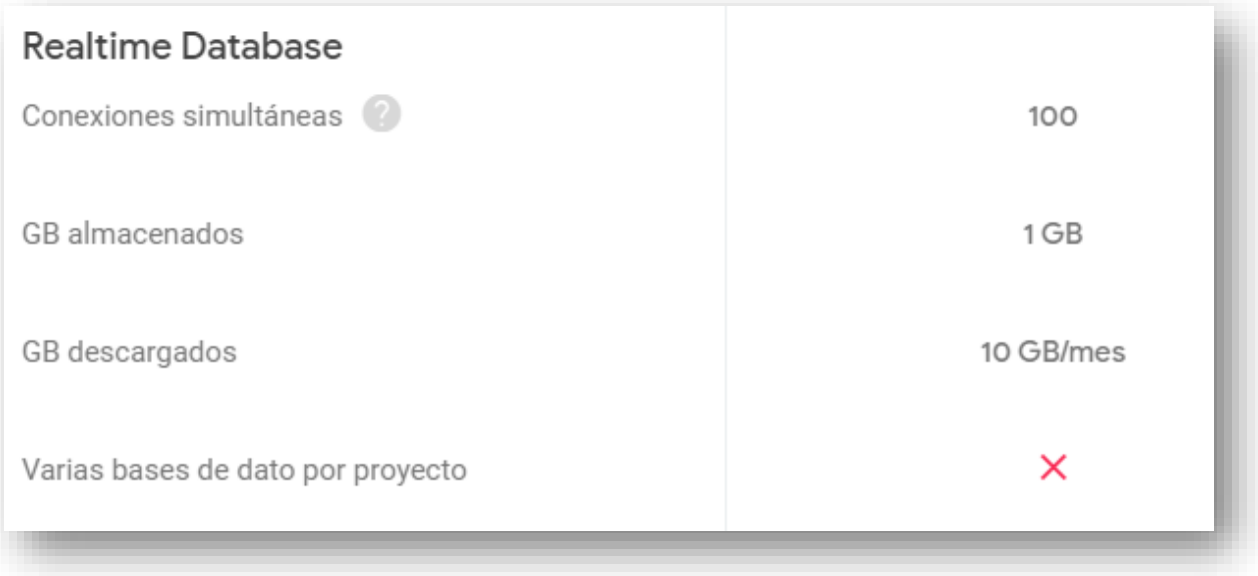
Estas plataformas cubren perfectamente nuestra funcionalidad a simple vista ya que podemos realizar cambios en la BBDD para realizar acciones en nuestro sistema Arduino. También pueden almacenar nuestros registros de los diferentes sensores periódicamente.

Nuestro sistema Arduino necesita una subida periódica mínima de 6h, y nuestra aplicación actuará a tiempo real realizando o consultando datos de nuestra BBDD.

La elección de la plataforma ThinkSpeak plantea varias dificultades. Su versión gratuita tiene una limitación de una petición al servidor cada 15 segundos, esta limitación, puede ralentizar la función de control de nuestro sistema.

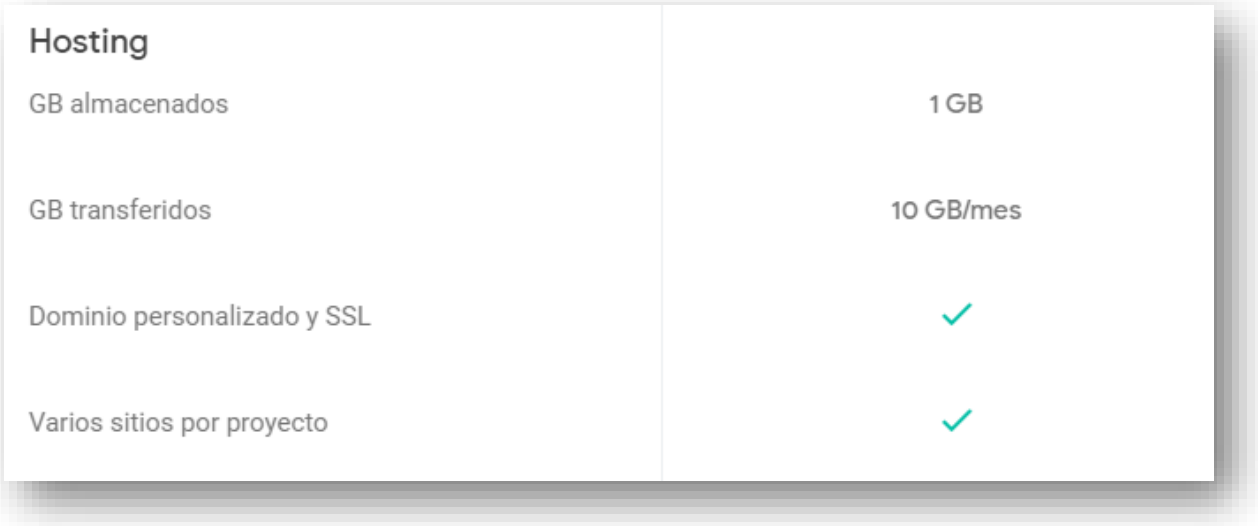
Una vez descartada la plataforma ThinkSpeak, analizamos en profundidad la plataforma de Google, Firebase.

Estas son las limitaciones que tiene la versión gratuita de Firebase respecto almacenamiento:



Realtime Database	
Conexiones simultáneas	100
GB almacenados	1 GB
GB descargados	10 GB/mes
Varias bases de dato por proyecto	✗

Ilustración 45: Limitaciones RealTime Firebase



Hosting	
GB almacenados	1 GB
GB transferidos	10 GB/mes
Dominio personalizado y SSL	✓
Varios sitios por proyecto	✓

Ilustración 46: Limitación Hosting Firebase

4.3.1.3 Autentificación

Nuestra aplicación utiliza el método de autentificación mediante email y contraseña que administra Firebase, en la que tenemos la posibilidad de ampliar en un futuro mediante redes sociales como Facebook entre otras. También podemos añadir login a través de correo electrónico Google.

Para ello nos ofrece un servicio multiplataforma el cual abre una ventana modal para realizar el logueo del usuario.

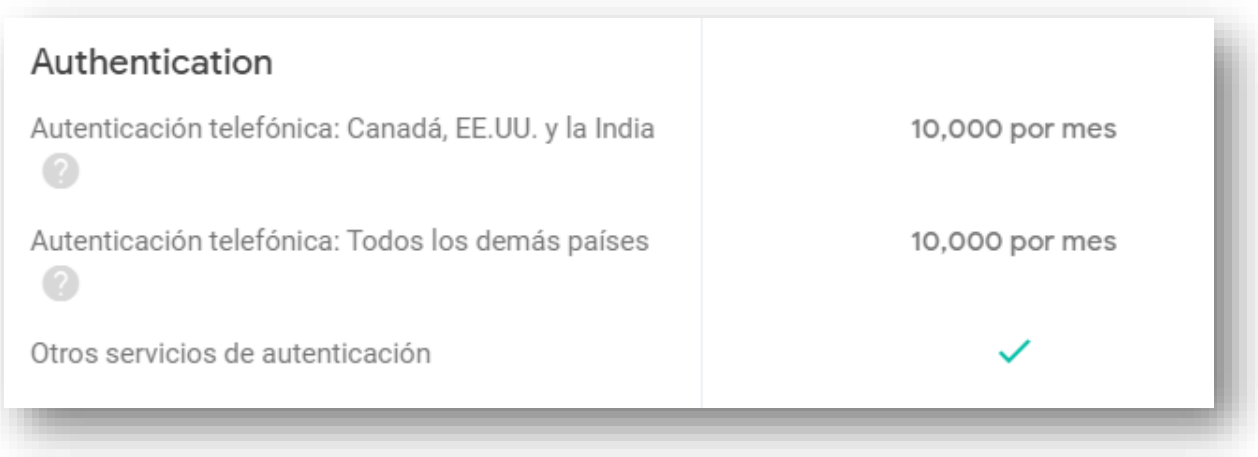


Ilustración 47:Limitación autentificación Firebase

4.4 Plataforma Angular 8

Es un framework para desarrollar aplicaciones en formato web, mediante un lenguaje de programación hibrido como es TypeScript, mantenido por Google y de código abierto. Se forma de componentes los cuales están formados a su vez por diferentes ficheros.

4.4.1 Definición y funcionalidad

Esta plataforma se basa en el conocido Modelo Vista Controlador (MVC)

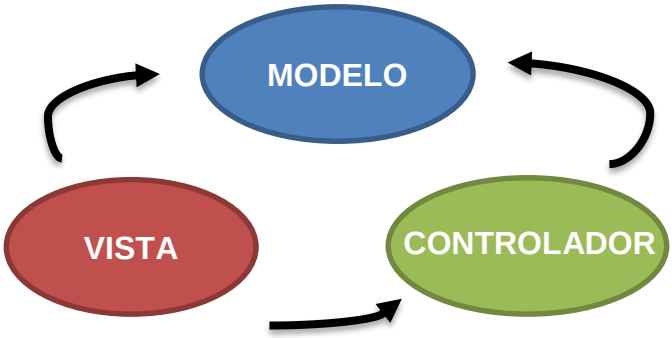


Ilustración 48: Esquema Modelo vista controlador

Angular se basa en clases tipo "componentes" formadas por diferentes ficheros cuyas propiedades son las usadas para hacer uso de los datos. En dichas clases tenemos propiedades “variables” y métodos.

