



UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería  
Informática**

**Aplicación Interacción  
Medicamentos  
Documentación Técnica**



Presentado por Iñigo Sanz Delgado  
en Universidad de Burgos — 11 de junio  
de 2024

Tutor: José Manuel Aroca Fernández



---

# Índice general

---

<b>Índice general</b>	<b>i</b>
<b>Índice de figuras</b>	<b>iii</b>
<b>Índice de tablas</b>	<b>v</b>
<b>Apéndice A Plan de Proyecto Software</b>	<b>1</b>
A.1. Introducción . . . . .	1
A.2. Planificación temporal . . . . .	1
A.3. Estudio de viabilidad . . . . .	5
<b>Apéndice B Especificación de Requisitos</b>	<b>11</b>
B.1. Introducción . . . . .	11
B.2. Objetivos generales . . . . .	11
B.3. Catálogo de requisitos . . . . .	12
B.4. Especificación de requisitos . . . . .	13
<b>Apéndice C Especificación de diseño</b>	<b>21</b>
C.1. Introducción . . . . .	21
C.2. Diseño de datos . . . . .	21
C.3. Diseño procedimental . . . . .	24
C.4. Diseño arquitectónico . . . . .	28
C.5. Guía de Estilos . . . . .	30
<b>Apéndice D Documentación técnica de programación</b>	<b>35</b>
D.1. Introducción . . . . .	35
D.2. Estructura de directorios . . . . .	35

D.3. Manual del programador . . . . .	37
D.4. Compilación, instalación y ejecución del proyecto . . . . .	42
D.5. Ejecución de la Aplicación . . . . .	46
D.6. Pruebas del sistema . . . . .	47
<b>Apéndice E Documentación de usuario</b>	<b>49</b>
E.1. Introducción . . . . .	49
E.2. Requisitos de usuarios . . . . .	49
E.3. Manual del usuario . . . . .	49
<b>Bibliografía</b>	<b>51</b>

---

# Índice de figuras

---

A.1. Línea de tiempo donde se muestran las fases, se puede ver como es la relevancia de las fases, haciendo hincapié en la primera, donde la duración de tiempo es grande pero donde el avance no es significativo, siendo el final de esta el pico del desarrollo del proyecto. . . . .	5
C.1. Diagrama entidad-relación de Usuarios y Roles . . . . .	22
C.2. Imagen de la relación de Usuarios y Roles . . . . .	23
C.3. Diagrama del DTO de Usuarios . . . . .	23
C.4. Diagrama del DTO de los Datos Excel . . . . .	24
C.5. Diagrama MVC, hecho con Canva . . . . .	29
C.6. Diagrama Interacción API, hecho con Canva . . . . .	29
C.7. Reverse Engineering MySQL Workbench . . . . .	30
C.8. Paleta de Colores de la plantilla Bootstrap . . . . .	31
C.9. Logo Activus . . . . .	32
C.10. Ejemplo de los botones . . . . .	32
C.11. Vista de los contenidos dentro de un estilo que implementa Card . . . . .	33
D.1. Captura donde se muestra el token de Mailtrap que hay que poner en la configuración del fichero application.properties . . . .	39
D.2. Captura donde hay que añadir el dominio requerido en Mailtrap . . . . .	39
D.3. Captura del panel del hosting del dominio, para ir a la siguiente imagen hay que pulsar en administrar DNS . . . . .	39
D.4. En esta captura se muestra la configuración de los registros de DNS de nuestro dominio, para ir a la siguiente página hay que pulsar en añadir un registro nuevo . . . . .	40
D.5. En esta captura se indican los valores del nuevo registro, los cuales los tenemos que copiar de Mailtrap, siguiente imagen . . . .	40

