ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADUADA/O EN INGENIERÍA DEL SOFTWARE

Informática aplicada a la toma de datos en invernaderos de investigación

Informatics applied to data gathering in research greenhouses

Realizado por
D. Manuel Leiva Gómez

Tutorizado por
Dr. D. Sergio Gálvez Rojas

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, septiembre 2022

Resumen

Para mejorar la calidad de los cultivos de interés agroecónomico es necesario llevar a cabo un riguroso estudio del crecimiento de la planta durante todas sus fases, desde el control de la calidad de las semillas a mediciones exhaustivas del tamaño del tallo y hojas, humedad, temperatura o nutrientes del sustrato. Normalmente estas plantas son cultivadas en macetas dentro de invernaderos para así poder controlar y mejorar las condiciones bajo las que se encuentran.

Para realizar el control de cada maceta y de esta forma evaluar hasta qué punto un tipo de semilla supone una mejora sobre otras normalmente se realizan mediciones de distinta índole una o dos veces por semana. Estas mediciones pueden ser desde fotos de distintos ángulos de la planta hasta análisis espectrales con aparatos especializados o toma de datos de la transpiración del agua o de la longitud de las hojas.

En un invernadero, con cientos o incluso miles de plantas, es muy importante asociar los datos tomados a cada planta con una lista de datos previos de forma correcta. Esta operación suele hacerse de forma manual en dos fases: una toma de datos en el invernadero y un trasvase de los datos a medios informáticos en laboratorio. Trabajar de esta forma puede dar lugar a confusión y a la mezcla de datos de forma errónea. Este trabajo da solución a este problema mediante una aplicación web capaz de generar y manipular códigos QR y configuraciones asociadas a cada una de las plantas con una serie de campos de datos a tomar. Para recoger dichos datos se hace uso de una aplicación móvil que permite escanear los códigos QR asociados a cada una de las plantas y realizar las tomas de datos de forma sencilla.

Palabras clave:

Invernadero, Planta, Código QR, Medición, Aplicación Web/Móvil

Abstract

To improve the quality of agro-economic interest crops it is necessary to carry out a rigorous study of plant growth during all its phases, ranging from seed quality control to exhaustive measurements of steam and leaf size, humidity, temperature or nutrients in the substrate. These plants are usually grown in pots inside of greenhouses in order to control and improve the conditions under which they are grown.

In order to monitor each pot and thus evaluate to what extent one type of seed is an improvement over another, measurements of various kinds are usually taken once or twice a week. These measurements can range from photos taken from different angles of the plant to spectral analysis with specialized equipment or taking data on water transpiration or leaf length.

In a greenhouse with hundreds or even thousands of plants, it is very important to associate the data taken for each plant with a list of previous data in a correct way. This process is usually done manually in two phases: a data collection in the greenhouse and a transfer of said data to computerized means in the laboratory. Working in this way can lead to confusion and to the possibility of the erroneous mixing of data. This work gives a solution to this problem by means of a web application capable of generating and manipulating QR codes and configurations associated to each plant with a series of data fields to be collected. To collect these data, a mobile application is used to scan the QR codes associated with each of the plants, which allows collecting data in an easy way.

Keywords:

Greenhouse, Plant, QR Code, Measuring, Web/Mobile application.

Índice

Índice de figuras

1

Introducción

1.1 Motivación

Mejorar la calidad de los cultivos de interés agroeconómico es una de las principales tareas que se pretenden llevar a cabo en el campo de la agricultura. Para conseguir mejorar de una forma sustancial rasgos como la cantidad de proteína, de gluten o de carotenos, la biomasa total, o incluso obtener plantas de menor altura para evitar la ruptura del tallo, es necesario llevar un seguimiento de todas las fases del crecimiento de las diferentes semillas plantadas.

Para que el cultivo a gran escala sea factible es necesario realizar estas pruebas en entornos cerrados como invernaderos de investigación, donde se plantan varias macetas que son estudiadas exhaustivamente con controles que normalmente se realizan una o dos veces por semana. Estos controles comprenden una gran variedad de aspectos distintos, desde el estudio de la evolución de la humedad de la maceta a análisis espectrales con aparatos especializados o mediciones de la longitud de la hoja bandera o las espigas de la planta en el caso del trigo y similares. [1]

Un invernadero de investigación puede albergar cientos o incluso miles de diferentes plantas siendo monitorizadas y estudiadas de forma rigurosa. Normalmente, los estudios que se realizan a las plantas se hacen de forma manual, siendo un trabajador del invernadero el que se encarga de apuntar de forma manual los datos tomados en cada una de ellas para luego tratarlos de forma digital en una base de datos o de documentos en la que se alberga el resto de datos anteriormente tomados para cada planta.

Este proceso puede dar lugar a confusión y a la mezcla inesperada de datos entre distintas macetas, lo que afectaría de forma negativa a largo plazo a la toma de decisión y estudio con respecto a qué tipo de semilla aporta una mayor calidad o un mejor resultado. En otras palabras, al tratar los datos de forma manual se da pie a la posibilidad de error humano a la hora de escribir o de confundir datos tomados, además del obvio esfuerzo que supone estudiar un número muy elevado de plantas.

1.2 Objetivos

El objetivo principal del proyecto es mejorar y facilitar la obtención y el tratamiento de los datos tomados en las diferentes plantas de un invernadero de investigación. Para conseguir este objetivo se ha desarrollado una aplicación web en Spring que permite crear un registro de las plantas que haya en los invernaderos, permitiéndonos obtener un código QR asociado a cada una y asociar una configuración con una serie de campos. La aplicación web ofrece una interfaz gráfica en la que consultar todos los datos tomados anteriormente y la capacidad de ver las estadísticas a lo largo del tiempo de los valores numéricos tomados en las plantas.

Para realizar la lectura de los códigos QR y la posterior toma de datos se ha desarrollado también una aplicación móvil en React Native, que mostrará los campos correspondientes a rellenar según la configuración que hayamos elegido en la web. Una vez rellenos estos campos se transferirán a una base de datos ubicada en un servidor de investigación.

1.3 Estructura de la memoria

La presente memoria está estructurada en seis capítulos que se describen a continuación:

1. Capítulo 1: Introducción. Este capítulo explica el porqué del trabajo que se ha realizado y cuáles son los objetivos a cumplir.
2. Capítulo 2: Tecnologías utilizadas. Este capítulo aborda las principales tecnologías utilizadas, en qué se basan, cómo funcionan y cuáles son sus principales ventajas e inconvenientes
3. Capítulo 3: Método de trabajo. Se explican el método de trabajo usado para programar las dos aplicaciones y los principales pasos tomados.
4. Capítulo 4: Requisitos y Casos de Uso. Se proporciona una lista de requisitos funcionales y no funcionales tanto de la aplicación web como de la aplicación móvil junto con una lista de casos de uso de las principales funcionalidades de ambas.
5. Capítulo 5: Diseño. Se muestran diagramas de flujo y de secuencia de las aplicaciones y se explica a fondo cómo funciona cada parte de ambas.
6. Capítulo 6: Conclusión y líneas futuras. Se proporciona una conclusión del trabajo realizado y se debaten posibles usos y modificaciones a realizar para adecuar las aplicaciones a cualquier otra necesidad de los investigadores.

2

Metodología de trabajo

2.1 Requisitos

La primera fase de trabajo antes de comenzar el desarrollo del proyecto consiste en la elaboración de un documento general de requisitos (DGR) en el que se especifican las funcionalidades que debe cumplir la página en forma de requisitos tanto funcionales como no funcionales. Además, se especifican varios aspectos del proyecto como pueden ser el objetivo, los interesados o las ventajas que puede aportar. Ver Capítulo 3.

2.2 *Mockups*

Antes de comenzar el desarrollo del proyecto se realizan varios *mockups* de la página web y de la aplicación móvil para usarlos como guías a la hora de implementar la interfaz y las funcionalidades requeridas. Estos *mockups* se han creado usando la herramienta Figma y aportan una interfaz interactiva con la que se pueden explorar dichos *mockups* como si se estuviera usando la página como tal. Ver Capítulo 4 Apartado 1.

2.3 Desarrollo

La metodología de trabajo a la hora de programar la página web y la aplicación móvil ha consistido en varios Sprints de desarrollo en los que se acordaban varias funcionalidades a desarrollar y, tras ser completadas, se realizaba una reunión con el tutor para comprobar si dichas funcionalidades se habían hecho de forma correcta, comentar posibles errores y acordar qué nuevas funcionalidades serían las siguientes a desarrollar. La duración de estos Sprints ha sido de entre 3 a 4 semanas cada uno, dependiendo del número de funcionalidades acordadas a hacer.

Se han creado repositorios en GitHub en los que se han creado varias tablas con las actividades a realizar del proyecto tal y como se haría en un proyecto Scrum. El código desarrollado se ha subido a dichos repositorios de forma incremental (*commit*).

Repositorio de la aplicación web: https://github.com/manuleivaUma/TFGInvernaderoWeb

Repositorio de la aplicación móvil: https://github.com/manuleivaUma/TFGInvernaderoMovil

2.4 Memoria

La redacción de esta memoria ha seguido una metodología similar, siendo escrita tras terminar el desarrollo del proyecto y siguiendo el esquema de reunión anteriormente mencionado, esta vez con apartados de la memoria en vez de conjuntos de funcionalidades. Como ya se ha indicado anteriormente, el análisis de requisitos y el diseño de las aplicaciones, incluyendo los *mockups*, se realizó antes de comenzar la programación del proyecto y se han usado para facilitar toda la fase principal de desarrollo. Se incluyen al momento del desarrollo final de la memoria

2.5 Tecnologías utilizadas

A continuación se hará un breve resumen de las tecnologías que se han utilizado para crear el proyecto:

2.5.1 Java 

El lenguaje de programación usado para la programación del *backend* y gran parte del *frontend* ha sido Java. Se ha hecho uso de la versión 1.8 para asegurar la compatibilidad con la mayor cantidad de bibliotecas posibles. El *frontend* está formado por páginas JSP que cargan atributos del *backend* y generan el código HTML en base a los atributos recibidos.

2.5.2 HTML 

Para programar la parte visual estática del *frontend* se ha usado HTML5. Éste se genera de forma estática mediante condiciones Java, y de forma dinámica usando JavaScript.

2.5.3 JavaScript 

Se recurre a JavaScript para manejar todas las interacciones que realiza el usuario con la página y para generar y modificar dinámicamente el código HTML. Todo ha sido programado mediante JavaScript base y Bootstrap, sin uso de bibliotecas como JQuery en el código escrito. La biblioteca JQuery aparece importada en los JSP pero su única función es permitir la compatibilidad con Bootstrap.

2.5.4 Spring

Para la creación de la aplicación web se ha hecho uso del *framework* Spring. Este *framework* de código abierto permite crear aplicaciones Java, Groovy o Kotlin de forma sencilla a la hora de inyectar nuevas dependencias en un proyecto.

Spring está dividido en diferentes módulos. Una aplicación puede elegir hacer uso de los módulos de Spring que desee. Los principales en los que se divide son: Contenedor Principal (Core), web, Acceso de Datos e Integración, Programación Orientada a Aspectos (AOP) , Instrumentación y '*Testing*' o Pruebas.

El Core de Spring hace uso del principio de Inversión de Control, también conocido como inyección de dependencias. Su principal función será la de crear, configurar y relacionar los objetos que creemos en una aplicación. [2]

Gracias al Core se puede cambiar el flujo de ejecución de los objetos del programa al contrario de cómo se haría en la programación estándar. El proceso de Inversión de Control consiste en que, tras crear un objeto Spring, se definen sus dependencias mediante constructores, argumentos o propiedades después de ser construidos de forma automática, es decir, al inverso del proceso de localización de dependencias habitual. Esto permite realizar la inyección de dependencias sin necesidad de realizar las definiciones en el lenguaje que estemos usando.

Otro de los principales módulos de Spring es el módulo Web. Este permite utilizar controladores web, ya sea usando el Modelo Vista-Controlador o una aplicación REST. Con el módulo web se pueden definir los '*endpoints*' de nuestra aplicación web y recibir y enviar datos con nuestro *frontend*. Funciona mediante un Servlet central llamado DispatcherServlet que se encarga de procesar las peticiones hechas a la aplicación. [3]

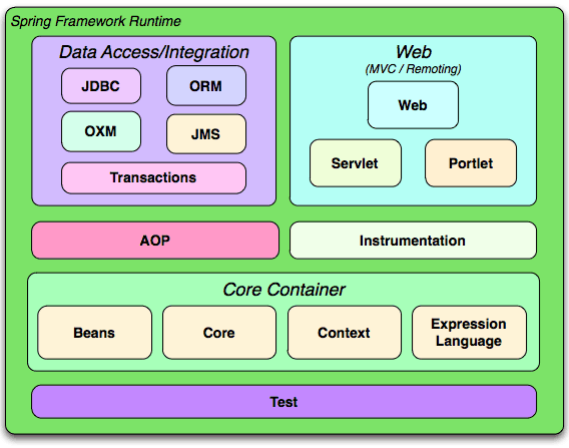


Figura 2.1: Representación de los módulos que ofrece el framework Spring..https://docs.spring.io/spring-framework/docs/3.2.x/spring-framework-reference/html/overview.html

2.5.5 Spring Boot 

Si bien puede verse que Spring es una herramienta de gran utilidad que puede facilitar la programación de muchas aplicaciones, también cabe decir que su uso y configuración pueden llegar a resultar complicados. Es por esto que se ha desarrollado una 'versión' alternativa llamada Spring Boot.

La creación de un proyecto Spring Boot puede realizarse tanto a mano como usando un inicializador de proyectos en forma web llamado Spring Initializr, que simplifica la búsqueda de dependencias inicial. [4]

Spring Boot permite realizar la configuración inicial de la aplicación de manera mucho más sencilla, además de ofrecer la posibilidad de desplegar la aplicación sin necesidad de crear un archivo .war que se deba desplegar en un servidor como podría ser Tomcat. De hecho, el propio Spring Boot ofrece la posibilidad de usar un Tomcat integrado en el propio *framework* mediante el uso de dependencias que se pueden incluir de forma simple en un archivo Maven pom.xml en cualquier momento de la creación del proyecto. Para hacer esto permite desplegar la aplicación mediante un .jar, sin necesidad de hacer uso del .war anteriormente mencionado.