Los servicios es una funcionabilidad indispensable en Angular para compartir datos y funciones entre los diferentes componentes.

Otra peculiaridad que nos encontramos frente a otros framework son las funciones propias de cada componente:

- Constructor() – Se ejecuta al realizar la primera llamada del componente.
- OnInit() – Se ejecuta después del constructor pero al cargar el componente.
- AfterView() – Se ejecuta una vez todo el contenido este cargado en nuestra web.

4.4.2 Esquema

Angular utiliza una estructura en árbol que hace posible las llamadas de un componente dentro de otro componente, quedando este segundo como hijo del primero y así progresivamente. Nuestra aplicación se basa en 5 componentes principales que serían las principales vistas. Cada uno de estos componentes en su creación llama a la creación de otros componentes como podrían ser el menú, la barra de navegación y el pie de página.

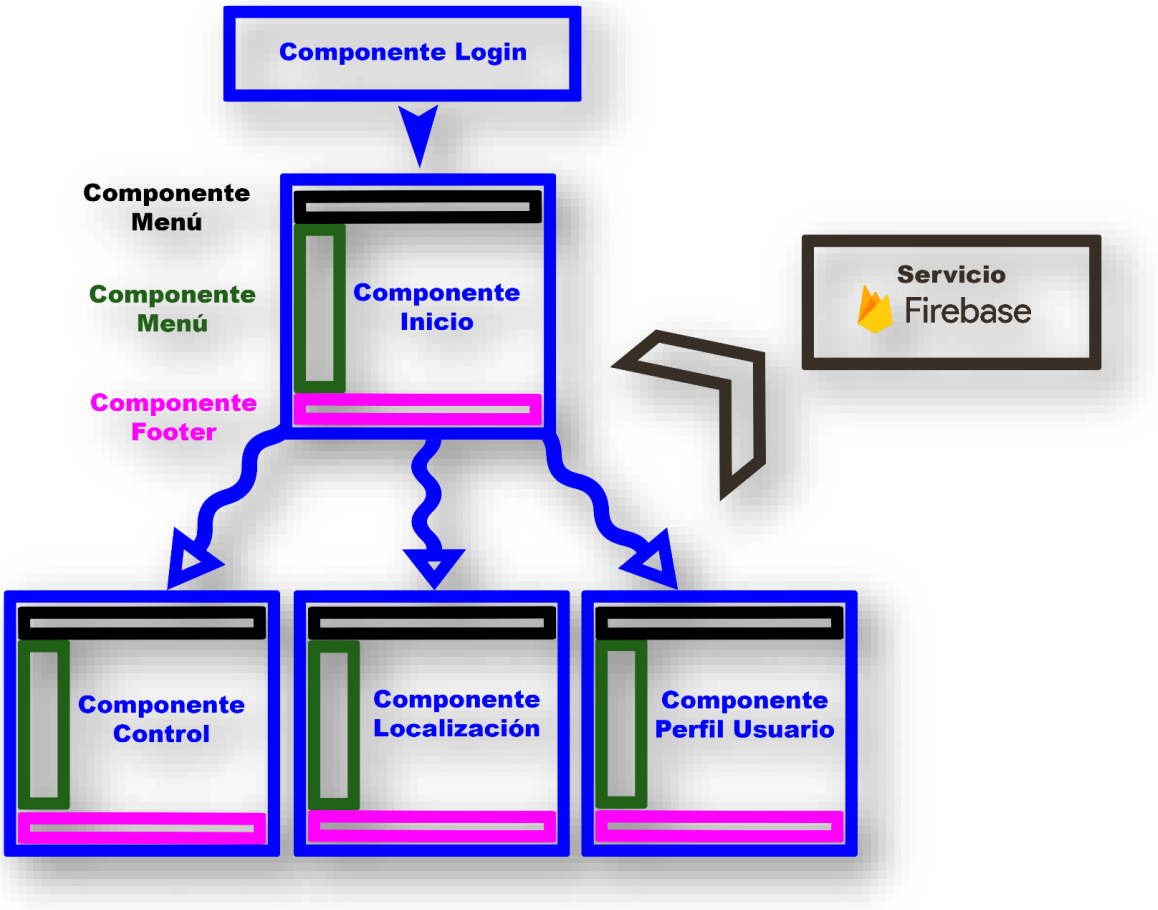


Ilustración 49: Esquema componentes aplicación Angular
Ilustración 50: Esquema de componentes de nuestra aplicación

4.4.3 Servicios, implementación y métodos por componentes

Veremos los diferentes servicios que utiliza nuestra aplicación, así como, la implementación y métodos de cada uno de los diferentes componentes con los que contamos.

4.4.3.1 Servicio Firebase2

Servicio Firebase2	
Void Auth(String Email, String Pass)	Este método de nuestro servicio será el encargado de loguear un usuario siempre y cuando sus credenciales sean las correctas. En caso de no ser correcto lanzará una excepción controlada.
Boolean Session()	Devuelve True en caso de que el usuario este correctamente logueado y la sesión de usuario esté disponible.

Tabla 9:Métodos servicio Firebase

4.4.3.2 Implementación por componentes

Componente Login	
Bolean ValidacionCampos()	Devuelve TRUE tras la validación de campos del formulario de login.
Void login()	Este método llama a nuestro servicio de autentificación de Firebase.

Tabla 10: Métodos componente login

Componente Inicio	
Int[] ObtenerValoresSensores()	Devuelve un array con los valores registrados por cada sensor. En caso de recibir -1 indicará que dicho sensor está averiado o desconectado y si devuelve -2 indicará que dicho sensor devuelve valores anómalos.
Void CargarGraficaHumedad()	Recoge los datos almacenados en nuestra BBDD de los últimos 7 días respecto la humedad de nuestra explotación para la cargar la gráfica.
Void CargarGraficaTemperatura()	Recoge los datos almacenados en nuestra BBDD de los últimos 7 días respecto la temperatura de nuestra explotación para la cargar la gráfica.
Void CargarUltimosRiegos()	Recoge los datos almacenados en nuestra BBDD de los últimos 7 riegos aplicados a nuestra explotación

Tabla 11: Métodos componente inicio

Componente Control	
Boolean RiegoActivo()	Devuelve TRUE en caso de que nuestro sistema este activo regando nuestra explotación.
Boolean RiegoInteligente()	Devuelve TRUE en caso de que el usuario haya activado previamente dicho tipo de riego y modificará la BBDD para indicarlo.
Boolean RiegoProgramado()	Devuelve TRUE en caso de que el usuario haya activado previamente dicho tipo de riego. En este caso se recogerán los datos introducidos por el usuario para programar el riego y almacenaremos estos parámetros en nuestra BBDD.
Boolean RiegoForzado()	Devolverá TRUE en el caso que el usuario active el riego manual, modificando dicho parámetro en nuestra BBDD.
Void CargarEstadoSistema()	Recogerá los valores almacenados en nuestra BBDD, pudiendo mostrar así, estos de forma gráfica en nuestra tabla del sistema para saber su estado.

Tabla 12: Métodos componente control

Componente Localización	
Void CargarSliders()	Método que recogerá los datos e imágenes de nuestra BBDD para cargar los 2 slider de nuestra aplicación.
Void CargaMapa()	Método que recogerá la dirección de la explotación para cargar el mapa de la API de Google Maps.

Tabla 13: Métodos componente localización

Componente Perfil	
Bolean VerificaciónEmail()	Devuelve TRUE en el caso que el usuario haya verificado su Email.
Void CargaDatosUsuario()	Recogerá los datos del usuario almacenados en nuestra BBDD mostrándolos en el formulario.
Void ActualizarCampos()	Validará los datos del formulario y actualizará dichos datos en nuestra BBDD
Void ReinicializarCampos()	Restablecerá los últimos datos almacenados en nuestra BBDD.
Void CambiarContrasenia()	Enviará un correo el servicio de Firebase a nuestro usuario, mediante un pequeño formulario que podrá actualizar la contraseña de su cuenta.

Tabla 14: Métodos componente perfil

4.4.3 Plataformas soportadas

Una vez finalizada la implementación del código en la plataforma Angular 8, este formato del código es para plataforma web. Nos bastaría con subir el contenido de nuestro proyecto Angular a cualquier hosting.

Uno de los requisitos que nos impusimos a la hora de desarrollar este proyecto, fue que el usuario final pudiese acceder a una aplicación multiplataforma para no tener ningún tipo de restricción a la hora del acceso.

Con lo cual este requisito previo nos dio a estudiar alternativas de exportación de dicha implementación.

4.4.3.1 Apache Cordova (anterior PhoneGap)

Es un marco de desarrollo de aplicaciones móviles, con el cual podremos generar los ejecutables para cada plataforma móvil existente, en nuestro caso Android. Este marco de desarrollo es compatible con toda nuestra plataforma Angular ya que utiliza:

- HTML5
- CSS3
- JAVASCRIPT

Aquí nos encontramos un problema, porque los componentes de Angular se programan en TypeScript y aunque es un lenguaje similar, no es exactamente igual, por lo que tuvimos que realizar una refactorización de código completa.