D.6. Captura Mailtrap donde se muestran todos los registros que tenemos que añadir en nuestro dominio . . . . .	41
D.7. Captura donde se muestra como van cambiando los estados de los registros según se vayan añadiendo al dominio . . . . .	41
D.8. Captrua donde debemos crear el correo de donde saldrán las notificaciones, la contraseña habrá que añadirla también en la configuración de la aplicación . . . . .	42
D.9. Conexión con WAMP Server . . . . .	43
D.10. Esquema utilizado para el proyecto . . . . .	44
D.11. Ejecución Spring donde se indica el JDK de Java utilizado . . .	45
D.12. Archivo requirements.txt para la instalación de dependencias . .	46
D.13. Ejemplo del uso de Postman para consultar la compatibilidad de dos medicamentos . . . . .	47

---

## Índice de tablas

---

A.1. Costes de hardware y amortización durante 6 meses . . . . .	6
A.2. Costes de software y amortización durante 6 meses . . . . .	7
A.3. Desglose del sueldo bruto anual de 22.000 € y deducciones . . . .	8
A.4. Costes de personal y amortización durante 6 meses . . . . .	9
A.5. Dependencias y sus respectivas licencias . . . . .	9
B.1. CU-1 Registrar Usuario. . . . .	14
B.2. CU-2 Iniciar Sesión. . . . .	15
B.3. CU-3 Cambiar Contraseña. . . . .	16
B.4. CU-4 Buscar Medicamentos. . . . .	17
B.5. CU-5 Comparar Medicamentos. . . . .	18
B.6. CU-6 Cambiar idioma de la aplicación. . . . .	19





## *Apéndice A*

---

# **Plan de Proyecto Software**

---

### **A.1. Introducción**

En este anexo se explica como ha sido la temporalización del desarrollo del proyecto, el análisis de la viabilidad legal y el coste económico.

### **A.2. Planificación temporal**

La planificación temporal se ha estructurado en diferentes fases, intentando abarcar en cada una de ellas las funciones necesarias para avanzar de la mejor manera. El desarrollo no ha sido lineal, sino que el desarrollo del proyecto ha sido casi nulo en las fases iniciales debido a factores como una planificación inadecuada, subestimación de la carga de trabajo y dificultad del mismo. El resultado lleva a que la carga de trabajo se concentre en las etapas finales. Esta situación se conoce como Crunch Time.

El proyecto se define en las siguientes fases:

1. **Fase 1**
2. **Fase 2**
3. **Fase 3**
4. **Fase 4**
5. **Fase 5**
6. **Implementaciones finales**

## FASES

La duración de las fases se va a segmentar en fechas.

### Fase 1

Duración: 14 de Noviembre de 2023 – 1 de Abril de 2024

En esta fase se inicia el proyecto recopilando información para realizar una estructuración de los pasos a seguir durante el futuro desarrollo. Se hace una gran inversión de tiempo, en vano, en el inicio, ya que se empieza sin seguir un planning competente.

Más adelante se empieza el trabajo efectivo, que se resume en desarrollar la aplicación en su parte Java de manera correcta pero sin funcionalidades complejas, junto con su comunicación con la base de datos.

- El inicio ocupa los meses noviembre, diciembre, enero y febrero, pero en diciembre y enero se para.
- El desarrollo de la aplicación se inicia en febrero, tanto con la configuración de la BBDD como con la codificación de la aplicación. Durante estos meses la inversión del tiempo se centra principalmente en entender las requisitos de las tecnologías que se utilizan, la relación de cada sección de la aplicación y en la corrección de errores.
- Durante los meses de marzo y abril se completan todos los requisitos para que la aplicación se pueda levantar sin errores o dejando los errores relevantes para una futura corrección.

Se exponen los tiempos de cada apartado, ya que en este punto la desorganización fue muy exagerada y no se lograron grandes avances.

### Fase 2

Duración: 1 de Abril de 2024 – 8 de Abril de 2024

Se empieza con el desarrollo de la API. Esta fase ocupa un espacio bastante reducido en la línea temporal, ya que abarca únicamente la preparación del entorno donde hay que desplegar la API y la creación de funcionalidades básicas de esta.