Por otro lado, también proporciona una serie de dependencias predeterminadas llamadas 'starter' que incluyen todas las dependencias necesarias para usar una herramienta o funcionalidad sin necesidad de configurar ninguna de forma manual.

2.5.6 Hibernate 

Hibernate es un *framework* Java que permite combinar el modelo creado en una base de datos relacional con objetos definidos en clases Java. Para conectar un objeto Java con una tabla concreta de la base de datos se definen los atributos como variables en el objeto que estemos creando y se crean *getters* y *setters* correspondientes a cada atributo. [5] A continuación, se crean anotaciones en el formato soportado por Hibernate, en este caso JPA [6], indicando el nombre de la tabla con la que se está trabajando y los detalles de cada columna de la tabla para cada una de las variables.

Una vez creado el objeto de forma correcta se puede modificar y guardar los cambios en la base de datos de forma sencilla mediante el uso de repositorios.

2.5.7 Spring Security 

Spring Security es un *framework* de autenticación y control de acceso especializado para proyectos Spring. Permite realizar el inicio de sesión y el registro de usuarios de forma mucho más simple que si se programase de forma personalizada, bloqueando el acceso a recursos de la página a usuarios que no tengan el nivel de acceso adecuado dictado por un sistema de roles. Para el inicio de sesión se ha conectado el manejador de autenticación con una clase que, a su vez, hace uso de la clase Usuario creada con Hibernate para comprobar si los datos introducidos coinciden con la base de datos. [7]

Además, existe la posibilidad de la utilización de filtros tras el inicio de sesión o registros para realizar funciones personalizadas programadas por el usuario. En este caso, se ha creado una clase que se encarga de realizar una petición a la API RecaptchaV3 de Google para evitar intentos de inicio de sesión maliciosos.

Con respecto a la seguridad de las contraseñas, Spring Security puede encriptar las contraseñas con algoritmos de cifrado y aplicarles sal para un guardado seguro. En este caso se ha hecho uso del algoritmo BCrypt.

2.5.8 MariaDB 

MariaDB es un sistema para la gestión de bases de datos relacionales que surgió a partir de MySQL. Es un sistema gratuito y de código abierto con licencia GPLv2. Su gran similitud con MySQL le permite ser compatible con la mayoría de software ya programado para este sistema. [8]

La intención tras la creación de MariaDB es crear un sistema de bases de datos que asegure que seguirá siendo gratuito y con licencia GPL independientemente de cualquier entidad comercial.

La base de datos del proyecto se ha creado usando este sistema con la ayuda de MySQL Workbench para poder usar una interfaz gráfica más intuitiva aprovechando la compatibilidad con MySQL.

2.5.9 Zxing 

Zxing es una biblioteca para el escaneo y la creación de códigos de barras en Java, aunque también está disponible en otros lenguajes de programación. [9]

Zxing ofrece soporte para muchos tipos de códigos de barras, entre los que se encuentran los códigos QR. La biblioteca se ha encargado de generar los códigos QR a partir de los datos introducidos a la hora de crear una planta.

2.5.10 Bootstrap 

Bootstrap es una biblioteca que añade una gran cantidad de componentes CSS y que facilita de manera significativa el desarrollo de las interfaces web reduciendo la necesidad del uso de CSS personalizado. Para la creación de la aplicación web se ha usado la versión 4.5.0 junto a CSS propio para así diferenciar y mejorar la interfaz de la aplicación.

2.5.11 React Native 

Para crear la aplicación móvil se ha hecho uso de React Native, un *framework* JavaScript que permite desarrollar aplicaciones móviles tanto para Android como para iOS. Se basa en el *framework* web React para implantar sus funcionalidades básicas e incluye los componentes necesarios que podríamos encontrar en cualquier otro *framework* de desarrollo móvil. [10]

React Native, al igual que React, hace uso del estado, lo que permite asignar variables de estado al código que se actualizarán según el usuario interactúe con la página o se reciban datos vía red. La actualización de estos datos permite modificar los componentes que muestra la página. [11]

2.5.12 Expo 

Expo es una plataforma gratuita y de código abierto que ofrece un conjunto de herramientas que permite trabajar de forma mucho más sencilla con React Native, sin tener que usar el propio CLI nativo y dando la opción de desarrollar para Android e iOS usando el mismo código. [12]

Expo tiene una gran cantidad de bibliotecas que pueden incluirse en un proyecto de React Native. Además, permite desarrollar gran cantidad de funcionalidades sin tener que programarlas de base, lo que agiliza el desarrollo de manera significativa.

Para probar el código desarrollado en Expo se puede hacer uso de una aplicación que permite escanear un código QR que se genera al arrancar el proyecto. Al realizar el escaneo se compila y despliega el código en el teléfono en tiempo real, por lo que si el código se modifica en el ordenador la aplicación reflejará estos cambios inmediatamente en la interfaz móvil. Además, permite exportar cualquier proyecto a una aplicación Android o una aplicación iOS, por lo que puede usarse en la mayoría de teléfonos móviles. Una vez generado el instalador no es necesario el uso de la aplicación de Expo ya que instalará una aplicación completamente independiente.

La principal desventaja de Expo es la obligatoriedad de ceñirse al conjunto de herramientas que ofrece y la obligación de construir el instalador de la aplicación usando los servidores de Expo, lo que puede provocar esperas prolongadas si hay mucho tráfico en ese instante en dichos servidores. [13]

2.5.13 Figma 

Figma permite crear de forma sencilla diagramas o "*mockups*" que representen a modo de prototipo cómo se verán las distintas páginas de una aplicación. Además, incluye la posibilidad de configurar los distintos elementos creados para visualizar el movimiento entre páginas al seleccionar los diferentes elementos creados en el esquema.

2.5.14 MySQL Workbench 

MySQL Workbench es una herramienta visual para el manejo de bases de datos basadas en MySQL. Permite la creación de tablas y relaciones entre ellas de forma manual y sin necesidad de escribir consultas, junto a la opción de generar scripts de la base de datos que haya sido creada para así ser importada a otro lugar. Además, ofrece una interfaz con la que visualizar las tablas de la base de datos de forma intuitiva y la capacidad de ver y modificar su contenido y estructura. También permite ejecutar cualquier consulta que se desee realizar junto a ayudas como resaltado de la sintaxis o autocompletado.

Gracias a que MariaDB está basada en MySQL es posible hacer uso de esta herramienta sin que ocurra ningún problema de compatibilidad.

3

Análisis de requisitos

3.1 Lista de requisitos

Tal y como se ha mencionado anteriormente, antes de comenzar la programación del proyecto se redactó una serie de requisitos a cumplir para ambas aplicaciones. Estos requisitos indican las funcionalidades que se deben cumplir, tanto de forma funcional como no funcional. Los requisitos funcionales son aquellos que indican las funcionalidades concretas, mientras que los no funcionales indican los aspectos o propiedades de las aplicaciones que no pueden describirse en forma de funcionalidades como tal pero que, aun así, deben cumplirse.

3.1.1 Requisitos funcionales de la aplicación web

Usuarios

RF1: Sesión

RF1.1: Los usuarios podrán iniciar sesión.

RF1.2: Los usuarios podrán cerrar sesión.

RF2: Registro

RF2.1: Los usuarios podrán registrarse.

RF3: Gestión de plantas

RF3.1: Los usuarios podrán crear una planta nueva en el invernadero que deseen.

RF3.2: Los usuarios podrán desactivar la planta que deseen si ha sido creada por ellos, lo que hará que no pueda ser accedida por ningún usuario.

RF3.3: Los usuarios podrán asignar una configuración a una planta.

RF3.3.1: Los usuarios podrán seleccionar una configuración ya creada

RF3.3.2: Los usuarios podrán seleccionar una configuración como plantilla y añadir o eliminar campos.

RF3.3.3: Los usuarios podrán crear una nueva configuración.

RF3.4: Los usuarios podrán modificar la configuración asignada a una planta.

RF3.5: Los usuarios podrán acceder a una lista de plantas con sus QRs asociados.

RF3.6: Los usuarios podrán cambiar el invernadero al que está asignada una planta.

RF4: Códigos QR

RF4.1: Los usuarios podrán imprimir los códigos QR que aparezcan en la lista.

RF4.2: Los usuarios podrán imprimir los códigos QR inmediatamente tras crearlos.

RF5: Estadísticas

RF5.1: Los usuarios podrán acceder a una página de estadísticas en la que ver la evolución de las plantas en las configuraciones.

RF5.1: Los usuarios podrán seleccionar la planta, el campo y el tipo de gráfica a imprimir entre gráfica de barras y de líneas.

RF5.1: Los usuarios podrán ver una lista numérica de los datos tomados en un campo.

RF6: Filtrado

RF6.1: Los usuarios podrán filtrar la lista de plantas por autor.

RF6.2: Los usuarios podrán filtrar la lista de plantas por nombre.

RF6.3: Los usuarios podrán filtrar la lista de plantas por fecha de creación .

RF6.4: Los usuarios podrán filtrar la lista de tomas de datos por autor.

RF6.5: Los usuarios podrán filtrar la lista de tomas de datos por configuración.

RF6.6: Los usuarios podrán filtrar la lista de tomas de datos por fecha.

RF6.7: Los usuarios podrán filtrar las plantas por nombre en la página de estadísticas

RF8: Creación de tomas de datos

RF8.1: Los usuarios podrán crear una toma de datos de forma manual asignada a una planta y a un día.

RF8.2: Los usuarios podrán subir archivos a dicha toma de datos.

RF8.2: Los usuarios podrán modificar la toma de datos después de haber sido creada.

Administrador

RF8: Funcionalidades de usuario

RF8.1: Los administradores podrán usar las mismas funcionalidades que se han descrito en los anteriores requisitos.

RF9: Lista de usuarios

RF9.1: Los administradores podrán acceder a una lista de los usuarios.

RF10: Gestión de usuarios

RF10.1: Los administradores podrán tener acceso exclusivo a una página de administrador.

RF10.2: Los administradores podrán desactivar la cuenta de cualquier usuario.

RF10.3: Los administradores podrán convertir en administrador a cualquier usuario.

RF11: Gestión de plantas

RF11.1: Los administradores podrán desactivar cualquier planta.

RF11.2: Los administradores podrán modificar la configuración de cualquier planta.

RF12: Gestión de invernaderos

RF12.1: Los administradores podrán crear un nuevo invernadero.

RF12.2: Los administradores podrán eliminar cualquier invernadero.

RF13: Creación de tomas de datos

RF11.2: Los administradores podrán modificar cualquier toma de datos.

3.1.2 Requisitos no funcionales de la aplicación web

RNF1: Interfaz de Usuario

RNF1.1: La interfaz de usuario será sencilla.

RNF1.2: La interfaz de usuarios utilizará tonos de verde y marrón como colores principales.

RNF2: Seguridad

RNF2.1: Los usuarios no podrán ver la lista de usuarios registrados en el sistema.

RNF2.2 Las contraseñas estarán encriptadas y se les aplicará sal.

RNF2.3 Los usuarios recibirán un correo de autenticación tras el registro con un código que deberán usar para activar su cuenta y poder acceder a la página.

RNF2.4 La página de inicio de sesión estará protegida con ReCaptcha V3.

RNF3: Registro

RNF3.1: Para registrar un usuario será necesario un email, un nombre de usuario y una contraseña de, al menos, 6 caracteres

3.1.3 Requisitos funcionales de la aplicación móvil

Usuarios

RF1: Sesión

RF1.1: Los usuarios podrán iniciar sesión.

RF1.2: Los usuarios podrán cerrar sesión.

RF2: Escaneo de QR

RF2.1: Los usuarios podrán escanear un código QR generado desde la web y crear una nueva toma de datos.

RF2.2: Tras el escaneo de un QR, los usuarios podrán ver una lista con los campos correspondientes a la planta y su configuración actual.

RF2.3: Tras el escaneo de un QR, los usuarios podrán ver la fecha de la última toma de datos.

RF2.4: Tras el escaneo de un QR, los usuarios podrán leer el texto libre contenido en él.

RF3: Creación de toma de datos:

RF3.1: Los usuarios podrán tomar fotos o elegir fotos de la galería en los campos así designados.

RF3.2: Los usuarios podrán introducir texto en los campos así designados.

RF3.3: Los usuarios podrán introducir valores numéricos en los campos así designados.

3.1.4 Requisitos no funcionales de la aplicación móvil

RNF1: Interfaz de Usuario

RNF1.1: La interfaz de usuario será sencilla.

RNF1.2: La interfaz de usuarios utilizará tonos de verde y marrón como colores principales.

3.2 Casos de uso

Los casos de uso proporcionan una definición más concreta de cómo deberían funcionar los requisitos explicando las personas que intervendrán en el cumplimiento de uno, las precondiciones y postcondiciones que se deben cumplir, junto con la serie detallada de acciones a tomar para que se cumpla lo descrito.

3.2.1 Aplicación web

Requisito (RF2.1)	El usuario podrá registrarse en la web.
Resumen	El usuario se registrará en la aplicación y activará su cuenta para así poder acceder a esta.
Actores	Usuario.
Precondiciones	El usuario no está registrado con anterioridad en la aplicación. La contraseña cumple las condiciones adecuadas. El correo cumple las condiciones adecuadas.
Postcondiciones	El usuario queda registrado en la página y su cuenta queda activada.
Curso de acción	1. El usuario abre la página inicial de la web. 2. El usuario pulsa el botón registrase. 3. El usuario introduce los datos adecuados. 4. El usuario accede a su correo electrónico. 5. El usuario pulsa el enlace recibido.
Curso de acción alternativo	3. El usuario introduce datos no adecuados. 4. La página informa sobre el error producido

Requisito (RF1.1)	El usuario podrá iniciar sesión en la web.
Resumen	El usuario introducirá sus datos de inicio de sesión para acceder a las funcionalidades de la página.
Actores	Usuario, Administrador.
Precondiciones	El usuario está registrado con anterioridad en la aplicación.
Postcondiciones	La sesión del usuario queda iniciada y el usuario puede acceder a las

	funcionalidades principales de la página.
Curso de acción	1. El usuario abre la página inicial de la web. 2. El usuario introduce sus datos en los campos adecuados. 3. El usuario entra a la página principal.
Curso de acción alternativo	2. El usuario introduce datos no adecuados. 3. La página informa sobre el error producido.