Este inconveniente nos produjo un retraso considerable en el desarrollo de este proyecto, pero por lo demás nos encontramos con una plataforma robusta, con miles de desarrolladores que forman una comunidad. Se encuentra mucho soporte frente a problemas y a la vez disponemos de múltiples librerías específicas para Cordova.

Se analizaron otros marcos de desarrollo móvil populares como podría ser Ionic que funciona internamente con librerías de Cordova y funcionalidades de PhoneGap. Así que, nos quedamos con Cordova por su facilidad y gran comunidad de desarrolladores.

4.4.3.2 Plataformas soportadas

Gracias a este marco de desarrollo podemos compilar nuestro código para generar un instalador diferente para cada plataforma.

- Android desde la versión 4.4, formato APK
- IOS todas las versiones, formato IPA
- Windows 7/8/8.1/10, formato EXE

Capítulo 5: Pruebas y validación

En este capítulo realizaremos las pruebas de nuestro proyecto. Lo dividiremos en los diferentes procesos y plataformas con las que contamos.

5.1 Arduino

Esta plataforma es la que más pruebas hemos tenido que realizar porque es la encargada de gestionar la funcionalidad final de nuestra aplicación. No pudiendo permitirnos ningún error.

Arduino tiene diferentes funciones las cuales hemos tenido que revisar y probar minuciosamente. Después de realizar una serie de pruebas funcionales nos hemos dado cuenta de una serie de problemas que harían que nuestro sistema pudiese llegar a bloquearse, o en su defecto acarrear dichos problemas a nuestra BBDD, lo cual afectaría a nuestra aplicación final.

A continuación, mostraremos los problemas detectados y la solución aplicada a estos para un correcto funcionamiento de nuestra plataforma IOT.

- **Apagado repentino**: La falta de alimentación es uno de los principales problemas que nos podemos encontrar. Para solucionar este problema nos aseguramos que al conectar este dispositivo vuelva a gestionar su conexión a internet, comprobará el último estado del riego registrado en nuestra base de datos y por último, realizará una revisión para asegurar que los sensores están correctamente.
- **Problemas de conexión a internet:** El problema que se nos presentaría sería la falta de registros de nuestros sensores en nuestra base de datos. Para solventar este problema, nuestro dispositivo almacenaría en memoria los valores registrados por nuestros sensores y cuando se solucione el problema de conexión almacenará automáticamente dichos registros.
- **Problemas en sensores:** Pueden presentarse problemas de valores anómalos tomados por nuestros sensores, o bien presentar problemas de conexión de estos. Para ello tenemos una función en Arduino la cual detectará un mal funcionamiento de estos y la cual nos devolvería el estado en el que se

encuentra cada uno de estos sensores. Esta función se utiliza previamente a la toma de valores de los sensores. Contamos con 3 estados:

  - OK → El sensor esta funcional.
  - ERROR → El sensor no se detecta o esta averiado.
  - WARNING → El sensor contempla valores anómalos.

- **Problema configuración de hora**: Que la hora este correctamente configurada en nuestro dispositivo es esencial porque trabajamos con ella para aplicar los diferentes riegos programados. También la necesitamos para la escritura en nuestra BBDD de registros tomados por los sensores en una fecha y hora concretas. Para solucionar este problema utilizamos un servidor NTP de Google. Cuando configuramos la conexión de nuestro dispositivo automáticamente enviará una petición a dicho servidor y este nos responderá con la fecha y hora actual.

5.1.2 Pruebas unitarias

Cada uno de los siguientes métodos ha sido testeado unitariamente tras su desarrollo, pasando las diferentes pruebas realizadas:

- Validez de los parámetros introducidos, comprobando y estableciendo límites inferiores y superiores en caso que fuese necesario.
- Paginación en memoria, establecimiento de reglas y limitaciones respecto la lectura/escritura en memoria. Para evitar el acceso incorrecto a diferentes posiciones de memoria.
- Control de excepciones, comprobación de problemas con servicios o la conexión.

Métodos	
Boolean EstablecerConexiónWifi(String SSID, String PASS)	
Boolean RiegoForzado()	
Void ProgramarRiego(Date Inicio ,Int Repeticion, int Duracion)	
Boolean RiegoProgramado()	
Boolean RiegoInteligente()	
Int[] RecogidaValoresSensores()	
Int[] ComprobacionValoresSensores(int[] valores)	
Boolean EnvioValoresSensores(int[] valores)	
Int Regar()	
Void ActivarRiego()	
Void DesactivarRiego()	

Tabla 15: Pruebas de métodos plataforma Arduino

5.2 Base de datos

Como comentamos en el capítulo 4, sección 4.3 utilizamos Firebase, esta plataforma de Google no solamente nos ofrece el almacenamiento a tiempo real sino que nos proporciona el método de autentificación de usuario, con lo cual estamos cubiertos y exentos de realizar pruebas en el login del usuario y en lecturas/escrituras en la BBDD.

Una de las pruebas que nos interesaba conocer es la respuesta del servidor Firebase. Para evitar que nuestra aplicación se ralentice a la hora de cargar los diferentes datos almacenados.

Encontramos un problema respecto el tiempo de respuesta de Firebase. Este problema es referente a la carga de las diferentes gráficas con las que contamos en nuestra aplicación final. Estas gráficas cada vez que navegamos entre vistas volvían a notarse un ligero retardo en la carga de dichos datos. Esto es referente a la consulta de la BBDD, con lo cual, para solucionar este problema al inicio de la aplicación realizamos una única consulta a la BBDD y así, almacenaremos en la memoria cache de nuestra aplicación para evitar dicho retardo.

5.3 Aplicación Final

Nuestra aplicación final también ha pasado una serie de pruebas, en las cuales hemos detectado problemas y hemos buscado soluciones para estos.

Realmente nuestra aplicación a parte de nuestro componente de control de riego cuenta con poca funcionalidad porque solamente mostraría los diferentes datos almacenados en nuestra BBDD. Así que, nos hemos centrado en realizar todas las pruebas a este componente de control del riego.

Dividiremos la diferente funcionabilidad de este componente el cual ha sido probado y testeado a fondo. Este componente es el más importante a la hora de la funcionalidad del usuario.

- **Monitoreo estado del riego**: Nos muestra el estado en el que se encuentra la llave de paso del agua, este estado puede ser On/Off, nos indicara si nuestro sistema está regando o no nuestra explotación. Al ser solamente funcionalidad de monitoreo no encontramos ningún problema.
- **Riego manual** (Activado / Desactivado): En esta funcionalidad si encontramos un problema de usabilidad, la cual si un usuario fuerza nuestro sistema a regar y automáticamente cierra la aplicación por el motivo que sea, podría olvidar que dicho riego se quedó activado y podríamos tener problemas con nuestra explotación. Con lo cual establecimos un límite máximo de horas de riego para evitar este problema, limitando así, el riego forzado a 4 horas. Automáticamente se desactivará el riego manual tras pasar este límite.
- **Riego programado**: podremos activar o desactivar esta función cuando queramos y configurarla a nuestro gusto. No hemos encontrado ningún problema en esta funcionalidad, pero sí en la configuración de dicho riego que comentaremos en el siguiente punto.
- **Configuración de riego programado** (Periodo de repetición, día de inicio y duración): Con esta funcionalidad no hemos encontrado problemas de validación. Estos problemas ocasionaban que nuestro riego programado nunca llegase a ejecutarse. Para solucionar este problema, necesitamos validar ciertos campos previamente mencionados. El período de repetición está

previamente definido con diferentes opciones. El día de inicio no puede ser inferior al día actual por lo que necesitamos establecer la fecha y hora actual a través de un servidor NTP como hacíamos en la plataforma Arduino. Y, por último, la duración la limitamos como mínimo a 30 minutos y como máximo a 12 horas.

- **Riego automático:** Podremos activar o desactivar esta funcionalidad que realiza un riego automático respecto los valores registrados por nuestros sensores. Esta funcionalidad presentaba un problema bastante grave, ya que tras una serie de pruebas un sensor puede dar valores anómalos que hacía que nuestro sistema estuviese en un riego constante. Para solucionar este problema hemos forzado a que solamente esté disponible cuando todos nuestros sensores estén correctamente funcionando. En el caso de que alguno presente problemas del tipo que sea, se bloqueará esta funcionalidad, no dejando al usuario activar el riego automático y en el caso de estar activado se desactivará hasta que se solucione el problema con los sensores.

5.3.1 Pruebas por componentes

Cada uno de los siguientes métodos ha sido testeado durante su desarrollo y tras su desarrollo, pasando las diferentes pruebas realizadas:

- Validez de los parámetros introducidos, comprobando y estableciendo límites inferiores y superiores en caso que fuese necesario.
- Paginación en memoria, establecimiento de reglas y limitaciones respecto la lectura/escritura en memoria. Para evitar el acceso incorrecto a diferentes posiciones de memoria.
- Control de excepciones, comprobación de problemas con servicios o la conexión.