**Fase 3**

Duración: 8 de Abril de 2024 – 10 de Mayo de 2024

Esta fase recopila el tiempo dedicado a la interfaz visual. Las primeras semanas se van creando las vistas preliminares de la aplicación, pero también se dedica una cantidad de tiempo considerable a entender el funcionamiento de Thymeleaf.

Por otro lado esta el correcto funcionamiento de los controladores, a los que se le dedica menos tiempo. Sin embargo, también se debía entender su funcionamiento.

- Implementar la plantilla Bootstrap que se va a utilizar.
- Crear las vistas iniciales que va a contener la aplicación.
- Crear y configurar los controladores para el uso de las vistas.
- Tocar la configuración de seguridad de Spring Boot para dar los permisos necesarios a cada parte.

No se expone el tiempo dedicado a cada apartado, ya que cada necesidad se iba implementando al mismo tiempo y conforme se iban completando los requisitos necesarios.

**Fase 4**

Duración: 1 de Mayo de 2024 – 27 de Mayo de 2024

En la cuarta fase se vuelve con el API y se empieza a programar cada uno de los servicios que debe ofrecer:

- Cargar en la base de datos la tabla de datos de medicamentos.
- Cargar en la BBDD la tabla de compatibilidades. El mayor inconveniente fue que se disponía de una tabla de compatibilidades en imagen, por lo que se tuvo que dedicar tiempo en implementar la forma de cargar los datos en forma de matriz para crear su correspondiente tabla en la base de datos.
- Cear la funcionalidad para poder comparar medicamentos según los principios activos que contengan. El comparador costo varios días que funcionase, y en su primera fase únicamente se recogían los nombres

de los medicamentos que se le pasaban. Luego se consigue que según el nombre se consiga el principio activo y más adelante utilizando ambas tablas que realizase las comparaciones.

### **Fase 5**

Duración: 13 de Mayo de 2024 – 3 de Junio de 2024

La última fase reúne lo realizado durante las fases anteriores, corrigiendo errores pendientes y añadiendo las funciones pendientes.

- Arreglar el registro de usuarios de la aplicación.
- Conectar la API para implementar las funcionalidades faltantes.
- Preparar y definir los estilos para las últimas vistas de la interfaz.

Las pinceladas finales de esta fase no fueron una complicación excesiva, pero si se tuvieron que investigar errores en la conexión entre ambas partes de la aplicación y su misma corrección.

### **Implementaciones finales**

Duración: Últimas semanas, comprendidas entre el 29 de Mayo y el 9 de Junio.

Se añade el panel de gestión de usuarios que va a ser utilizado por el administrador del sistema. Se implementa la funcionalidad para enviar correos electrónicos a los nuevos usuarios y para que estos mismos puedan hacer un reset de password.

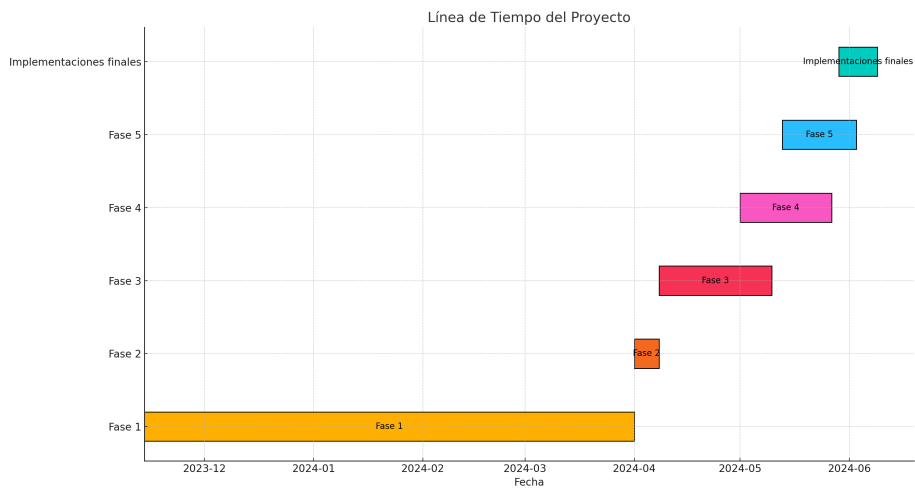


Figura A.1: Línea de tiempo donde se muestran las fases, se puede ver como es la relevancia de las fases, haciendo hincapié en la primera, donde la duración de tiempo es grande pero donde el avance no es significativo, siendo el final de esta el pico del desarrollo del proyecto.