Requisito (RF3.1)	Crear una nueva planta.
Resumen	El usuario creará una nueva planta en un invernadero concreto y seleccionará la configuración que usará.
Actores	Usuario, Administrador.
Precondiciones	El usuario ha iniciado sesión en la aplicación.
Postcondiciones	La nueva planta queda guardada en la base de datos. La configuración queda asociada a la planta. Cualquier nueva configuración o campos nuevos quedarán guardados en la base de datos.
Curso de acción	1. El usuario pulsa el botón de crear planta. 2. El usuario introduce el nombre y descripción y selecciona un invernadero. 3. El usuario selecciona una configuración de las ya creadas. 4. El usuario puede ver un resumen de la planta creada junto con el código QR.
Curso de acción alternativo	3. El usuario selecciona el botón de crear una nueva configuración. 4. El usuario elige una configuración como base, añade campos individuales o crea campos nuevos. 5. El usuario introduce el nombre de la configuración. 6. El usuario puede ver un resumen de la planta creada junto con el código QR.

Requisito (RF8.1)	Crear nueva toma de datos para una planta.
Resumen	El usuario seleccionará una planta y creará una nueva toma de datos para la configuración seleccionada.
Actores	Usuario, Administrador.
Precondiciones	El usuario ha iniciado sesión en la aplicación. Hay al menos una planta creada.
Postcondiciones	Se crea la nueva toma de datos.

Curso de acción	1. El usuario pulsa el botón de lista de plantas. 2. El usuario selecciona una planta. 3. El usuario pulsa el botón de introducir nuevos datos. 4. El usuario introduce los datos deseados. 5. El usuario pulsa el botón de guardar. 6. Los datos quedan guardados.

Requisito (RF 8.2)	Editar toma de datos
Resumen	El usuario accederá a la lista de datos tomados de una planta, seleccionará una concreta y editará los datos tomados.
Actores	Usuario, Administrador.
Precondiciones	El usuario ha iniciado sesión en la aplicación. Hay al menos una planta creada. Hay al menos una toma de datos hecha en la planta. La toma de datos ha sido realizada por el usuario o el usuario es un administrador.
Postcondiciones	Los datos quedan actualizados.
Curso de acción	1. El usuario pulsa el botón de lista de plantas. 2. El usuario selecciona una planta. 3. El usuario pulsa el botón de ver datos tomados. 4. El usuario selecciona una toma de datos. 5. El usuario pulsa el botón de editar datos. 6. El usuario introduce los nuevos datos. 7. El usuario pulsa el botón de guardar. 8. Los datos quedan editados.

Requisito (RF3.4)	Editar configuración seleccionada
Resumen	El usuario editará la configuración asignada a una planta.
Actores	Usuario, Administrador.
Precondiciones	El usuario ha iniciado sesión en la aplicación. Hay al menos una planta creada.
Postcondiciones	Los configuración asignada a la planta queda modificada.
Curso de acción	1. El usuario pulsa el botón de lista de plantas. 2. El usuario selecciona una planta. 3. El usuario pulsa el botón de ver editar configuración. 4. El usuario selecciona una configuración de la lista de configuraciones ya creadas. 5. El usuario pulsa el botón de seleccionar. 6. La configuración queda modificada.

Curso de acción alternativo	4. El usuario selecciona el botón de crear una nueva configuración. 5. El usuario elige una configuración como base, añade campos individuales o crea campos nuevos. 6. El usuario introduce el nombre de la configuración. 7. La configuración queda modificada.

Requisito (RF4.2)	Imprimir código QR.
Resumen	El usuario podrá imprimir el código QR de una planta que haya seleccionado de la lista en cualquier momento.
Actores	Usuario, Administrador.
Precondiciones	El usuario ha iniciado sesión en la aplicación. Hay al menos una planta creada.
Postcondiciones	El código QR y los detalles de la configuración se añaden a la cola de impresión.
Curso de acción	1. El usuario pulsa el botón de lista de plantas. 2. El usuario selecciona una planta. 3. El usuario pulsa el botón de imprimir QR. 4. El usuario pulsa el botón de imprimir en la ventana emergente.

Requisito (RF3.2)	Desactivar planta
Resumen	El usuario podrá desactivar una planta creada por sí mismo
Actores	Usuario, Administrador.
Precondiciones	El usuario ha iniciado sesión en la aplicación. Hay al menos una planta creada por el usuario o el usuario debe ser un administrador.
Postcondiciones	La planta queda desactivada y deja de aparecer en la lista.
Curso de acción	1. El usuario pulsa el botón de lista de plantas. 2. El usuario selecciona una planta. 3. El usuario pulsa el botón de desactivar planta.

Requisito (RF3.6)	Cambiar el invernadero al que está asignada una planta.
Resumen	El usuario podrá cambiar el invernadero al que está asignada una planta a partir de la lista de invernaderos ya creados.
Actores	Usuario, Administrador.
Precondiciones	El usuario ha iniciado sesión en la aplicación. Hay al menos una planta creada por el usuario o el usuario debe ser

	un administrador.
Postcondiciones	El invernadero al que está asignada la planta es cambiado.
Curso de acción	1. El usuario pulsa el botón de lista de plantas. 2. El usuario selecciona una planta. 3. El usuario pulsa el botón de cambiar invernadero. 4. El usuario selecciona el invernadero deseado de la lista de invernaderos. 5. El usuario pula el botón guardar. 6. El invernadero asignado queda actualizado.

Requisito (RF5)	Mostrar estadísticas de una planta
Resumen	El usuario podrá ver las estadísticas de los campos con valores numéricos creados a lo largo del tiempo de una planta.
Actores	Usuario, Administrador.
Precondiciones	El usuario ha iniciado sesión en la aplicación. Hay al menos una planta creada. La planta tiene datos tomados en campos con valores numéricos.
Postcondiciones	Se muestran una gráfica o na lista con las estadísticas seleccionadas.
Curso de acción	1. El usuario pulsa el botón de estadísticas. 2. El usuario selecciona una planta. 3. El usuario selecciona un campo. 4. El usuario selecciona un tipo de gráfica o lista 5. Las estadísticas se muestran por pantalla.

Requisito (RF10.2)	Desactivar usuario
Resumen	El administrador podrá ver una lista de todos los usuarios registrados en la página y podrá desactivar la cuenta del que desee.
Actores	Administrador.
Precondiciones	El administrador ha iniciado sesión en la aplicación. Hay al menos un usuario creado. El usuario no ha sido desactivado previamente.
Postcondiciones	El usuario queda desactivado.
Curso de acción	1. El administrador pulsa el botón de página del administrador. 2. El administrador busca un usuario. 3. El usuario pulsa el botón de desactivar ese usuario. 4. El usuario queda desactivado.

Requisito (RF10.3)	Convertir usuario en administrador
Resumen	El administrador podrá ver una lista de todos los usuarios registrados en la página y podrá convertir en administrador al que desee.
Actores	Administrador.
Precondiciones	El administrador ha iniciado sesión en la aplicación. Hay al menos un usuario creado. El usuario no es ya un administrador.
Postcondiciones	El usuario queda asignado al rol del administrador.
Curso de acción	1. El administrador pulsa el botón de página del administrador. 2. El administrador busca un usuario. 3. El usuario pulsa el botón de convertir en administrador. 4. El usuario obtiene el rol de administrador.

Requisito (RF12.1)	Crear nuevo invernadero.
Resumen	El administrador podrá crear un nuevo invernadero al que asignar plantas.
Actores	Administrador.
Precondiciones	El administrador ha iniciado sesión en la aplicación.
Postcondiciones	El nuevo invernadero queda creado.
Curso de acción	1. El administrador pulsa el botón de página del administrador. 2. El administrador introduce un nuevo nombre para un invernadero. 3. El usuario pulsa el botón de crear nuevo invernadero. 4. El nuevo invernadero queda creado.

Requisito (RF12.2)	Desactivar invernadero.
Resumen	El administrador podrá desactivar un invernadero ya creado.
Actores	Administrador.
Precondiciones	El administrador ha iniciado sesión en la aplicación. Hay al menos un invernadero creado.
Postcondiciones	El invernadero seleccionado queda desactivado.
Curso de acción	1. El administrador pulsa el botón de página del administrador. 2. El administrador selecciona un invernadero de la lista. 3. El usuario pulsa el botón de desactivar invernadero. 4. El invernadero queda desactivado.

Requisito (RF1.2)	Cerrar sesión.
Resumen	El usuario podrá cerrar su sesión.
Actores	Usuario, Administrador.
Precondiciones	El usuario ha iniciado sesión en la aplicación.
Postcondiciones	La sesión del usuario queda cerrada.
Curso de acción	1. El usuario pulsa el botón de cerrar sesión. 2. La sesión del usuario queda cerrada.

3.2.2 Aplicación móvil

Requisito (RF1.1)	El usuario podrá iniciar sesión en la aplicación móvil.
Resumen	El usuario introducirá sus datos de inicio de sesión para acceder a las funcionalidades de la aplicación.
Actores	Usuario, Administrador.
Precondiciones	El usuario está registrado con anterioridad en la aplicación web.
Postcondiciones	La sesión del usuario queda iniciada y el usuario puede acceder a las funcionalidades principales de la aplicación.
Curso de acción	1. El usuario abre la aplicación móvil. 2. El usuario introduce sus datos en los campos adecuados. 3. El usuario entra a la página principal.
Curso de acción alternativo	2. El usuario introduce datos no adecuados. 3. La aplicación informa sobre el error producido.

Requisito (RF 3)	El usuario podrá realizar una toma de datos con la aplicación móvil.
Resumen	El usuario podrá realizar una toma de datos con la aplicación móvil tras escanear el código QR de una planta.
Actores	Usuario, Administrador.
Precondiciones	El usuario ha iniciado sesión en la aplicación. Hay al menos una planta creada en la página web. Se dispone de un código QR para escanear.
Postcondiciones	La toma de datos queda guardada en la base de datos y puede verse al acceder posteriormente en la web.
Curso de acción	1. El usuario pulsa el botón escanear código QR. 2. El usuario escanea el código QR de una planta. 3. El usuario pulsa el botón de crear nueva toma de datos. 4. El usuario introduce los valores numéricos y textuales que

	quiera guardar. 5. El usuario realiza las fotos que desee en los campos de imagen. 6. El usuario pulsa el botón de guardar datos. 7. Los datos quedan guardados.
Curso de acción alternativo	5. El usuario selecciona las imágenes ya tomadas que desee de la galería de su teléfono. 6. El usuario pulsa el botón de guardar datos. 7. Los datos quedan guardados.

4

Diseño y arquitectura

4.1 *Mockups*

Antes de comenzar el desarrollo de la aplicación web se realizaron varios prototipos con la herramienta Figma para facilitar el desarrollo futuro y obtener una idea general de cómo se debería ver la web. A continuación se muestran capturas de dichos prototipos:

Figura 4.1: *Mockups* de inicio de sesión y registro

En la capturas superiores de la figura 4.1 se muestran los *mockups* de las páginas de inicio de sesión y registro. Inicialmente se propuso la idea de permitir el inicio de sesión con usuarios locales y con Google mediante OauthV2. Sin embargo, debido a problemas de compatibilidad con la versión de Spring Security y de conveniencia de manejo de usuarios locales se optó por sólo permitir usuarios registrados de forma local en la página.

En las capturas inferiores se muestran la página principal a la que se redirige al usuario una vez ha iniciado sesión y la página que se muestra cuando se pulsa en crear una nueva planta. Ya que la lista de requisitos se redactó al mismo tiempo que se hacían los prototipos, al principio no se incluyó el requisito de manejar varios invernaderos, por lo que no aparece la opción de selección de invernadero al crear la planta en la imagen. Posteriormente, se añadió el manejo de invernaderos durante el desarrollo a la lista de requisitos.

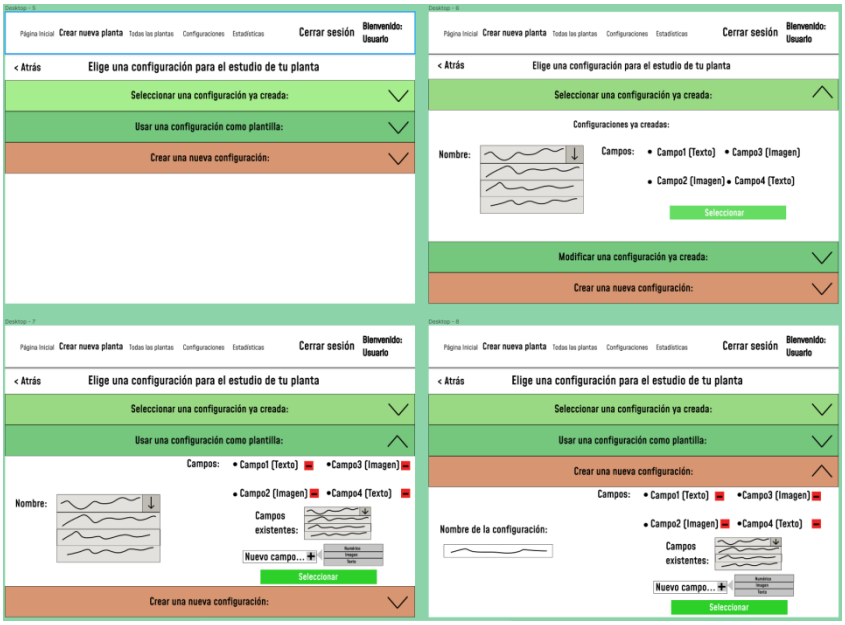


Figura 4.2: *Mockups* para la selección de configuración

Figura 4.3: *Mockups* de finalización de creación y lista de plantas

En la figura 4.2 aparece el proceso de selección de la configuración de una planta una vez se ha seleccionado su nombre, descripción e invernadero. Al comienzo se plantearon las tres opciones distintas que aparecen. Sin embargo, durante el desarrollo se decidió combinar las dos últimas opciones en una sola añadiendo la opción de usar como plantilla en la opción de crear una nueva configuración. De esta forma se simplifica el proceso de creación y se eliminan opciones innecesarias.

En la captura superior de la figura 4.3 se muestra la página final de creación de la planta. En la versión final también se muestra el invernadero al que se ha asignado.

En la captura inferior de dicha figura se muestra la lista de plantas creadas y las opciones que aparecen cuando se selecciona una. Durante el desarrollo se añadieron las opciones de introducir datos y de cambiar el invernadero al que pertenece.