Servicio Firebase2	
Void Auth(String Email, String Pass)	
Boolean Session()	

Tabla 16: Pruebas métodos servicio Firebase

Componente Login	
Bolean ValidacionCampos()	
Void login()	

Tabla 17: Pruebas métodos componente login

Componente Inicio	
Int[] ObtenerValoresSensores()	
Void CargarGraficaHumedad()	
Void CargarGraficaTemperatura()	
Void CargarUltimosRiegos()	

Tabla 18: Pruebas métodos componente inicio

Componente Control	
Boolean RiegoActivo()	
Boolean RiegoInteligente()	
Boolean RiegoProgramado()	
Boolean RiegoForzado()	
Void CargarEstadoSistema()	

Tabla 19: Pruebas métodos componente control

Componente Localización	
Void CargarSliders()	
Void CargaMapa()	

Tabla 20: Pruebas métodos componente localización

Componente Perfil	
Bolean VerificaciónEmail()	
Void CargaDatosUsuario()	
Void ActualizarCampos()	
Void ReinicializarCampos()	
Void CambiarContrasenia()	

Tabla 21: Pruebas métodos componente perfil

5.4 Pruebas de usabilidad

Estas pruebas las realizaremos a nuestra aplicación final. Para la realización de esta, encuestaremos a ciertos usuarios.

Las pruebas se realizarán a 5 usuarios que utilizarán nuestra aplicación libremente mientras responden nuestras preguntas.

Las posibles respuestas son: “Si”, “No”, “NS/NC” (No sabe / No contesta). En nuestras encuestas no existe ninguna respuesta válida para todas las preguntas. Así, evitaremos que los usuarios encuestados sigan un patrón o contesten de forma positiva o negativa ante el descarte de ciertas preguntas.

Estos 5 encuestados mantendrán su nombre y datos en el anonimato. Han sido seleccionados minuciosamente para contar con diferentes perfiles de encuestados. La edad de estos usuarios varía entre 16 y 56 años, dependiendo de su conocimiento respecto la tecnología web y manejo de aplicaciones móviles. Lo que sí tendrán en común es el conocimiento del sector agrícola y la gestión de sus explotaciones. Para ello estos 5 encuestados serán miembros de la Cooperativa olivarera “Virgen de la Fuensanta” de Fuensanta de Martos.

ENCUESTA 1	
Edad:	16
Nivel de conocimiento	Alto

Pregunta	Respuesta		
	Si	No	NS/NC
¿Se indica de forma clara la sección o ubicación en la que se encuentra en cada momento?	Si		
¿El diseño de la interfaz sigue el mismo patrón para cada una de las secciones o ubicaciones?	Si		
¿Se utilizan enlaces del estilo “Click aquí” o “Ver más”?		No	
¿Existen esperas excesivas al mostrar alguna sección o se notan retardos al cargar ciertos datos?		No	
¿Aparecen sonidos en la aplicación sin previa autorización del usuario?		No	
¿Funciona correctamente la función de los botones “atrás” o “adelante”?	Si		
¿Se utilizan metáforas o dobles sentidos en textos o títulos de nuestra aplicación?		No	
¿Se utiliza correctamente y moderadamente el uso de letra cursiva, itálica o negrita?	Si		
¿Le resultó complejo utilizar o configurar el riego de su explotación?		No	
¿Le resultó complejo la consulta de datos o monitoreo de los datos registrados por los sensores?		No	
¿Ve compleja la utilización en general de la aplicación para cualquier tipo de persona ?		No	
¿Considera inseguro o inadecuado el trato de sus datos?		No	
¿Considera necesario una buena formación para la utilización de nuestra aplicación?		No	

Tabla 22: Encuesta usabilidad encuesta 1

ENCUESTA 2	
Edad:	20
Nivel de conocimiento	Medio-Alto

Pregunta	Respuesta		
	Si	No	NS/NC
¿Se indica de forma clara la sección o ubicación en la que se encuentra en cada momento?	Si		
¿El diseño de la interfaz sigue el mismo patrón para cada una de las secciones o ubicaciones?	Si		
¿Se utilizan enlaces del estilo “Click aquí” o “Ver más”?		No	
¿Existen esperas excesivas al mostrar alguna sección o se notan retardos al cargar ciertos datos?		No	
¿Aparecen sonidos en la aplicación sin previa autorización del usuario?		No	
¿Funciona correctamente la función de los botones “atrás” o “adelante”?	Si		
¿Se utilizan metáforas o dobles sentidos en textos o títulos de nuestra aplicación?	Si		
¿Se utiliza correctamente y moderadamente el uso de letra cursiva, itálica o negrita?	Si		
¿Le resultó complejo utilizar o configurar el riego de su explotación?		No	
¿Le resultó complejo la consulta de datos o monitoreo de los datos registrados por los sensores?		No	
¿Ve compleja la utilización en general de la aplicación para cualquier tipo de persona ?		No	
¿Considera inseguro o inadecuado el trato de sus datos?		No	
¿Considera necesario una buena formación para la utilización de nuestra aplicación?		No	

Tabla 23: Encuesta usabilidad encuesta 2

ENCUESTA 3	
Edad:	24
Nivel de conocimiento	Medio-Bajo

Pregunta	Respuesta		
	Si	No	NS/NC
¿Se indica de forma clara la sección o ubicación en la que se encuentra en cada momento?	Si		
¿El diseño de la interfaz sigue el mismo patrón para cada una de las secciones o ubicaciones?	Si		
¿Se utilizan enlaces del estilo “Click aquí” o “Ver más”?		No	
¿Existen esperas excesivas al mostrar alguna sección o se notan retardos al cargar ciertos datos?		No	
¿Aparecen sonidos en la aplicación sin previa autorización del usuario?		No	
¿Funciona correctamente la función de los botones “atrás” o “adelante”?	Si		
¿Se utilizan metáforas o dobles sentidos en textos o títulos de nuestra aplicación?		No	
¿Se utiliza correctamente y moderadamente el uso de letra cursiva, itálica o negrita?		No	
¿Le resultó complejo utilizar o configurar el riego de su explotación?		No	
¿Le resultó complejo la consulta de datos o monitoreo de los datos registrados por los sensores?		No	
¿Ve compleja la utilización en general de la aplicación para cualquier tipo de persona ?			NS/NC
¿Considera inseguro o inadecuado el trato de sus datos?		No	
¿Considera necesario una buena formación para la utilización de nuestra aplicación?		No	

Tabla 24: Encuesta usabilidad encuesta 3

ENCUESTA 4	
Edad:	32
Nivel de conocimiento	Bajo

Pregunta	Respuesta		
	Si	No	NS/NC
¿Se indica de forma clara la sección o ubicación en la que se encuentra en cada momento?			NS/NC
¿El diseño de la interfaz sigue el mismo patrón para cada una de las secciones o ubicaciones?	Si		
¿Se utilizan enlaces del estilo “Click aquí” o “Ver más”?		No	
¿Existen esperas excesivas al mostrar alguna sección o se notan retardos al cargar ciertos datos?		No	
¿Aparecen sonidos en la aplicación sin previa autorización del usuario?		No	
¿Funciona correctamente la función de los botones “atrás” o “adelante”?	Si		
¿Se utilizan metáforas o dobles sentidos en textos o títulos de nuestra aplicación?			NS/NC
¿Se utiliza correctamente y moderadamente el uso de letra cursiva, itálica o negrita?			NS/NC
¿Le resultó complejo utilizar o configurar el riego de su explotación?		No	
¿Le resultó complejo la consulta de datos o monitoreo de los datos registrados por los sensores?		No	
¿Ve compleja la utilización en general de la aplicación para cualquier tipo de persona ?		No	
¿Considera inseguro o inadecuado el trato de sus datos?			NS/NC
¿Considera necesario una buena formación para la utilización de nuestra aplicación?		No	

Tabla 25: Encuesta usabilidad encuesta 4

ENCUESTA 5	
Edad:	59
Nivel de conocimiento	Muy Bajo

Pregunta	Respuesta		
	Si	No	NS/NC
¿Se indica de forma clara la sección o ubicación en la que se encuentra en cada momento?	Si		
¿El diseño de la interfaz sigue el mismo patrón para cada una de las secciones o ubicaciones?	Si		
¿Se utilizan enlaces del estilo “Click aquí” o “Ver más”?		No	
¿Existen esperas excesivas al mostrar alguna sección o se notan retardos al cargar ciertos datos?		No	
¿Aparecen sonidos en la aplicación sin previa autorización del usuario?		No	
¿Funciona correctamente la función de los botones “atrás” o “adelante”?	Si		
¿Se utilizan metáforas o dobles sentidos en textos o títulos de nuestra aplicación?			NS/NC
¿Se utiliza correctamente y moderadamente el uso de letra cursiva, itálica o negrita?			NS/NC
¿Le resultó complejo utilizar o configurar el riego de su explotación?		No	
¿Le resultó complejo la consulta de datos o monitoreo de los datos registrados por los sensores?		No	
¿Ve compleja la utilización en general de la aplicación para cualquier tipo de persona ?		No	
¿Considera inseguro o inadecuado el trato de sus datos?		No	
¿Considera necesario una buena formación para la utilización de nuestra aplicación?	Si		