### A.3. Estudio de viabilidad

En este apartado se realiza un estudio sobre la viabilidad económica del proyecto y sobre la viabilidad, teniendo en cuenta diferentes aspectos:

#### Viabilidad económica

La viabilidad económica del proyecto puede ser factible. En el siguiente punto se desglosan los motivos para que la aplicación se pueda monetizar y los costes económicos de ella.

#### Costes

Los costes se pueden dividir en dos, los costes de hardware y los de software. Pero también se incluyen los costes de usuarios y personal.

#### Costes Hardware

Estos costes pueden variar, principalmente por la calidad y especificaciones de los componentes que se utilicen. En este caso se ha utilizado un ordenador de sobremesa con sus respectivos periféricos (dos monitores). Para

no entrar en detalles sobre las especificaciones de los elementos hardware se indica un precio medio aproximado.

Según páginas oficiales de los fabricantes, la vida media útil de un equipo de sobremesa es de 6 a 8 años.

Los periféricos:

- Teclado.
- Ratón.
- Monitor (x2).

Elemento	Coste Medio (€)	Amortización 6 meses (€)
Ordenador de sobremesa	800	57.14
Teclado	50	3.57
Ratón	30	2.14
Monitor (x2)	300	21.43
<b>Total</b>	<b>1180</b>	<b>84.29</b>

Tabla A.1: Costes de hardware y amortización durante 6 meses

La amortización se puede calcular dividiendo el coste total por la vida media útil del equipo (vamos a suponer 7 años, un promedio entre 6 y 8 años) y luego multiplicando por la fracción del año que representa el proyecto (6 meses es 0.5 años).

### Costes Software

En los costes Software hay más variedad, debido a las tecnologías utilizadas y el enfoque de la aplicación. Se indicarán los precios medios del mercado:

Software:

- Visual Studio Code
- IntelliJ

- MySQL Workbench
- WAMP
- Postman

Utilidades:

- Github Copilot
- ChatGPT

Servicios (opcionales):

- Dominio
- Hosting
- Servidor de correo

Elemento	Coste Medio (€)	Amortización 6 meses (€)
Visual Studio Code	0	0
IntelliJ (estudiante)	0	0
MySQL Workbench	0	0
WAMP	0	0
Postman	0	0
GitHub Copilot	10/mes	60
ChatGPT	20/mes	120
Dominio	12/año	6
Hosting	60/año	30
Servidor de correo	24/año	12
<b>Total</b>	<b>126</b>	<b>228</b>

Tabla A.2: Costes de software y amortización durante 6 meses

### Precio Usuarios

En este apartado se van a indicar los costes de usuarios para la aplicación, estos costes van a ser hipotéticos, debido a que de cara al futuro aun no están decididas muchas de las opciones o enfoques que se le van a dar a la aplicación.

Después de hacer un pequeño estudio de mercado, se decide poner los siguientes precios para la utilización de la APP:

1. **Plan básico:** 12 €/mes
2. **Plan completo:** 25 €/mes
3. **Plan colectivo:** Se estudiará el colectivo de personas de una organización para fijar el precio.

### Costes de Personal

Para este coste se va a tener en cuenta el impacto económico de tener a una persona desarrollando la aplicación. Se incluirá también el coste de mantenimiento y las constantes mejoras a implementar.