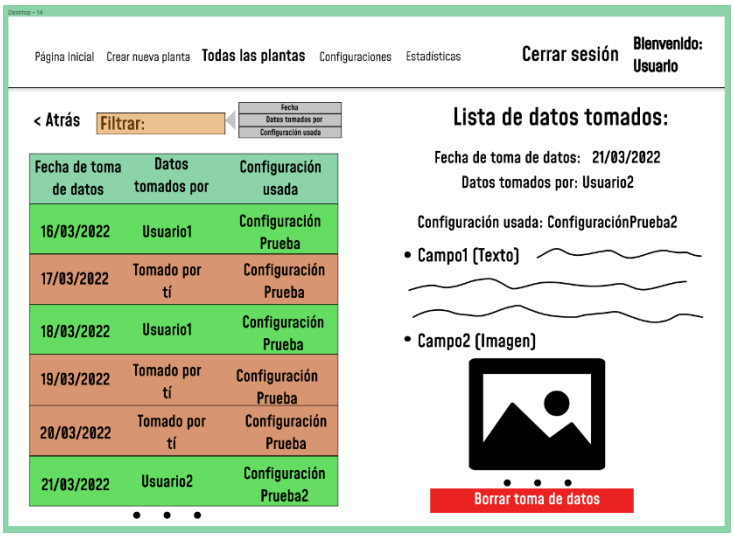


Figura 4.4: *Mockup* de la lista de datos tomados de una planta

En la figura 4.4 se muestra la página de las tomas de datos realizadas a una planta y un ejemplo de toma de datos. Al principio se planteó permitir borrar una toma de datos pero durante el desarrollo se decidió simplemente desactivarla y hacer que deje de aparecer sin borrarla de la base de datos, al igual que con las plantas. Además, se incluyó el nuevo requisito de permitir modificar los datos introducidos en una toma de datos.

En las capturas de la figura 4.5 se muestra la página de cambio de configuración de una planta. Durante la fase de creación de los prototipos se pensó en no usar la opción de usar cómo plantilla. En la versión final se usó la misma página usada en la creación de plantas.

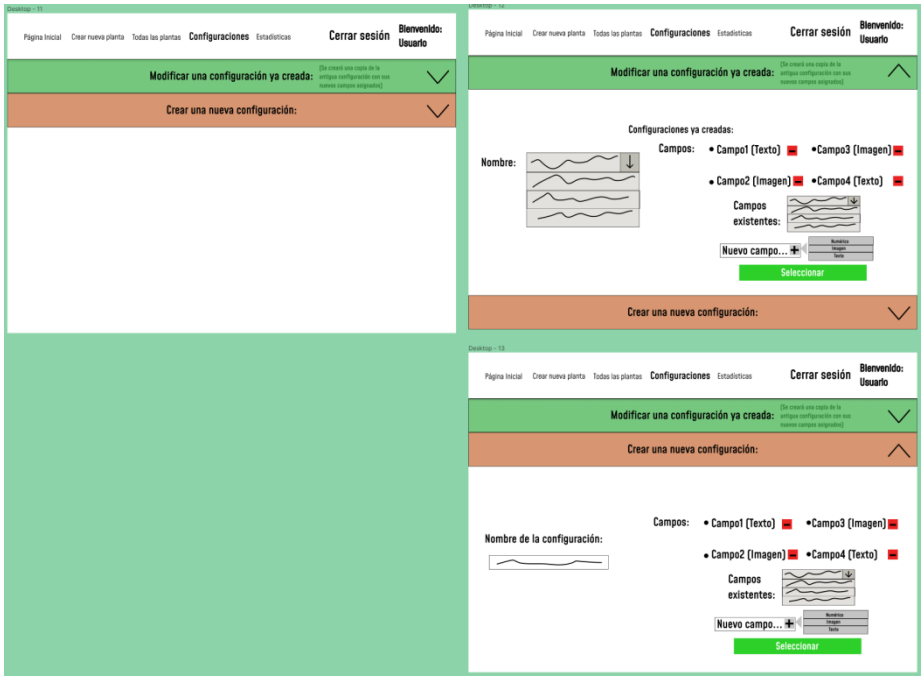


Figura 4.5: *Mockups* para el cambio de configuración de una planta

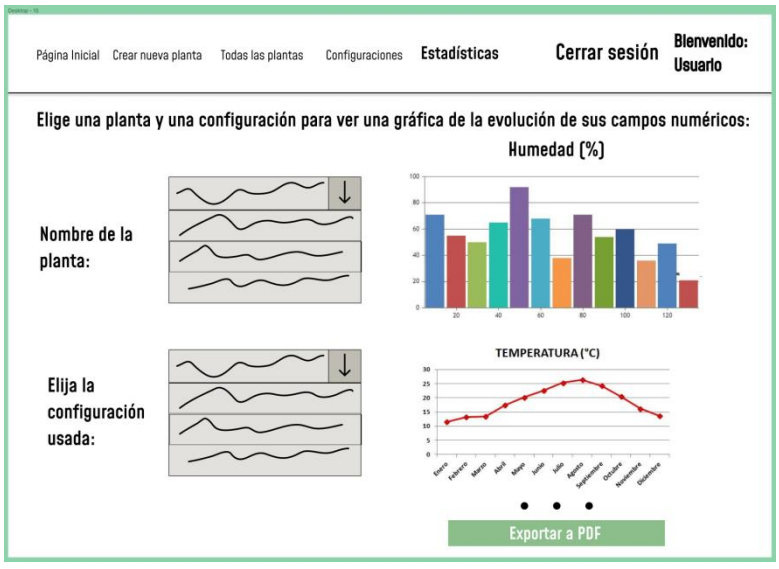


Figura 4.6: *Mockup* de la página de estadísticas

Finalmente, en la figura 4.6 se muestra la página de las estadísticas de una planta. Primero, se planteó seleccionar una de las configuraciones que ha usado una planta y mostrar todos los gráficos correspondientes a los campos de esa configuración. Al final se decidió seleccionar un solo campo de la lista total de campos en los que se han registrado datos que haya usado la planta y elegir el tipo de gráfico a mostrar. También se optó por no implementar la opción de exportar a PDF ya que tras hablarlo con el tutor no se vio necesario pudiéndose optar por capturar desde el navegador la propia página a PDF.

Enlace al documento: https://www.figma.com/file/AHkkaEYs2R3fjn07T26h6v/TFG-Web?node-id=0%3A1

4.2 Arquitectura

La arquitectura de la aplicación web se basa en un diseño clásico en el que el cliente se comunica con el servidor, que ha sido probado usando *Tomcat* en un servidor local para realizar despliegues de forma rápida. El servidor se comunica con la base de datos MariaDB y se divide en *frontend* y *backend*.

El *frontend* se ha escrito en JSP usando como lenguajes principales HTML, Java y JavaScript. Java se ha usado para realizar la carga inicial de forma correcta entre diferentes enlaces de la web, mientras que JavaScript se ha usado para programar todas las funcionalidades interactivas de la página junto a la conexión con el *backend*.

El *backend* consta de varios controladores Spring a los que se encomienda la carga de los diferentes JSP y de una API REST que se ocupa de recibir las peticiones y datos del *frontend* además de realizar el tratamiento necesario de esos datos junto con las llamadas correspondientes a la base de datos, para más tarde enviar un mensaje de respuesta al *frontend o a la aplicación móvil* con los resultados.

La aplicación móvil usa un *frontend* compuesto de distintos componentes proporcionados por React Native y Expo que son manejados por código JavaScript de React Native. Este código también se encarga de realizar peticiones al *backend* de la aplicación web.

4.3 Diagramas de la arquitectura de los sistemas

A continuación se muestran diagramas de secuencia de todos los casos de uso descritos junto a una explicación textual de cómo funcionan los sistemas que hacen funcionar cada uno de dichos casos de uso. Algunos de los más importantes también incluyen un diagrama de caso de uso.

4.3.1 Registro

En el proceso de registro el usuario introduce en el formulario su nombre de usuario, su email y su contraseña. El *frontend* los envía al *backend*, que se encarga de comprobar si ese nombre o ese email ya existen en la base de datos. Si ya existen se devuelve un mensaje de error que se muestra por pantalla. Si no se produce ningún error, el sistema crea el usuario en la base de datos junto con un *token* que deberá ser activado a través de un enlace en el correo del usuario para que así pueda activar su cuenta.

Una vez creados el usuario y el *token* se genera el correo y la página muestra que el registro ha sido correcto.

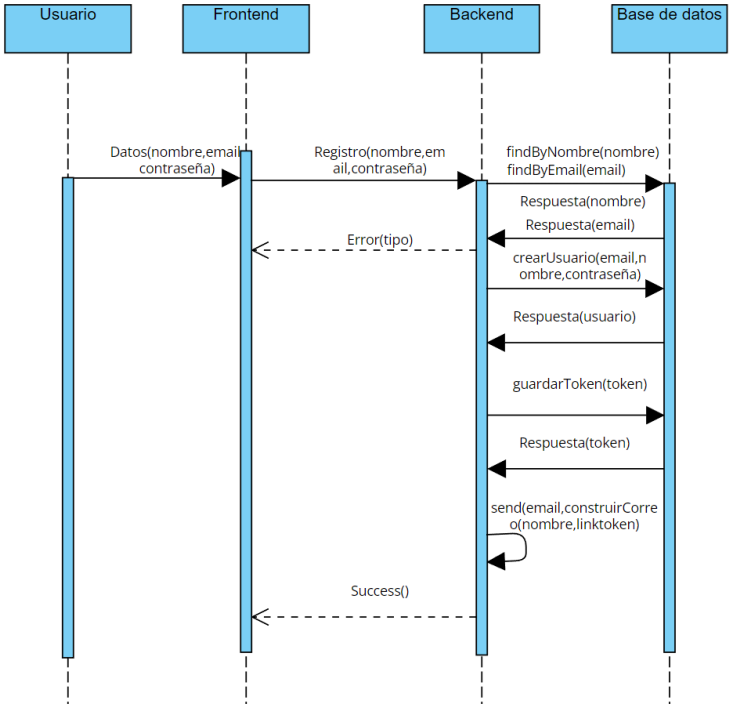


Figura 4.7: Diagrama de secuencia para el registro de un usuario

4.3.2 Inicio de sesión

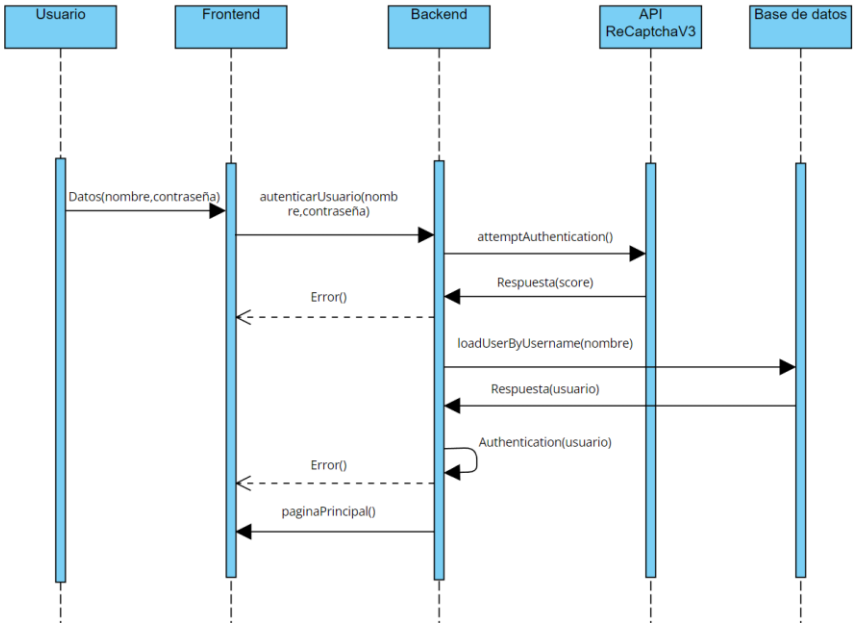


Figura 4.8: Diagrama de secuencia para el inicio de sesión

En el proceso de inicio de sesión el usuario escribe su nombre de usuario y contraseña en la página correspondiente. Estos datos son tratados por Spring Security, que primero aplica un filtro a la autenticación en el que se envía una petición a la API ReCaptchaV3 de Google, que devuelve una puntuación con respecto a la petición que hemos hecho. Esta puntuación irá de 0 a 1 y será más baja según haya mayor probabilidad de que la petición no haya sido hecha

por un humano. Si la puntuación es menor a 0.5 el sistema no dejará entrar al usuario y mostrará un error.

Una vez realizado el filtro, se hace una petición a la base de datos para buscar el usuario que tenga ese nombre y así comprobar si las contraseñas coinciden. Si no coinciden o no existe el usuario se mostrará un error. Si el proceso funciona de manera correcta la sesión del usuario quedará abierta y este podrá ver la página principal de la web.

4.3.3 Crear una nueva planta

Para crear la planta el usuario tendrá que elegir un nombre, escribir un texto para describirla y elegir el invernadero al que pertenecerá. Una vez seleccionados los datos, la interfaz cambiará para permitir al usuario elegir la configuración de la planta. Se presentarán dos opciones: elegir una configuración ya creada y crear una configuración nueva. En el caso de que se seleccione una configuración ya existente se pasará a crear la planta con los campos asignados. En el caso de que se decida crear una configuración nueva, el usuario tendrá que elegir un nombre para la nueva configuración y elegir los campos que tendrá. Una vez elegidos, esta se creará y después se creará la planta con esta nueva serie de campos.

La creación de la planta se realizará en dos pasos, el primero consistirá en su creación en la base de datos, mientras que el segundo consistirá en asignarle la dirección del código QR que se ha generado usando la nueva id que devuelva la base de datos. El código QR llevará esta id como nombre de archivo para que se pueda leer de forma más sencilla en el futuro.