Tabla 26: Encuesta usabilidad encuesta 5

RESPUESTAS			
Pregunta	**Respuesta**		
	Acertadas	Falladas	NS/NC
¿Se indica de forma clara la sección o ubicación en la que se encuentra en cada momento?	4	0	1
¿El diseño de la interfaz sigue el mismo patrón para cada una de las secciones o ubicaciones?	5	0	0
¿Se utilizan enlaces del estilo “Click aquí” o “Ver más”?	5	0	0
¿Existen esperas excesivas al mostrar alguna sección o se notan retardos al cargar ciertos datos?	5	0	0
¿Aparecen sonidos en la aplicación sin previa autorización del usuario?	5	0	0
¿Funciona correctamente la función de los botones “atrás” o “adelante”?	5	0	0
¿Se utilizan metáforas o dobles sentidos en textos o títulos de nuestra aplicación?	2	1	2
¿Se utiliza correctamente y moderadamente el uso de letra cursiva, itálica o negrita?	2	1	2
¿Le resultó complejo utilizar o configurar el riego de su explotación?	5	0	0
¿Le resultó complejo la consulta de datos o monitoreo de los datos registrados por los sensores?	5	0	0
¿Ve compleja la utilización en general de la aplicación para cualquier tipo de persona ?	4	0	1
¿Considera inseguro o inadecuado el trato de sus datos?	4	0	1
¿Considera necesario una buena formación para la utilización de nuestra aplicación?	4	1	0

Tabla 27: Resultados encuestas usabilidad

Capítulo 6: Conclusiones y trabajo futuro

6.1 Conclusiones

Este proyecto surgió con la idea de automatizar tareas en el sector primario, más concretamente a la agricultura. La tarea de riego es una de las tareas más complejas ya que un exceso o déficit de humedad puede afectar directamente a la producción de nuestra explotación. Ví como una buena opción incorporar la tecnología en esta tarea, pudiendo monitorear o controlar el riego a través de un dispositivo IOT y sus respectivos sensores para registrar valores ambientales.

Nuestro primer objetivo era el control de riego remotamente. Por ello, utilizamos un dispositivo IOT como es Arduino para la conexión a Internet. El segundo objetivo era el monitoreo a tiempo real de los diferentes valores ambientales, así que, utilizamos la plataforma Firebase Data Base RealTime y sensores en nuestro dispositivo IOT. El tercer y último objetivo que nos planteamos fue la realización de una aplicación multiplataforma, intuitiva y fácil de utilizar, en la cual se pudiese controlar, programar y monitorear el riego en nuestra explotación a tiempo real.

Para el desarrollo de la interfaz web sería responsiva para cualquier tipo de dispositivo que se desarrolló con el Framework Angular. Además cuenta con una gran comunidad de desarrolladores y podemos encontrar documentación actualizada y resolución de problemas.

En la fase de Análisis y Diseño se establecieron objetivos y directrices que no fueron totalmente capaces de llevarse a cabo en la parte de implementación. Durante el desarrollo del mismo proyecto tuvimos que replantearnos e incluso modificar en algunos de ellos por los cambios que tuvimos que realizar.

Todas las tecnologías escogidas para el desarrollo de este proyecto han tenido una curva de aprendizaje rápida y positiva, dotando a nuestro proyecto con una amplia funcionalidad siendo es capaz de satisfacer con creces los objetivos funcionales necesarios y exigidos por la tarea de riego de nuestros usuarios finales.

6.2 Trabajo futuro

Este proyecto se implantará pronto en diferentes explotaciones agrarias, gracias a presentación de nuestra idea en una asamblea de la Cooperativa Olivarera de Fuensanta de Martos “Virgen de la Fuensanta”. Este primer lanzamiento lo utilizaremos como prueba durante varios meses. Así, tendremos expuestos a los diferentes factores ambientales nuestros dispositivos, comprobando el correcto funcionamiento y resistencia de estos.

Tendremos una primera inversión en la compra de los diferentes dispositivos y servicios, pero nos van a dar una subvención de 350€ por parte de la Cooperativa anteriormente mencionada. Gracias a estos ingresos diseñaremos contenedores con impresión 3D para almacenar los dispositivos y sensores de calidad.

Una vez hemos terminado el desarrollo de nuestra aplicación respecto los límites de tiempos establecidos, plantearemos nuevas funciones que mejoraran las capacidades de nuestro sistema en versiones futuras.

En primer lugar, añadiríamos una nueva sección. Sería una agenda agraria para que los usuarios finales puedan anotar datos interesantes como son los registros de trabajos realizados por alguien durante un determinado tiempo, producción obtenida, cantidad y periodicidad de los fitosanitarios aplicados.

Una funcionalidad propuesta por la Cooperativa “Virgen de la Fuensanta” para el futuro, es la consulta y manipulación de la sección de crédito de esta entidad. Esta sociedad cuenta con una sección de crédito propia que solo se puede acceder a ella de forma física en su banco y carece de acceso a través de Internet.

Otras de las posibles funciones futuras es la incorporación de cámaras IP, para ver en todo momento una o distintas partes de nuestra explotación a tiempo real.

Y por último destacar la posibilidad de comparar los registros y datos de explotaciones de los diferentes usuarios y publicar los diferentes datos de interés obtenidos en nuestra aplicación.

REFERENCIAS

[1] Tecnologías aplicadas a la agricultura: https://www.agroptima.com/es/blog/5-innovaciones-tecnologicas-que-todo-agricultor-deberia-conocer/

[2] Industria y tecnología agraria: https://www.bialarblog.com/tecnologia-agricola-agtech-agricultura/

[3] SmartAgro: https://www.bialarblog.com/smart-agro-agricultura-4-0-tecnologias-agricolas/

[4] Tendencias y evolución: https://agriculturers.com/nuevas-tendencias-en-tecnologia-agricola/

[5] Manual de Arduino: https://arduinobot.pbworks.com/f/Manual+Programacion+Arduino.pdf

[6] Tutoriales Arduino: http://cursoarduino.proserquisa.com/category/tutoriales/

[7] Manual de Firebase: https://firebase.google.com/docs/web/setup?hl=es

[8] Manual de Angular: https://desarrolloweb.com/manuales/manual-angular-2.html

[9] Documentación de Angular: https://angular.io/docs

[10] Proveedor Hardware: https://www.aliexpress.com

[11] Conexión Firebase – Arduino: https://create.arduino.cc/projecthub/electropeak/connecting-arduino-to-firebase-to-send-receive-data-cd8805

[12] Conexión Angular – Firebase (Material Design): https://www.adictosaltrabajo.com/2017/09/05/angular-material-firebase/

[13] Angular y Firebase Front-end: https://www.adictosaltrabajo.com/2017/09/05/angular-material-firebase/

[14] Arduino y conexión físicas de sensores: https://descubrearduino.com/sensores-para-arduino-que-debes-aprender-a-utilizar/

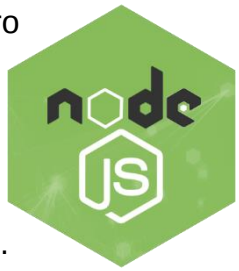
[15] Control de bombas de agua con Arduino: https://www.luisllamas.es/bomba-de-agua-con-arduino/

[16] Libro: Arduino y el Internet de las cosas - Johnny Novillo-VicuñaDixys Hernández RojasBertha Mazón OlivoJimmy Molina RíosOscar Cárdenas Villavicencio : https://play.google.com/store/books/details/Johnny_Novillo_Vicu%C3%B1a_Arduino_y_el_Internet_de_las?id=FIlyDwAAQBAJ&gl=ES

ANEXO I: GUIA DE INSTALACIÓN

En esta sección veremos las diferentes instalaciones necesarias, así como la ejecución de estas para la ejecución de nuestro proyecto.

Instalación y ejecución de Node JS y Angular 8



En primer lugar descargaremos la versión compatible con nuestro sistema operativo de su página oficial Node JS : https://nodejs.org/es/, tras la descarga procederemos a la instalación desde el ejecutable o paquete descargado. Esta instalación no requiere de ninguna configuración adicional salvo añadir al PATH del sistema la ruta en la que se ha instalado.

Gracias a la instalación previa de Node JS contamos con el gestor de paquetes “npm” el cual nos facilitará la instalación del framework de Angular. En nuestro caso hemos utilizado una versión especifica cómo es la 8.0.1, para su instalación tenemos ejecutar el siguiente comando desde la terminal:

```
npm install -g @angular/cli@8.0.1
```

Ya podemos ejecutar nuestra aplicación en formato web desde nuestro servidor local, para ello nos posicionaremos desde la terminal en la carpeta del proyecto y ejecutamos el siguiente comando:

```
npm start
```

Una vez arrancado nuestro servidor podremos consultar a través del navegador nuestra aplicación web desde la dirección: http://localhost:4200

Instalación de Cordova y compilación

Una vez tengamos nuestra aplicación web ejecutada y hayamos comprobado que funciona correctamente, instalaremos las siguientes herramientas para compilar nuestro código generado con Angular para poder ejecutarlo en cualquier otra plataforma. Para la instalación de Cordova ejecutaremos el siguiente comando desde nuestra terminal:

```
npm install -g cordova
```

Una vez terminada su instalación, ya dependiendo de las plataformas en las cuales queramos compilar nuestro proyecto, necesitaremos una serie de herramientas. La guía de instalación y configuración para cada uno de ellos podemos encontrarlos en su página oficial: https://cordova.apache.org/docs/en/latest/

Los comandos necesarios para incorporar diferentes plataformas son:

```
$ cordova platform add ios
$ cordova platform add amazon-fireos
$ cordova platform add android
$ cordova platform add blackberry10
$ cordova platform add firefoxos
```

Para generar los diferentes instaladores o ejecutables dependiendo de la plataforma que queramos, será necesario ejecutar el siguiente comando variando el ultimo atributo que será el que hace referencia a la plataforma deseada.

```
cordova build “plataforma deseada”
```

ANEXO II: MANUAL DE USUARIO

En esta sección veremos cada una de las vistas en formato web, indicando la funcionalidad y las opciones disponibles:

Login

Login creado con Bootstrap en su totalidad. Es un formulario html, el cual mediante JavaScript en tiempo real comprueba que ambos campos no estén vacíos y comprueba que el email sea una cadena válida. A parte la contraseña, su único requisito es que tenga más de 6 caracteres alfanuméricos.

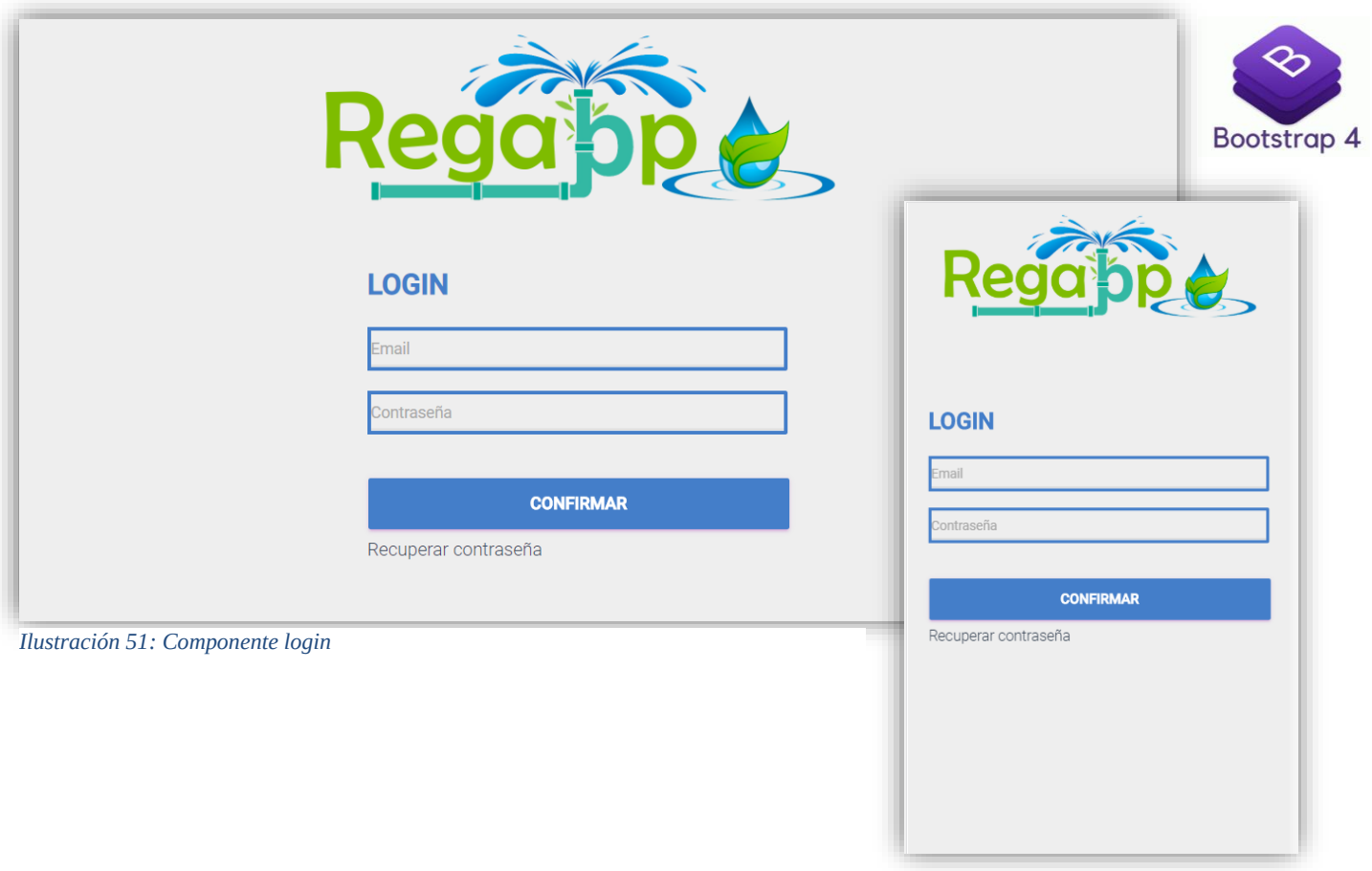


Ilustración 51: Componente login

Inicio

Esta vista muestra los últimos datos registrados por nuestro sistema. Como podemos ver en la siguiente vista escritorio completa, contamos con un menú a la izquierda para movernos por las diferentes vistas de nuestra aplicación. También contamos con un botón en dicho menú para cerrar la sesión del usuario en nuestra aplicación.

Ilustración 52: Componente Inicio

1. Últimos datos registrados por nuestro sistema.
    a. Sensor humedad de la tierra (%).
    b. Sensor temperatura (ºC).
    c. Sensor humedad superficial.
    d. Sensor lluvia (Aproximación en Litros).
2. Gráficas animadas.
    a. Gráfica de 4 registros (cada 6 horas) de los últimos 7 días que muestra la humedad de la tierra de nuestra finca.
    b. Gráfica de 4 registros (cada 6 horas) de los últimos 7 días que muestra la temperatura en nuestra finca.
3. Widget que nos muestra la previsión meteorológica para los futuros 4 días y nos redirecciona a la página de AEMET para ver más concretamente la previsión meteorológica hasta un máximo de 14 días.
4. Tabla que nos indican los datos registrados de los últimos 7 riegos aplicados, indicando la fecha, duración, litros aproximados de agua y tipo de riego:
    a. Manual - Efectuado forzosamente por el usuario.
    b. Automático – Efectuado mediante nuestro algoritmo.
    c. Programado – Programado previamente por el usuario.
5. Pie de página: podemos consultar los diferentes términos y licencias y aparte consultar los datos de contacto para comunicarse con nosotros.
6. Menú: índice de paginación que tiene una funcionalidad de navegación entre los diferentes componentes de nuestra aplicación.

La vista vertical o estrecha (móvil) completa está estructurada de una forma diferente para optimizar el espacio.

Unos de los cambios más relevantes es el menú que aparecerá predeterminadamente oculto en la parte derecha de la pantalla y se mostrará animadamente mediante JavaScript a través de un botón de hamburguesa flotante. Este botón está posicionado fijo en la pantalla en la parte superior derecha.