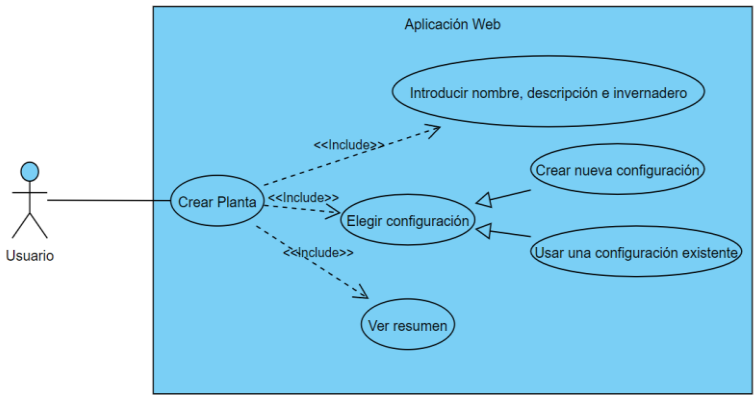


Figura 4.9: Diagrama de caso de uso para la creación de una nueva planta

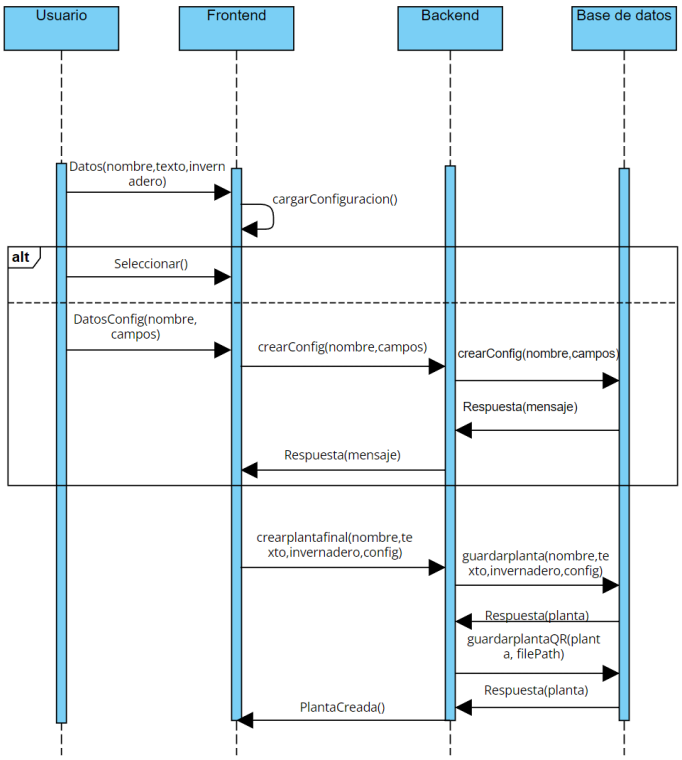


Figura 4.10: Diagrama de secuencia para la creación de una nueva planta

4.3.4 Crear una nueva toma de datos para una planta

Para realizar una nueva toma de datos el *frontend* hace una petición al *backend*, que se encarga de recoger todos los campos asignados a la configuración de la planta usando su id para que así el *frontend* pueda cargar una lista de entradas de datos para dichos campos.

Una vez se muestren por pantalla, el usuario deberá introducir los datos correspondientes a los que haya tomado en el invernadero, pudiendo ser imágenes, textos o valores numéricos, dependiendo de los tipos de campos que se hayan seleccionado en su configuración y, finalmente, pulsará el botón de introducir datos. Una vez pulsado se enviarán dichos datos al *backend*, que se encargará de guardarlos como elementos datosesion en la base de datos. Una vez guardados el *backend* devolverá un mensaje de 'ok' sí todo ha salido bien o un mensaje vacío que el *frontend* interpretará cómo un error.

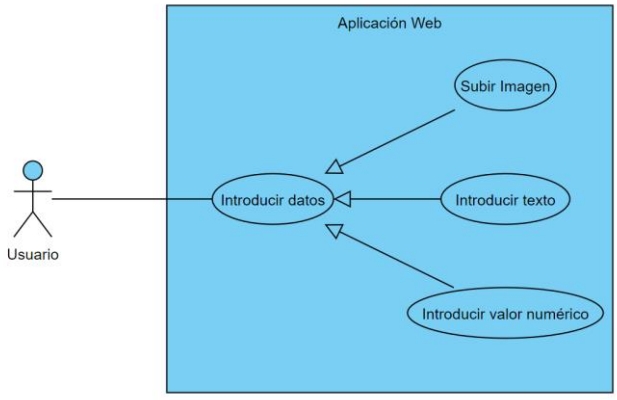


Figura 4.11: Diagrama de caso de uso para la creación de toma de datos de una planta

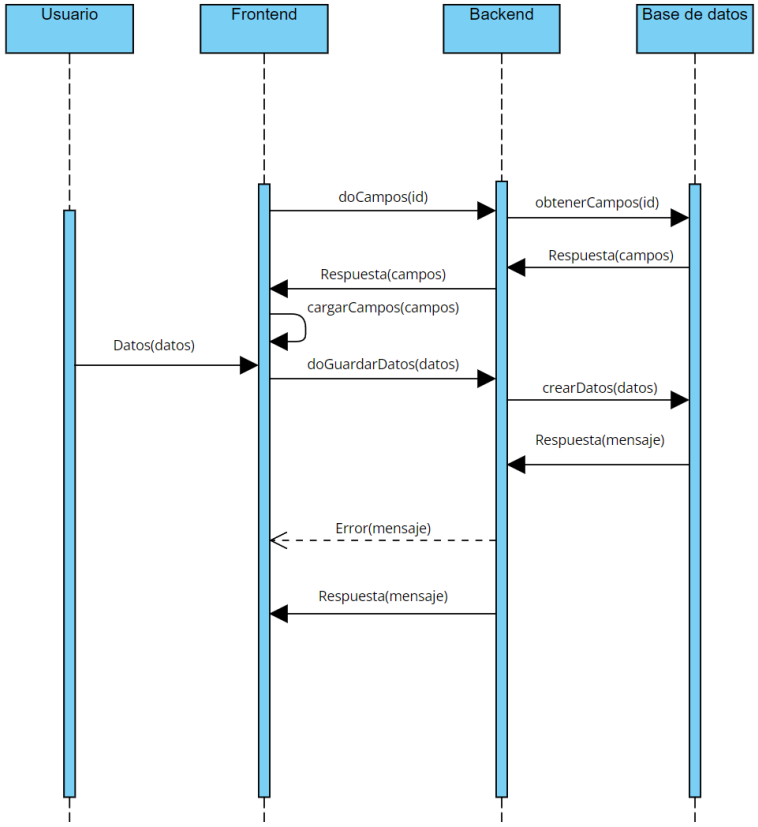


Figura 4.12: Diagrama secuencia para la creación de toma de datos de una planta

4.3.5 Editar datos tomados

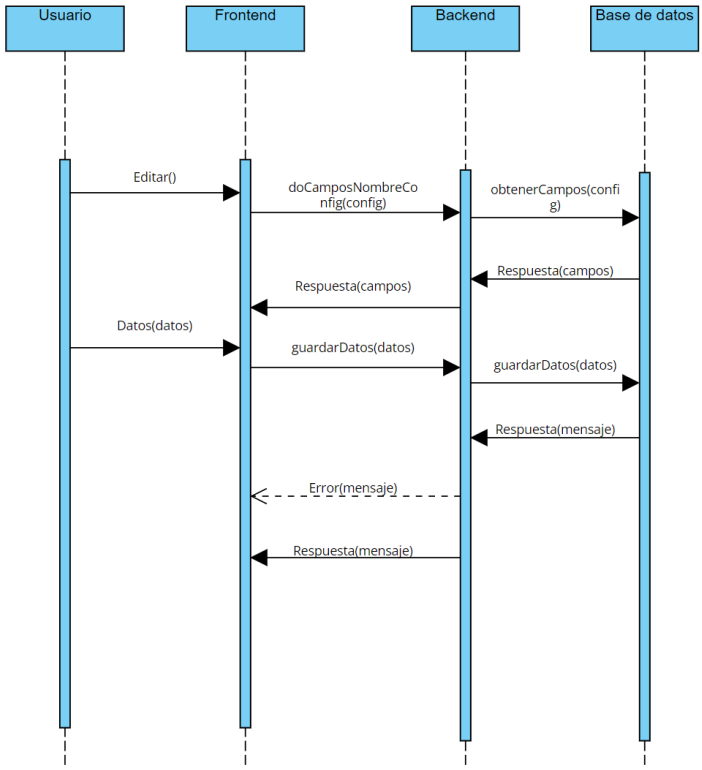


Figura 4.13: Diagrama secuencia para la edición de una toma de datos de una planta

Si el usuario está viendo una toma de datos de una planta puede pulsar el botón de editar. Esto hará que el *frontend* llame al *backend* para obtener una lista de todos los campos asignados a la configuración que usó esa toma de datos. Una vez se obtienen los campos de la base de datos estos son devueltos al *frontend*, que carga campos con los valores ya asignados y campos vacíos para los campos que no hubiesen sido tomados en un principio.

El usuario podrá modificar cualquiera de los datos y pulsar el botón guardar. Esto hará que se llame al *backend* para guardar los datos en la base de datos, sobrescribiendo los datos antiguos. La base de datos devolverá un mensaje de «ok» o de «error» dependiendo de si se han guardado de forma correcta.

4.3.6 Modificar la configuración de una planta

Para acceder a la página de cambio de configuración el usuario pulsará el botón de la planta correspondiente en la lista de plantas, lo que hará que el *frontend* llame a un controlador que se encargará de invocar al nuevo JSP.

Para modificar la configuración de una planta se hace uso del mismo JSP que se usó a la hora de seleccionar la configuración de la planta cuando se creó. Sin embargo, esta vez se llama a un método de la API REST encargado de modificar la configuración que hayamos seleccionado o creado.

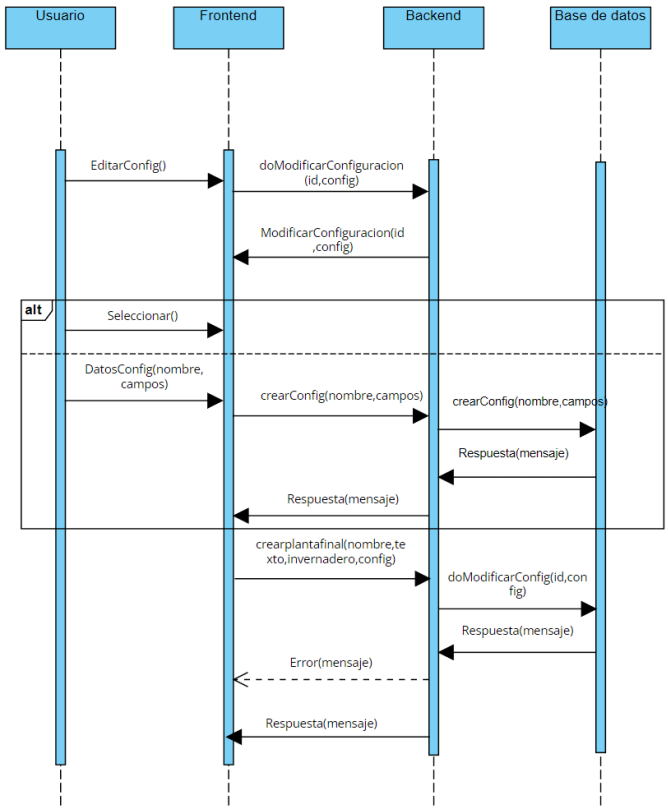


Figura 4.14: Diagrama de secuencia para la modificación de la configuración de una planta

4.3.7 Imprimir código QR

Para imprimir el código QR correspondiente a una planta habrá que pulsar el botón de Imprimir QR que se encuentra en el menú que aparece al seleccionar una planta de la lista. Una vez pulsado el botón de imprimir aparecerá una ventana del navegador en la que se podrá ver una vista previa de la hoja a imprimir, donde aparecerá el QR, su nombre, configuración e invernadero.

Para imprimir la hoja habrá que seleccionar la impresora correspondiente y pulsar en Aceptar.

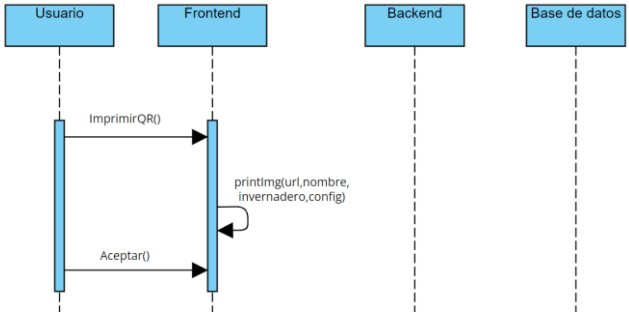


Figura 4.15: Diagrama secuencia para la impresión de un código qr

4.3.8 Desactivar una Planta

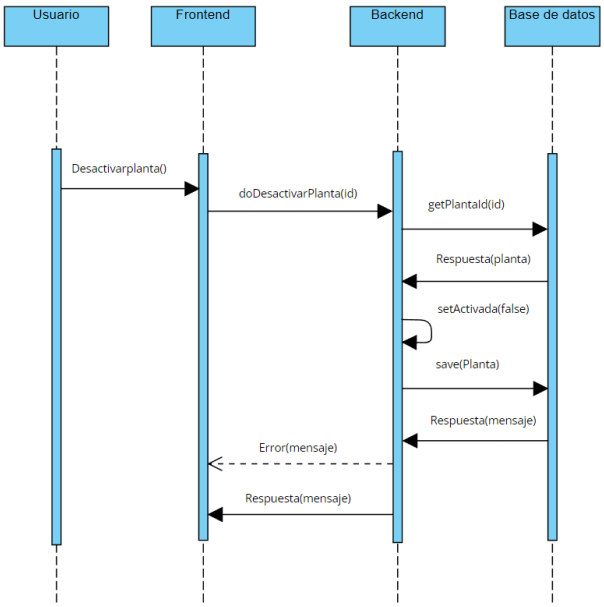


Figura 4.16: Diagrama de secuencia para la desactivación de una planta

Para desactivar la planta habrá que seleccionar de nuevo una planta de la lista. A continuación habrá que pulsar el botón de Desactivar Planta resaltado en rojo para que así deje de aparecer en dicha lista. Una vez pulsado el botón se producirá una llamada al *backend* con la id de la planta. El *backend* se encargará de recibir la planta correspondiente a la id proporcionada, cambiar su atributo a desactivada y guardarla en la base de datos, enviando de vuelta un mensaje que indica si el proceso se ha llevado a cabo de forma correcta.