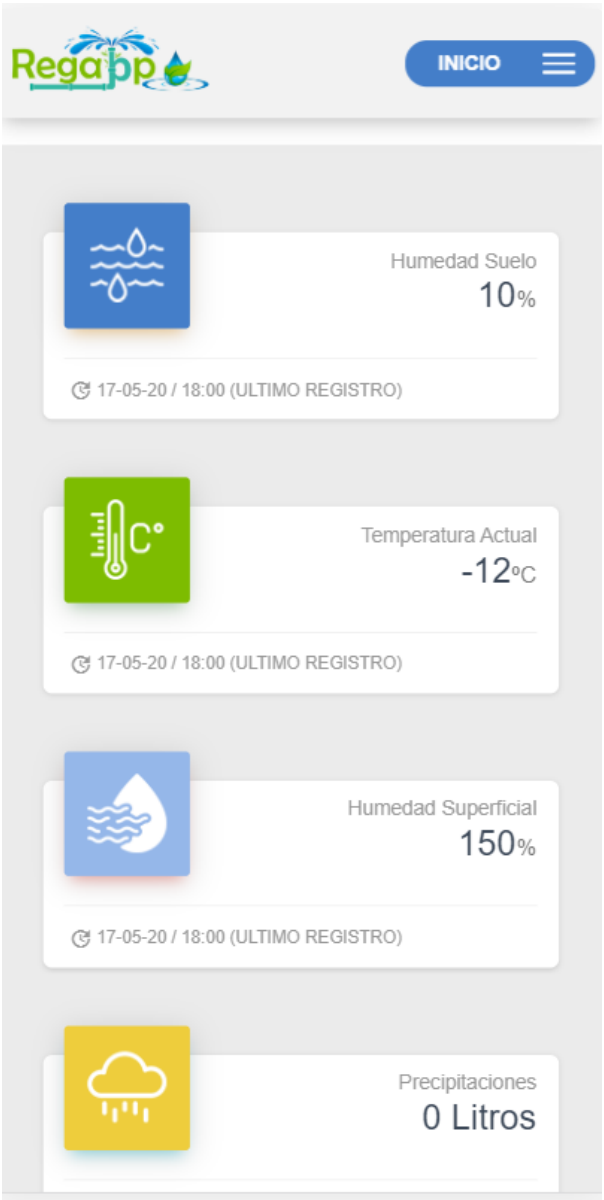


Ilustración 54: Componente Inicio 1

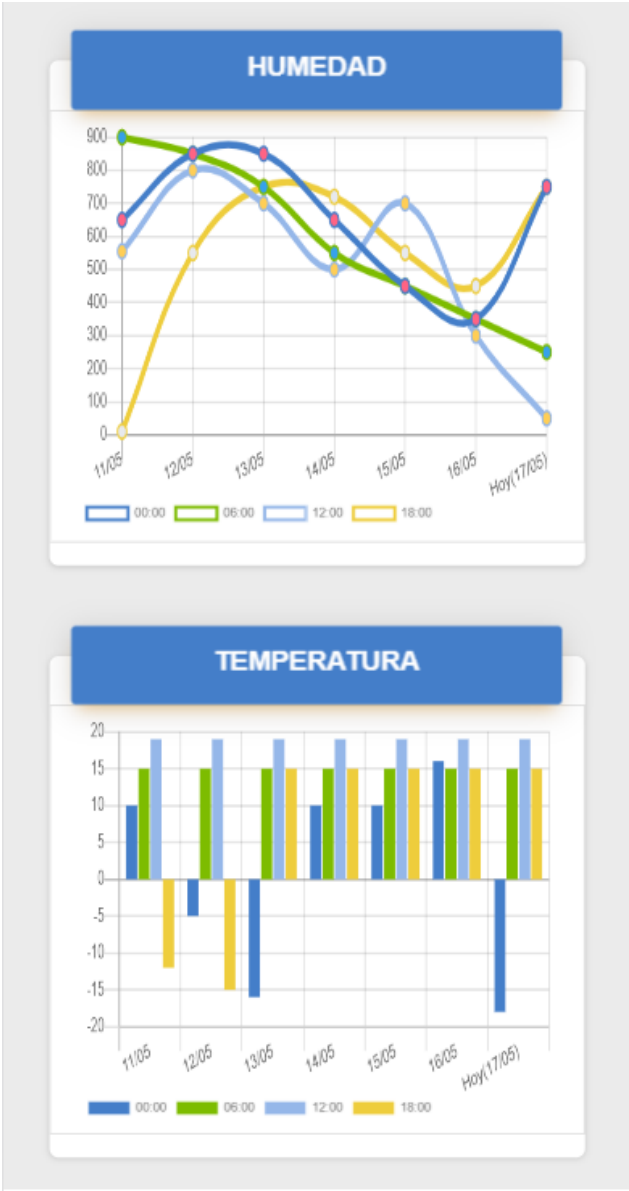


Ilustración 53:Componente Inicio 2

Ilustración 55: Componente Inicio 3

Control

Esta segunda vista se centra en el control del riego total e interactivo con el usuario final.

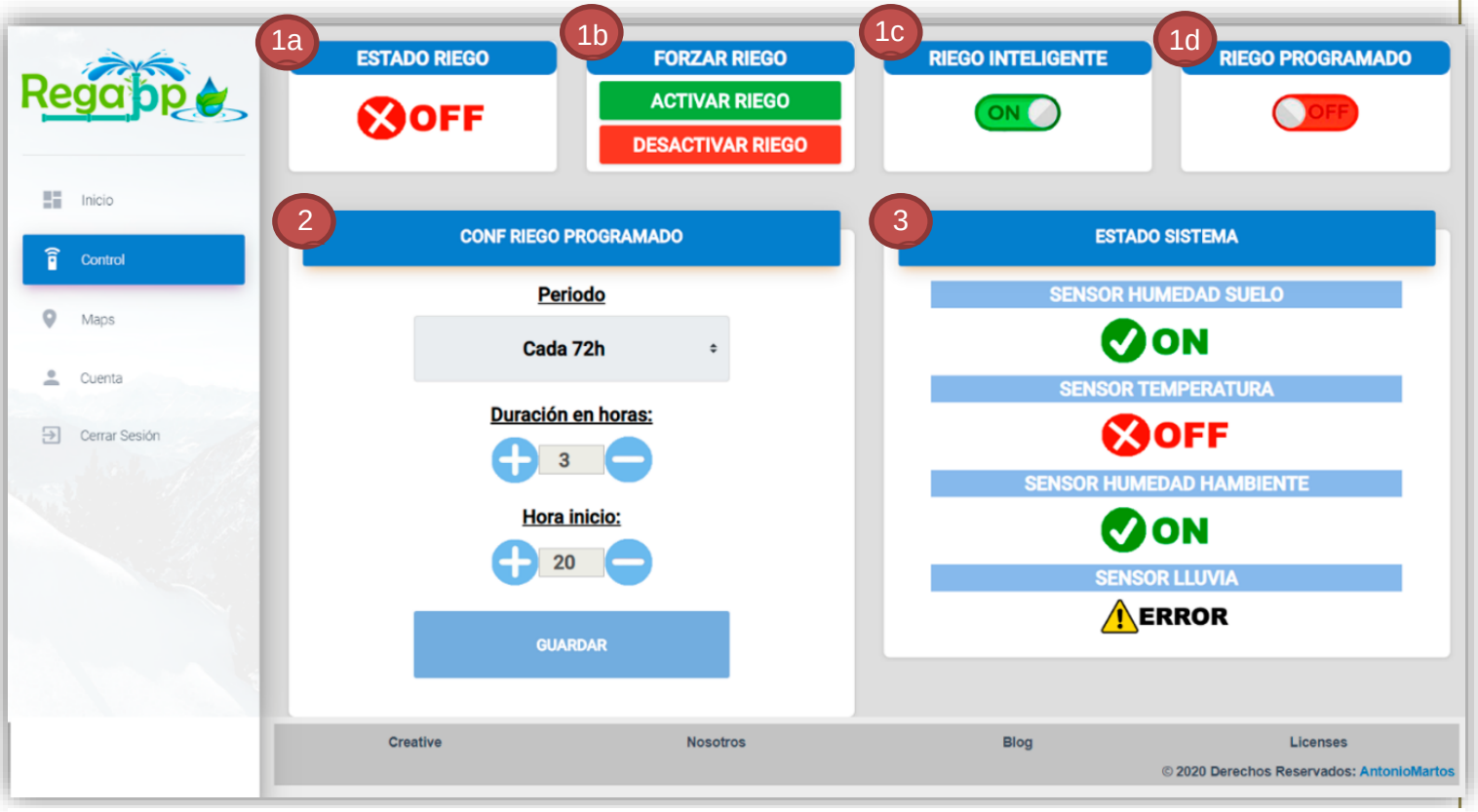


Ilustración 56: Componente Control

1. Interacción y monitoreo a tiempo real de nuestro sistema.
    a. Estado de la bomba o bombas de riego instaladas: Muestra “On” en verde en caso de estar activado el riego en nuestro sistema y “Off” en caso contrario
    b. Podremos activar o desactivar el riego manualmente.
    c. Activar o desactivar el algoritmo inteligente que incorpora nuestro sistema para determinar el momento y duración automáticos para nuestra explotación, respecto los registros obtenidos por nuestros sensores.
    d. Podremos activar o desactivar el riego programado por el usuario en esta misma vista en el “elemento 2”.
2. Configuración y programación del riego.
    a. Periodicidad del riego mediante selector de opciones.

   b. Duración del riego: viene establecida en formato hora y tiene una limitación máxima de 24 horas.
   c. Hora de Inicio: hora a la que empezará a activarse el riego programado.

3. Estado del sistema: Aquí se mostrará el estado en el que se encuentra cada sensor, devolviendo 3 diferentes códigos de color como aparece en la anterior ilustración.
   a. On: El sensor se encuentra en buen estado y funcional.
   b. Off: El sensor esta desconectado o no es reconocido por el sistema.
   c. Error: El sensor muestra valores anómalos que pueden derivarse de una rotura, problema de alimentación o avería del mismo.

Ahora veremos las diferentes vistas desde una versión móvil, la única diferencia es el posicionamiento del menú flotante:

Ilustración 58: Componente Control 1

Ilustración 57: Componente Control 2

Localización

En la siguiente vista veremos información acerca de nuestra explotación.

Ilustración 59: Componente Localización

1. Mapa con la geolocalización de Google Maps que muestra el punto exacto donde se encuentra nuestra explotación y en el que podemos consultar su imagen desde satélite.
2. Slider de imágenes con información de nuestra explotación.
3. Enlaces externos a aplicaciones para la geolocalización de nuestra finca y delimitaciones de esta.
4. Slider con imágenes e información del tipo de terreno y esquema del sistema de riego.