Cabe decir que una planta sólo podrá desactivarse por su creador o por un administrador.

4.3.9 Cambiar el invernadero de una planta

Para cambiar el invernadero de una planta habrá que pulsar el botón de cambiar invernadero tras seleccionar una de la lista. Una vez pulsado se llamará al *backend* para obtener una lista de todos los invernaderos, que se mostrarán en una lista de selección nueva en la página.

A continuación, el usuario deberá seleccionar un invernadero de la lista y pulsar el botón Seleccionar. El *frontend* enviará la petición para cambiar el invernadero pasando al *backend* la id de la planta y el nombre del nuevo invernadero. El *backend* pedirá a la base de datos la planta a la que le corresponde esa id y al recibirla cambiará su invernadero por el que se le ha pasado. A continuación se guardará en la base de datos y se recibirá una respuesta que indicará si ha habido un error o no.

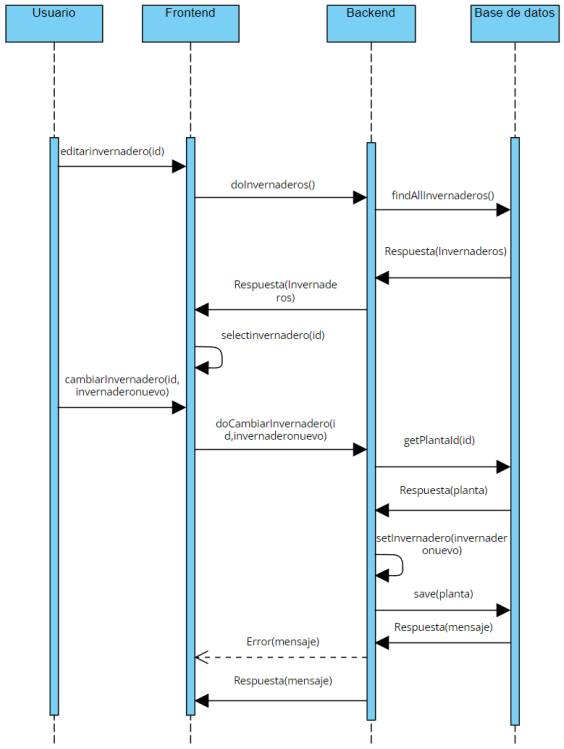


Figura 4.17: Diagrama de secuencia para la modificación del invernadero asignado a una planta

4.3.10 Ver las estadísticas de los campos de una planta

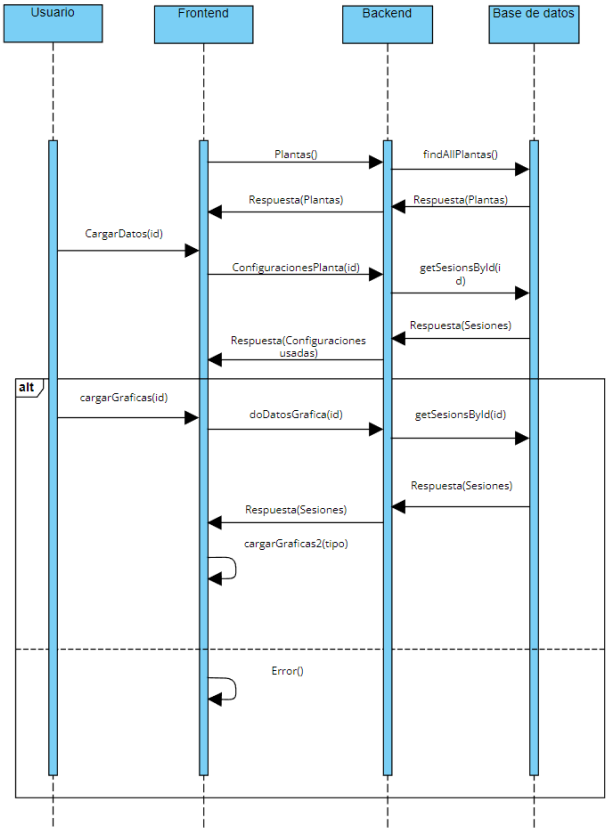


Figura 4.18: Diagrama de secuencia para el acceso a las estadísticas de una planta

Para ver las estadísticas de los campos de una planta el *frontend* se encarga de llamar al *backend* para recibir una lista de todas las plantas que han sido creadas. Estas plantas aparecerán para ser seleccionadas en una lista. Una vez se haya seleccionado una planta por parte del Usuario el *frontend* vuelve a consultar al *backend* para recibir una lista de las configuraciones usadas en las tomas de datos. Si la lista de configuraciones usadas está vacía aparecerá por pantalla un texto indicando que esa planta aún no tiene datos tomados.

Si la planta tiene datos tomados aparecerá otra lista de selección que permitirá seleccionar el tipo de lista que aparecerá, ofreciendo la posibilidad de una lista de datos por escrito, una gráfica de barras o una gráfica de líneas.

Una vez seleccionado el tipo a representar, el *frontend* llama al *backend* para cargar la lista de datos tomados y estos se cargan en el tipo de gráfica o lista seleccionado.

4.3.11 Desactivar un usuario

Para desactivar un usuario, un administrador pulsará el botón correspondiente al usuario deseado en la página del administrador. Al pulsar este botón el *frontend* enviará una petición de desactivar el usuario con una id concreta al *backend*. El *backend* se encargará de tomar el usuario en la base de datos, modificar su estado a desactivado y de guardarlo de nuevo. Finalmente, se recibirá un mensaje de la base de datos y se enviará un mensaje al *frontend* indicando si ha habido algún error en el proceso.

Cabe indicar que cuando se desactiva un usuario aparecerá un nuevo botón para reactivarlo. Si el administrador quisiera volver a activar dicho usuario simplemente debería pulsar el botón, lo que haría que se realizase un proceso similar al anterior, solo que activándolo de nuevo.

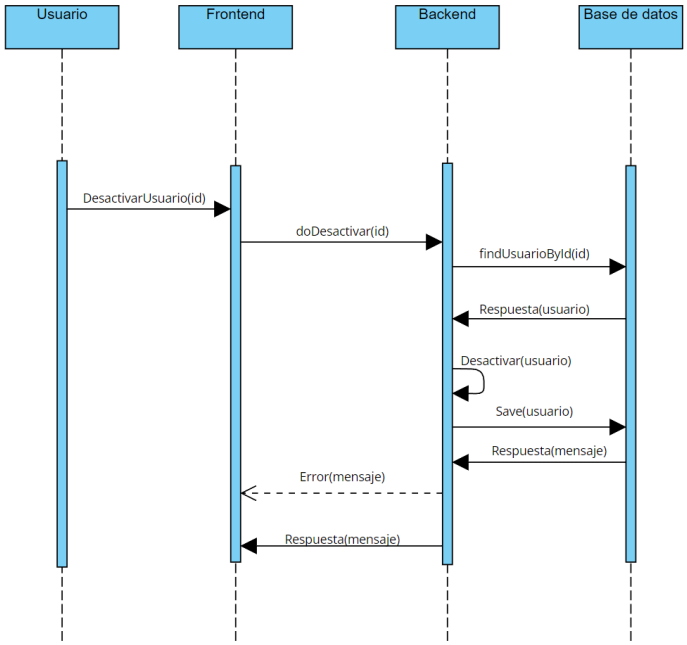


Figura 4.19: Diagrama de secuencia para la desactivación de un usuario

4.3.12 Convertir un usuario en administrador

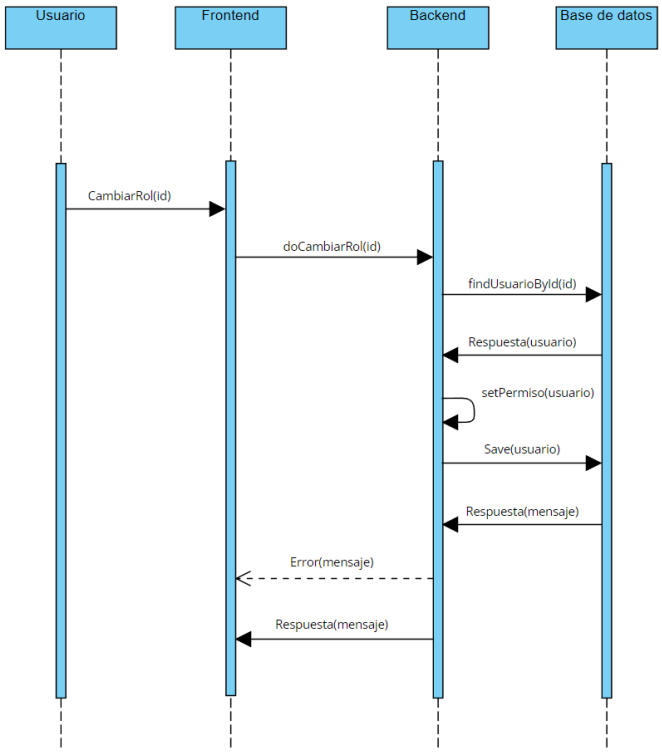


Figura 4.20: Diagrama de secuencia para la conversión de un usuario a administrador

Para convertir un usuario en administrador se deberá pulsar el botón de cambiar rol del usuario deseado en la página del administrador. Cuando se pulse el botón, se realizará una petición al *backend* desde el *frontend* en la que se enviará la id del usuario a cambiar. El *backend* hará una petición para recibir ese usuario de la base de datos y modificará su rol. A continuación, se guardará dicho usuario en la base de datos de nuevo y se enviará un mensaje al *frontend* indicando si se ha producido un error o si el proceso se ha realizado de forma correcta.

4.3.13 Crear un nuevo invernadero

Para crear un nuevo invernadero se introducirá un nombre en un cuadro de texto indicado de la página del administrador y se pulsará un botón de crear invernadero. Una vez pulsado, el *frontend* enviará una petición de creación de invernadero al *backend* enviándole el nombre introducido anteriormente. El *backend* se encargará de crear un nuevo invernadero y de guardarlo en la base de datos. Una vez guardado se enviará un mensaje al *frontend* indicando si la creación se ha realizado de forma correcta.

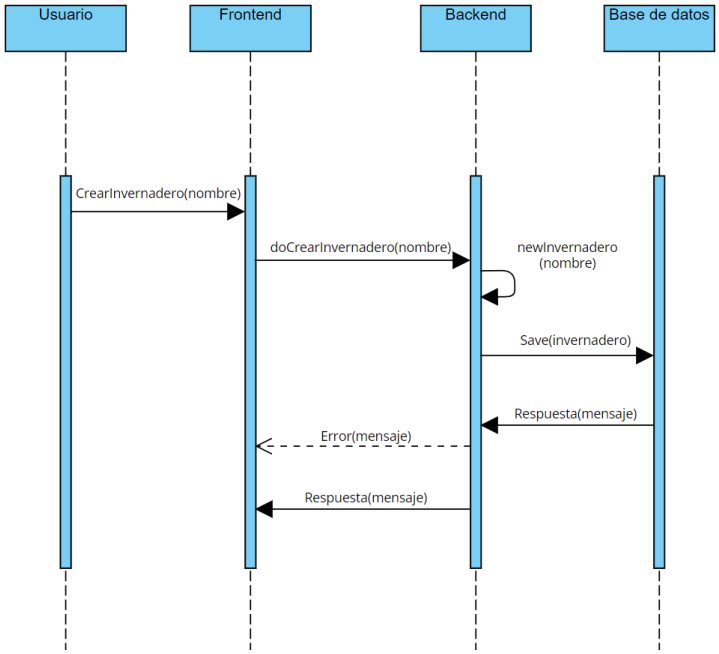


Figura 4.21: Diagrama de secuencia para la creación de un nuevo invernadero

4.3.14 Desactivar un invernadero

Para desactivar un invernadero –hacer que no se pueda usar de nuevo para crear una planta- habrá que seleccionar uno de la lista de invernaderos en la página del administrador y pulsar el botón Desactivar invernadero. Una vez pulsado el *frontend* hará una llamada al *backend* enviándole el nombre del invernadero. El *backend* hará una petición a la base de datos y desactivará el invernadero que reciba. Una vez desactivado lo enviará de nuevo a la base de datos para ser guardado y se enviará un mensaje al *frontend* indicando si el proceso ha tenido éxito.

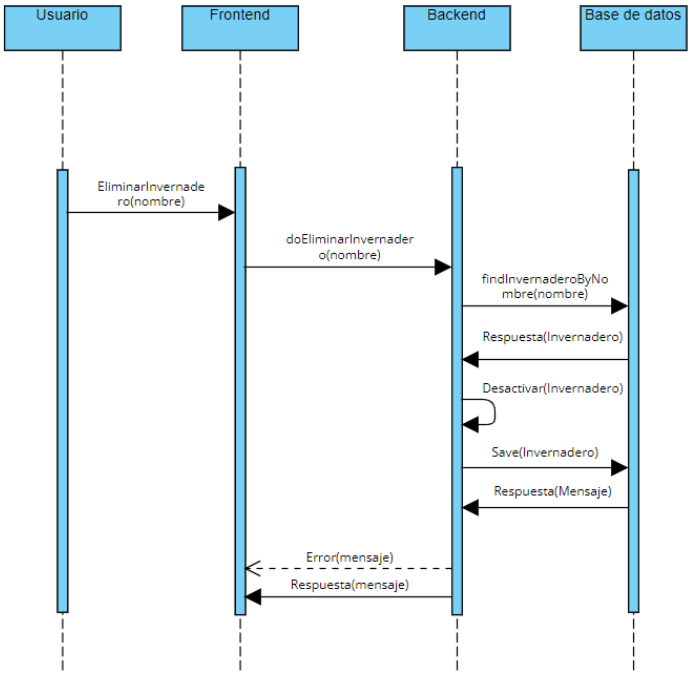


Figura 4.22: Diagrama de secuencia para la desactivación de un invernadero

4.3.15 Cerrar sesión

Para cerrar la sesión se pulsará el botón correspondiente en la barra de navegación superior de la página. El *frontend* se encargará de hacer una petición al proceso de cerrar sesión de SpringSecurity que devolverá al usuario a la página de inicio de sesión, donde podrá volver a entrar con su cuenta o acceder a la página de registro.