Mostraremos la vista móvil en la cual solamente cambia el posicionamiento del menú y su función de activar u ocultar dicho menú:

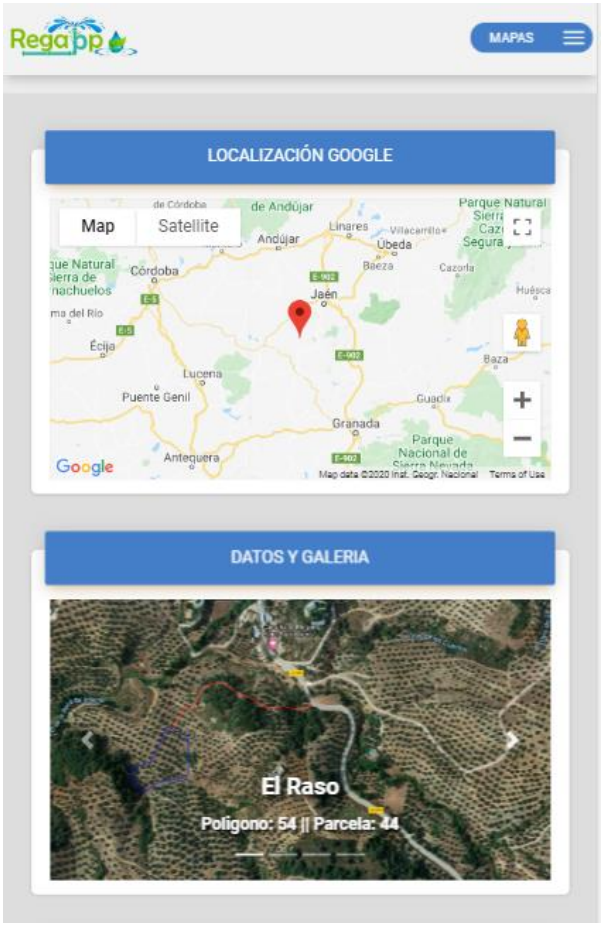


Ilustración 61: Componente Localización 1

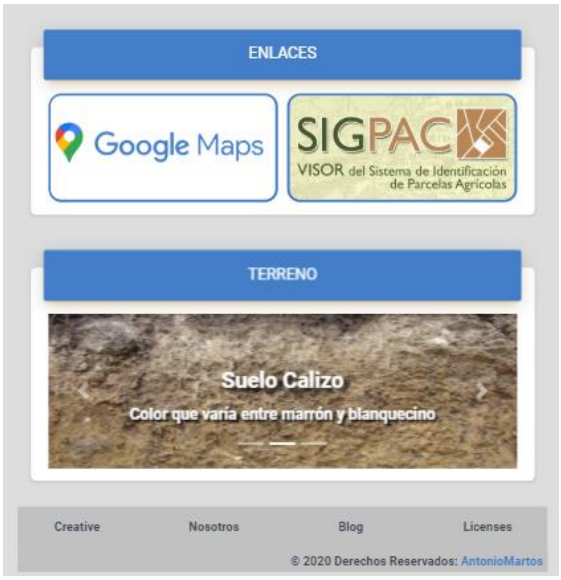


Ilustración 60: Componente Localización 2

Cuenta

En esta vista modificaremos y consultaremos los datos de usuario y podremos cambiar la contraseña de inicio de sesión de la aplicación.

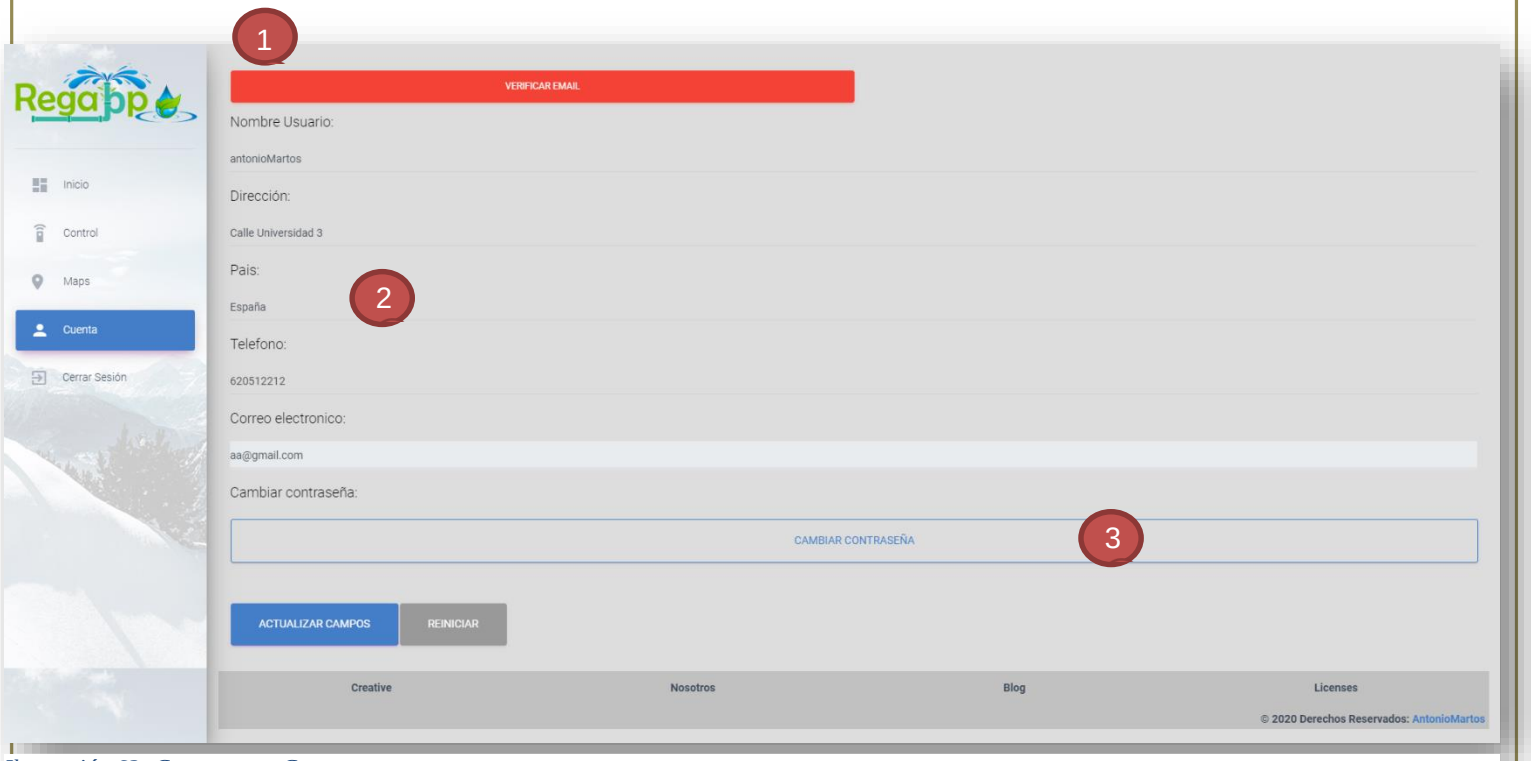


Ilustración 62: Componente Cuenta

1. Verificación de email: en caso de no haber realizado esta acción, saltara cada vez que se acceda a esta vista Cuenta, ya que es necesario por primera vez para confirmar el correo electrónico del usuario.
2. Formulario de datos del usuario: se puede actualizar o inicializar mediante los botones de acción.
3. Cambio de contraseña: enviará un correo electrónico para proceder al cambio de contraseña.

A continuación, mostraremos la vista móvil, en la cual se mantiene el mismo menú flotante de navegación.

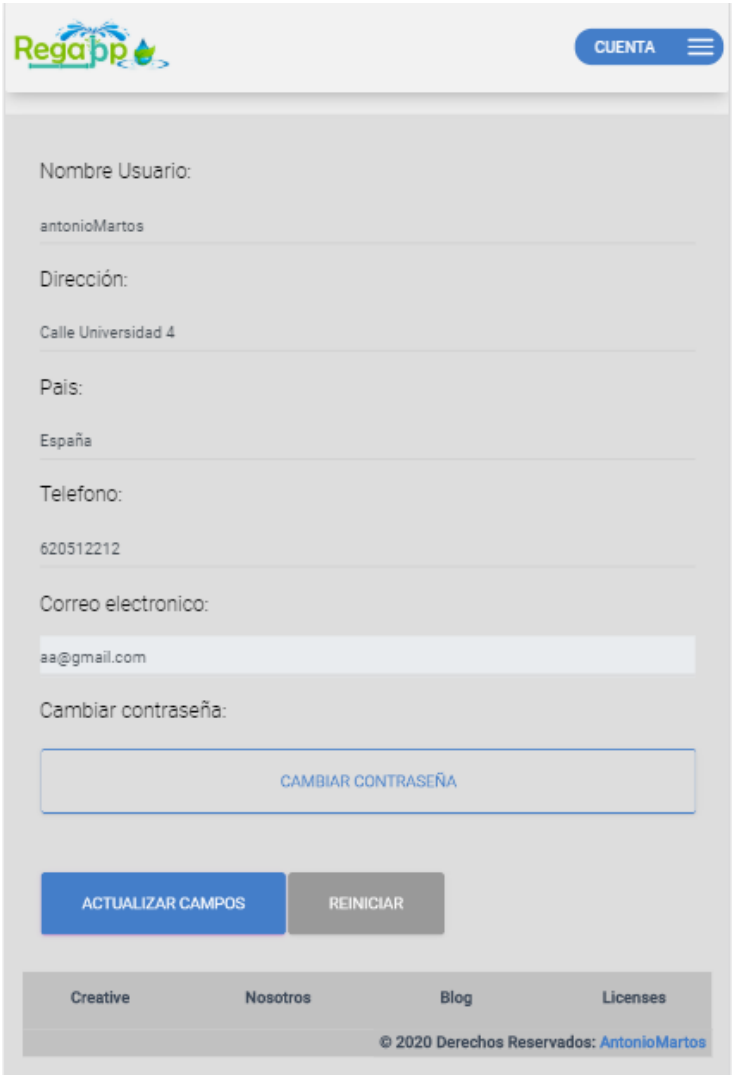


Ilustración 63: Componente Cuenta Móvil