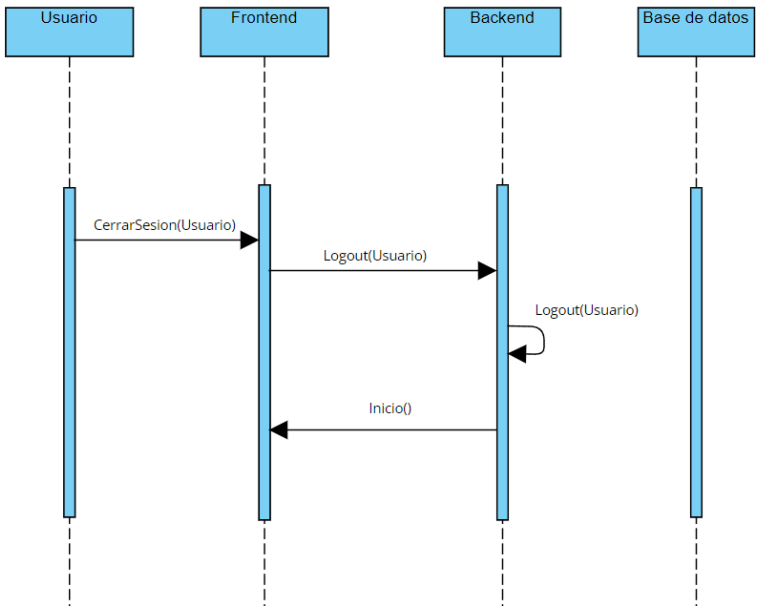


Figura 4.23: Diagrama de secuencia para el cierre de sesión

4.3.16 Iniciar sesión en la aplicación móvil

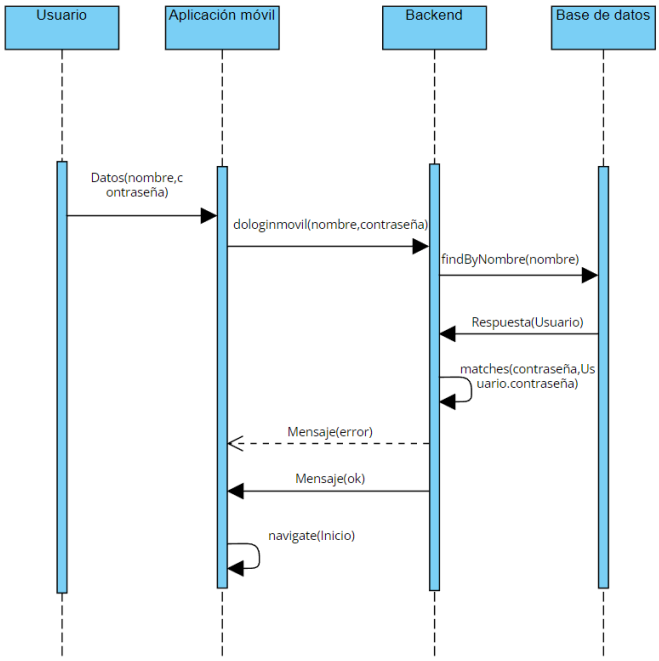


Figura 4.24: Diagrama de secuencia para el inicio de sesión en la aplicación móvil

Para iniciar sesión en la aplicación móvil el usuario debe introducir su nombre de usuario y su contraseña y pulsar el botón de iniciar sesión. Los datos se enviarán al *backend* de la aplicación web, donde se realizará una petición a la base de datos para recibir el usuario cuyo nombre coincida con el introducido. Una vez recibido el usuario, se comprobará si la contraseña encriptada coincide con la introducida usando un método que proporciona Spring Security. Una vez comprobado se enviará un mensaje a la aplicación móvil indicando si los datos del usuario coinciden o si ha habido algún error en el proceso.

4.3.17 Tomar datos desde la aplicación móvil

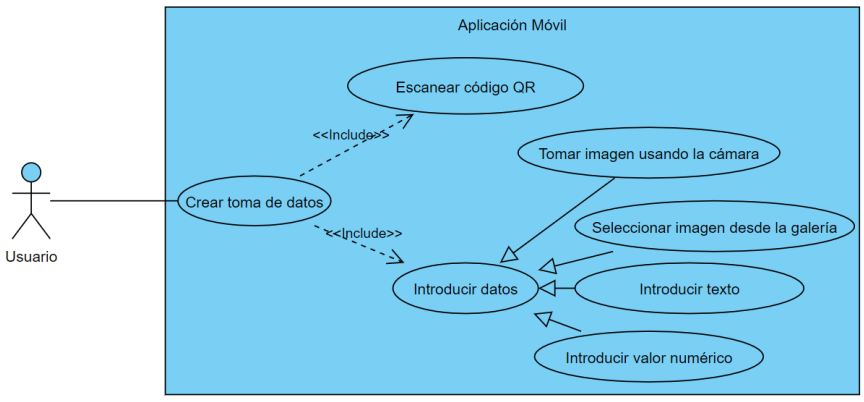


Figura 4.25: Diagrama de caso de uso para la toma de datos desde la aplicación móvil

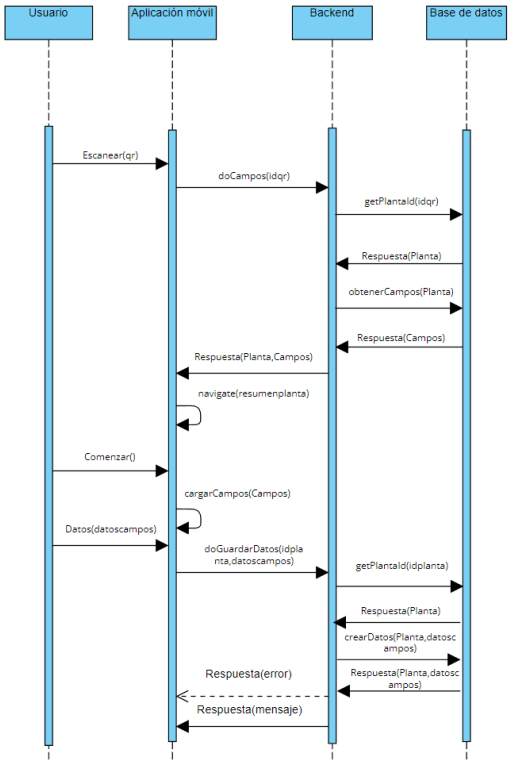


Figura 4.26: Diagrama de secuencia para la toma de datos desde la aplicación móvil

Para hacer una toma de datos desde la aplicación móvil el usuario debe pulsar el botón de escaneo de QR y escanear uno de los códigos QR generados desde la aplicación web. Una vez escaneado el código la aplicación envía una petición al *backend* indicando el QR de la planta de la que quiere obtener los detalles y los campos de la configuración que usa. El *backend* se encarga de recibir todos estos datos de la base de datos y los devuelve a la aplicación móvil.

Una vez recibidos los detalles de la planta, la aplicación móvil muestra un resumen de lo recibido y da la opción al usuario de tomar datos para esos campos. Una vez que el usuario ha pulsado el botón de toma de datos la aplicación muestra campos de entrada de datos para cada uno de los campos de la configuración. El usuario puede introducir los datos e imágenes que desee y, una vez haya terminado, pulsar el botón Enviar datos. Estos datos se envían al *backend* junto a la id de la planta, desde donde se envían a la base de datos, que devuelve un mensaje indicando si el guardado de datos se ha realizado de forma correcta o no.

4.4 Diseño del *backend* web

A continuación se detalla de forma general el diseño del código que se ha seguido para programar cada una de los casos de uso mediante capturas del algunas de las clases utilizadas junto a breves explicaciones de cada una.

4.4.1 Entidades

Para hacer uso de las tablas creadas en la base de datos de forma sencilla mediante código Java se han creado entidades correspondientes a cada una de las tablas. Estas entidades hacen uso de anotaciones JPA para indicar a qué tabla corresponden y a qué variable se

asignarán los valores de cada columna de la tabla. En los casos de relaciones entre varias tablas se ha indicado si es una relación "uno a uno", una relación "uno a muchos" o una relación "muchos a muchos", además de la dirección de la relación. Cabe decir que, a la hora de recibir los datos de las relaciones en forma de JSON, en ocasiones ocurría un bucle infinito en el que se devolvían los datos de una clase y esta devolvía los datos de la otra. Para solucionar esto hubo que indicar con otra anotación en qué dirección concreta debía devolverse la referencia del JSON.

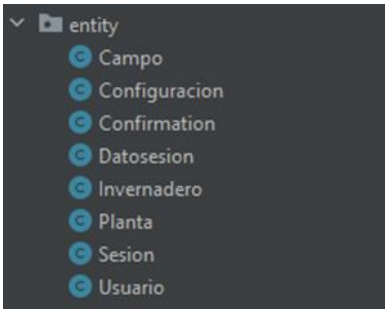


Figura 4.27: Entidades de la aplicación web

4.4.2 Repositorios

Para conectar las entidades con la base de datos es necesario crear un repositorio. Los repositorios son interfaces en las que se indica el tipo de entidad con la que se está trabajando y dónde se definen consultas personalizadas que queramos hacer con respecto a los datos guardados en la base de datos.

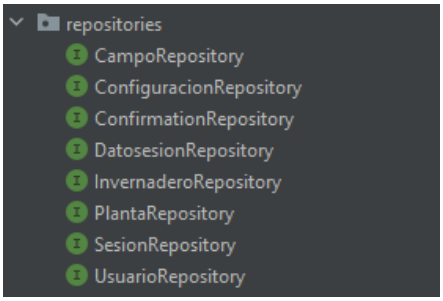


Figura 4.28: Repositorios de la aplicación web

Cuando se realiza una consulta a la base de datos usando el repositorio los datos devueltos serán de tipo opcional, indicando que es posible que no exista ningún dato que coincida con los parámetros establecidos o directamente del tipo de la entidad correspondiente a dicho repositorio. Los datos opcionales serán comprobados para asegurar que existen datos y serán transformados posteriormente a entidades de la clase correspondiente.

4.4.3 Servicios

Para hacer un uso más sencillo y modular de las operaciones que ofrecen los repositorios con respecto a la base de datos se decidió hacer uso del patrón Service. Para implementar este patrón se crearon clases correspondientes a cada uno de los repositorios. En estas clases se implementó cualquier método complejo en el que se debiera interactuar con las entidades para leer o guardar datos en la base de datos.

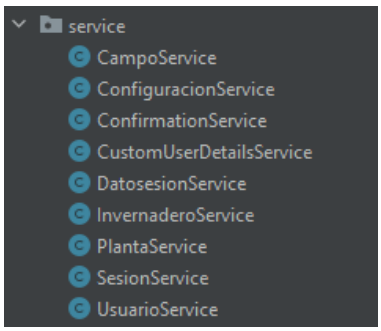
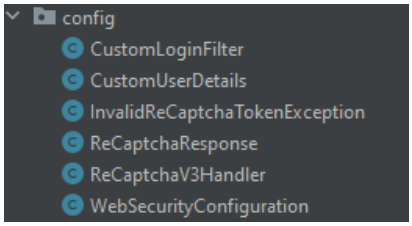


Figura 4.29: Servicios de la aplicación web

Si sólo es necesario realizar una consulta simple a la base de datos que devuelve un valor de entidad no se ha hecho uso de este patrón. Sin embargo, en el caso de acciones cómo, por ejemplo, guardar una nueva planta a partir de todos los datos recibidos desde el *frontend* se optó por crear la clase PlantaService junto con un método llamado guardarPlanta que se encarga de crear la entidad, asignarle los valores recibidos y, finalmente, guardarlo en la base de datos a través de la interfaz.

4.4.4 Seguridad



En la carpeta de configuración se encuentran todas las clases que manejan aspectos relacionados con la seguridad de la aplicación.

Figura 4.30: Archivos relacionados con la seguridad

En la clase WebSecurityConfiguration se manejan todos los detalles de la configuración de SpringSecurity. Se configuró indicando que las credenciales del usuario se manejarían usando la clase CustomUserDetails, que recibiría sus datos usando el repositorio general de la entidad Usuario en su propio Servicio, donde se convertirían los datos a esta clase para ser interpretados de forma correcta por SpringSecurity.

También se indicó el tipo de codificación que se usaría para las contraseñas de los usuarios registrados

Además, se configuraron los permisos de acceso de cada una de las direcciones de la aplicación, indicando el tipo de rol necesario junto a las direcciones usadas para el inicio de sesión y las direcciones a las que podría acceder la aplicación móvil.

Por último, se conformó un filtro que se aplica tras iniciar sesión y que hace uso de las clases relacionadas con ReCaptchaV3 de Google. Este filtro hace que se envíe una petición al servidor de Google con los detalles del inicio de sesión, que devuelve una puntuación con respecto a estos datos. Si la puntuación es menor a 0.5 el usuario es considerado como un robot y no se le permite el acceso a la aplicación. ReCaptchaV3 es una nueva versión del clásico "Captcha" en la que no es necesario introducir un texto o seleccionar imágenes sino que la puntuación es asignada de forma automática según los movimientos realizados por el usuario en la página.

4.4.5 Controladores web

Los controladores son las clases en las que se maneja la mayor parte de la lógica del *backend* de la aplicación. En ellas se manejan las acciones realizadas en cada una de las URIs de la aplicación. Están separadas en dos tipos: @Controller, que se encarga de controlar los cambios entre páginas de la aplicación y las acciones a realizar durante el cambio y @RestController, que se encarga de manejar las direcciones de la API REST de la aplicación, las acciones a realizar en cada una y los elementos a devolver al *frontend*.

En la clase RegistroController se manejan los datos introducidos por un nuevo usuario a la hora de registrarse, introduciéndose en la base de datos, encriptando su contraseña y creando una nueva confirmación de correo pendiente.

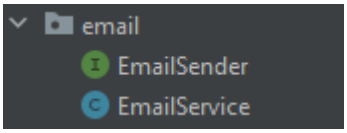
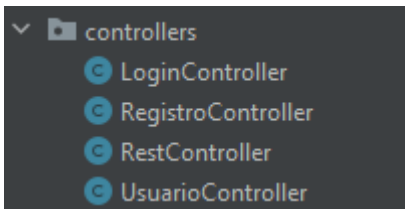


Figura 4.31: Archivos relacionados con el envío del correo de confirmación de registro

Para el envío del correo se ha usado una plantilla que es enviada por otra clase, EmailService, que implementa una

interfaz EmailSender. Es posible que sea necesario eliminar el comando de activación SSL en dicha clase debido a errores de conexión con la API del correo usada.



La clase LoginController simplemente redirige al usuario a la página de inicio de sesión y establece la dirección de la página principal una vez iniciada la sesión.

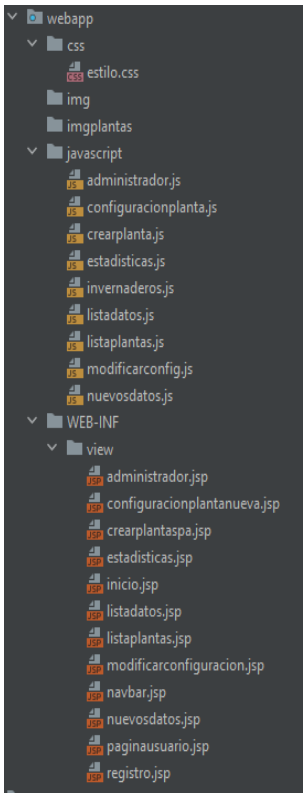
Figura 4.32: Controladores de la aplicación web

En la clase UsuarioController se establecen todas las direcciones de páginas de la aplicación web, desde crear una planta hasta ver las estadísticas de los datos introducidos.

Finalmente, en la clase RestController se establece la API REST de la aplicación. Aquí se reciben todas las peticiones realizadas desde el *frontend* y es donde se realiza la mayor parte de la lógica de la aplicación. Se hace uso de los Servicios e Interfaces establecidos previamente para recibir y guardar los datos que sean necesarios en la base de datos y se realiza un manejo de posibles errores durante el proceso. Una vez realizada la lógica de la petición recibida los datos tratados se devuelven al *frontend* en forma de JSON.

4.4.6 Configuración Spring

Todo proyecto Spring Boot contiene un archivo llamado application.properties. En este archivo se establecen todas las propiedades del proyecto, desde su nombre y tipo de archivos que serán leídos por el proyecto hasta variables de entorno, en las que se guardan todas las variables que usarán las clases de todo el proyecto, además de la configuración de la base de datos.



Cabe decir que, debido a un problema a la hora de crear y leer las variables de entorno relacionadas con la conexión a la API ReCaptchaV3 de Google, estas variables se han tenido que escribir a mano en la clase encargada de manejar dicha API para desplegar el proyecto en un nuevo servidor es necesario hacer varios cambios en la configuración de Spring cambiando algunas variables. En primer lugar, hay que determinar las carpetas en las que se guardaran las imágenes de los códigos QR y las imágenes subidas por los usuarios. Además, habrá que configurar el usuario y la contraseña de la cuenta de correo que enviará los correos de confirmación. Por último, también será posible cambiar la cuenta que administrará la consola de permisos de ReCaptchaV3 cambiando la clave API y su secreto correspondiente.

Figura 4.33: Archivos usados para la creación del *frontend* de la aplicación web

4.5 Diseño del *frontend* web

El *frontend* consta de diferentes páginas JSP correspondientes a direcciones establecidas en los controladores del *backend*, desde

los que se les envían parámetros a través del modelo para mostrar diferentes mensajes o elementos usando el código Java que permite usar JSP.

Además, para controlar todos los elementos dinámicos y el envío y el recibimiento de datos, se importan clases JavaScript en cada uno de los JSP. Estas clases JavaScript se encargar de realizar las peticiones necesarias al *backend* y de crear o eliminar elementos HTML dependiendo de los datos que hayan recibido.

En la carpeta css se encuentra la clase estilo que, junto a Bootstrap, se importa en cada uno de los JSP para aportar un estilo CSS personalizado a la aplicación web. Este estilo se ha configurado para ser "*responsive*", es decir, adaptarse a la web desde cualquier dispositivo móvil además del ordenador.

4.6 Diseño de la aplicación móvil

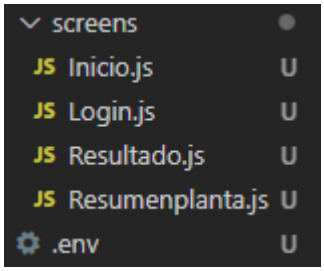


Figura 4.34: Archivos usados para la creación de la aplicación móvil

La aplicación móvil consta de cuatro pantallas principales creadas con componentes de React Native y Expo junto a código JavaScript que interactúa con las páginas y realiza llamadas al *backend* de la aplicación web para realizar consultas a la base de datos.

La pantalla Login se encarga de llevar a cabo todo el proceso de inicio de sesión del usuario, realizando una llamada al *backend* y comprobando si los datos introducidos coinciden con los guardados en la base de datos.

La pantalla Inicio aparece una vez se ha iniciado sesión y permite al usuario pulsar un botón para escanear un código QR creado para alguna planta. Una vez escaneado, se cargan de la base de datos los detalles de la planta y de su configuración y se pasa a la pantalla Resumenplanta. Esta pantalla se encarga de mostrar al usuario los detalles de la planta escaneada y de mostrar un botón de toma de datos que, al pulsarse, muestra campos de entrada de datos acordes a la configuración de la planta para que el usuario pueda introducir los que desee. Una vez que termina de introducirlos, el usuario puede pulsar el botón de guardar, haciendo que los datos se envíen a la base de datos y se muestre la pantalla de Resultado.

El archivo .env contiene la variable de entorno de la dirección de la aplicación web, lo que permite cambiarla dependiendo de donde se haya desplegado esta para que el valor se aplique en todas las pantallas anteriormente mencionadas.

4.7 Diseño de la base de datos

La base de datos ha sido creada usando MariaDB y se ha usado MySQL Workbench para crear las tablas y las relaciones entre ellas de forma más sencilla.

En primer lugar, consta de una tabla Usuario en la que se guardan todos los usuarios registrados en el sistema. En esta tabla se guardarán el email, nombre, contraseña cifrada, nivel de permiso y un valor booleano indicando si el usuario está activado.

Cuando un usuario se registra se crea una nueva fila en la tabla Confirmation. Esta tabla representa el *token* de confirmación que se genera cuando se envía el correo de confirmación del usuario. Cada confirmación constará de un *token* generado de forma

aleatoria que será enviado al usuario, su fecha y hora de creación y de expiración. También tiene otro atributo que indica si ya ha sido confirmado o no y una relación con el usuario que lo ha creado.

En segundo lugar, contiene una tabla Planta, en la que se guardan cada una de las plantas creadas en la base de datos. Cada planta tiene un nombre, una fecha de creación, la dirección del archivo correspondiente a su código QR, el nombre de la configuración que usa y otro valor booleano indicando si está activada. Además, tiene otro atributo creado que relaciona plantas con el usuario que las haya creado mediante su id. También tiene un atributo invernadero que relaciona plantas de la misma forma con el invernadero al que pertenecen. El Invernadero constará de un nombre y de otro valor booleano indicando si está activado.

La tabla Configuración representa el conjunto de campos que forman cada una de las configuraciones creadas. Consta de un nombre y de una relación con el usuario que la ha creado junto a un Campo. Para registrar una configuración se crean varias filas en la tabla con el mismo nombre y se relacionan con cada uno de los campos seleccionados. Cada campo está formado por un nombre y un tipo que puede ser textual, numérico o una imagen.

Para asignar una configuración a una planta se ha optado por añadir un atributo en la planta con el nombre de la configuración y obtenerla mediante consultas para así simplificar las relaciones.

Por último lugar, para crear una toma de datos, se plantea la tabla Sesion. Esta tabla indica la fecha junto al usuario y la planta que forman parte de dicha toma de datos. Para guardar los datos tomados existe otra tabla Datosesion que recoge un dato tomado y lo relaciona con el campo correspondiente de la configuración que está usando y con la sesión que se ha creado.

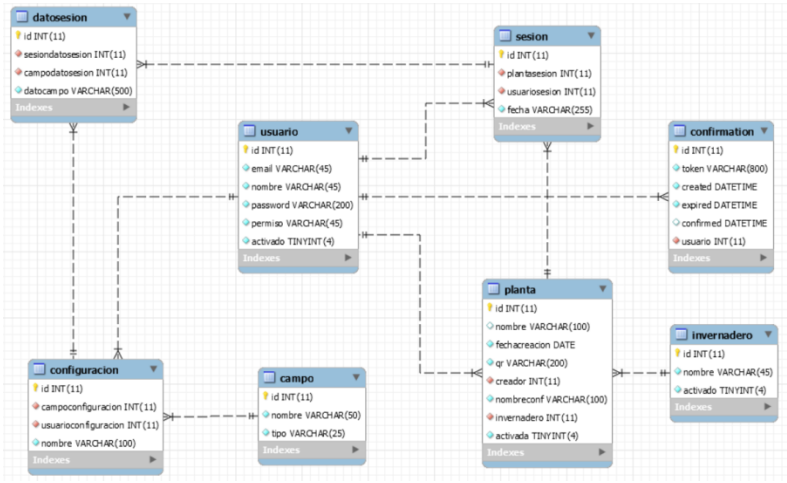


Figura 4.35: Diagrama de la base de datos

5

Conclusiones y líneas futuras

5.1 Conclusiones

Como conclusión, se han cumplido todos los objetivos principales y secundarios del proyecto, ofreciéndose una solución simple y eficaz al problema que se había planteado desde un principio, siendo mínimos los cambios que ha habido que realizar con respecto a la idea inicial de la propuesta.

Se ofrece la capacidad de manejar las plantas de diferentes invernaderos desde cualquier lugar usando una aplicación web sencilla y funcional junto a la capacidad de tomar datos de forma rápida usando la aplicación móvil. El proyecto desarrollado tiene la suficiente calidad y funcionalidades como para usarlo en entornos reales de invernaderos de investigación.

Personalmente considero que el desarrollo de este trabajo fin de grado ha sido una interesante experiencia que, como una aproximación a la Bioinformática, ha sido una gran oportunidad de poner en práctica todos los conocimientos obtenidos estos últimos años en la Escuela, además de haberme permitido aumentar mi conocimiento y agilidad a la hora de desarrollar un proyecto software. Trabajar recibiendo consejo y ayuda de un tutor ha sido de gran ayuda a la hora de aprender a desarrollar un proyecto por mi propia cuenta. Considero que la ágil supervisión durante todas sus fases de desarrollo es comparable a lo que ocurriría en un entorno de desarrollo externo a la universidad, por lo que es una buena forma de acercarse a lo que serían desarrollos externos. En general, considero todo el trabajo una experiencia positiva y una gran oportunidad para aprender por mi cuenta junto a la gran ayuda de mi tutor.

5.2 Líneas futuras

En general, las aplicaciones desarrolladas contienen todas las funcionalidades deseables y resulta difícil vaticinar qué otras funcionalidades adicionales puedan ser necesarias sin que todavía se estén usando a pleno rendimiento en un entorno de producción.

No obstante, con respecto a líneas futuras sería posible añadir nuevas funcionalidades a la aplicación móvil o a la aplicación web.

En primer lugar, se podría estudiar la necesidad de añadir un inicio de sesión basado en OauthV2 además del existente en la aplicación web. También se podrían añadir más funcionalidades a la herramienta de estadísticas de los datos tomados, recibiendo así más información aparte de las gráficas. Además, se podría recuperar la funcionalidad de imprimir estas estadísticas en formato PDF si se viera necesario.

La aplicación móvil también se podría ampliar de forma que se le añadiesen funcionalidades pertenecientes a la aplicación web como crear una planta o modificar la configuración de alguna planta escaneada.

Por último, en caso de invernaderos con cientos de plantas, es de prever la necesidad de incorporar algún mecanismo para la creación automática de plantas especificando una modelo ya creada y un sufijo que pueda ir, por ejemplo, de 1 a 100.

Referencias

[1] IHSM. *Invernadero de Teatinos*. Consultado el 10 de Julio de 2022 https://www.ihsm.uma-csic.es/servicios/invernaderoteatinos

[2] Pahino, R. (31 de marzo de 2020). *¿Qué son spring framework y spring boot? Tu Primer Programa Java Con Este framework*. campusMVP.es. Consultado el 10 de Julio de 2022 en https://www.campusmvp.es/recursos/post/que-son-spring-framework-y-spring-boot-tu-primer-programa-java-con-este-framework.aspx

[3] Spring. *Overview of Spring Framework*. Introduction to spring framework. Consultado el 10 de Julio de 2022 en https://docs.spring.io/spring-framework/docs/3.2.x/spring-framework-reference/html/overview.html

[4] Caules, C. Á. (11 de enero de 2022). *¿Qué es spring boot?* Arquitectura Java. Consultado el 10 de Julio de 2022 en https://www.arquitecturajava.com/que-es-spring-boot/

[5] Hibernate. Your relational data. objectively. Consultado el 18 de Agosto de 2022 en https://hibernate.org/orm/

[6] Caules, C. Á. (19 de Junio de 2021). ¿JPA vs Hibernate? Arquitectura Java. Consultado el 24 de Agosto de 2022 https://www.arquitecturajava.com/jpa-vs-hibernate/

[7] Spring. Spring security. Consultado el 18 de Agosto de 2022 en https://spring.io/projects/spring-security

[8] MariaDB. *MariaDB foundation*. Consultado el 14 de Julio de 2022 en https://mariadb.org/

[9] Zxing. *Zxing ("Zebra crossing") Barcode Scanning Library for Java, Android*. GitHub. Consultado el 17 de Agosto de 2022 en https://github.com/zxing/zxing

[10] React Native. Consultado el 11 de Julio de 2022 de https://reactnative.dev/

[11] Blanes, J. A. (13 de marzo de 2019). *¿Qué es react native?* Deloitte Spain. Consultado el 11 de Julio de 2022 https://www2.deloitte.com/es/es/pages/technology/articles/que-es-react-native.html

[12] Expo. Consultado el 11 de Julio de 2022 de https://expo.dev/

[13] Ancheta Feb 20, W. (20 de febrero de 2018). *Desarrollo más fácil react native Con Expo*. Code Envato Tuts+. Consultado el 11 de Julio de 2022 de https://code.tutsplus.com/es/tutorials/easier-react-native-development-with-expo--cms-30546

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S. DE INGENIERÍA INFORMÁTICA

E.T.S de Ingeniería Informática

Bulevar Louis Pasteur, 35

Campus de Teatinos