Puesto que actualmente se dispone del nivel Junior-Trainee, se ha escogido un sueldo bruto de 22.000 €, es el el sueldo medio de este puesto en España. Se desglosa de la siguiente manera:

Concepto	Coste Anual (€)
Salario base	16.000
Complemento por antigüedad	2.000
Pagas extras (2 pagas extra anuales)	2.000
Bonificaciones y otros complementos	2.000
<b>Total Sueldo Bruto</b>	<b>22.000</b>
Contribuciones a la Seguridad Social (6.35 %)	1.397
Retención IRPF (15 %)	3.300
<b>Total Deducciones</b>	<b>4.697</b>
<b>Total Sueldo Neto</b>	<b>17.303</b>

Tabla A.3: Desglose del sueldo bruto anual de 22.000 € y deducciones



Concepto	Coste Anual (€)	Coste 6 meses (€)
Sueldo Bruto	20.000	10.000
<b>Total</b>	<b>20.000</b>	<b>10.000</b>

Tabla A.4: Costes de personal y amortización durante 6 meses

## Viabilidad legal

En la sección de viabilidad legal se van a especificar las licencias que requiere la aplicación.

### Licencias Software

Estas licencias son contratos legales que definen los límites y las condiciones para el uso del software. Las licencias principales que se han encontrado en el uso de este proyecto son:

Dependencia	Licencia
Spring Boot	Apache 2.0
Hibernate (JPA)	LGPL
MySQL Connector/J	GPL
Thymeleaf	Apache 2.0
Bootstrap	MIT
jQuery	MIT
FontAwesome	SIL OFL 1.1
Python	PSF License
Flask	BSD

Tabla A.5: Dependencias y sus respectivas licencias

Las licencias que se mencionan en la tabla de arriba permiten que se puedan utilizar sus componentes en proyectos tanto comerciales como no comerciales bajo las condiciones de cada licencia.

## Términos y Condiciones de las APIs

### API de Gestión de Medicamentos

Esta API se ha desarrollado internamente como servicio para la aplicación. A pesar de ello, cualquier información utilizada o almacenada debe cumplir con

la normativa de protección de datos y privacidad. Incluyendo el Reglamento General de Protección de Datos (GDPR), el cual establece ciertas directrices estrictas para la recopilación, almacenamiento y procesamiento de los datos personales de los usuarios.

### Cumplimiento de Normativas y Regulaciones

Se ha recopilado información sobre que normal debe cumplir la aplicación y las regulaciones a las que tiene que estar sujeta, especialmente en el sector sanitario.

- **Ley de Protección de Datos Personales:** Asegura que todos los datos personales de los usuarios se manejan de acuerdo con las leyes locales e internacionales de protección de datos [1].
- **Normativas Sanitarias:** Dependiendo de la jurisdicción, puede ser necesario cumplir con regulaciones específicas sobre el manejo de información médica y sanitaria [2].

## *Apéndice B*

---

# Especificación de Requisitos

---

### B.1. Introducción

En esta sección se recogen los diferentes requisitos del proyecto. Se organizan en dos apartados:

- Catálogo de Requisitos: Una lista que detalla todos los requisitos funcionales y no funcionales del sistema.
- Casos de Uso: Descripciones de las interacciones entre los usuarios que utilizan la aplicación y la plataforma para lograr los objetivos.

### B.2. Objetivos generales

En este apartado se recogen los objetivos generales del proyecto. Estos objetivos se centran en ofrecer una solución o servicio eficiente para la gestión médica y el uso de medicamentos, de manera que los especialistas sanitarios dispondrán de facilidades para el desempeño de sus labores.

Por otro lado se implementará su correspondiente servicio para los administradores del sistema, en el que podrán gestionar usuarios, permisos y otro tipo de funcionalidades.

- Diseñar e implementar un servicio web para la gestión de usuarios y aplicaciones.
- Interconectar una arquitectura que soporte múltiples servicios y tecnologías.

- Facilitar la administración de medicamentos y pacientes a través de un sistema web.

## B.3. Catálogo de requisitos

### Requisitos Funcionales

- **RF1:** El sistema debe permitir a los usuarios registrarse y autenticarse.
  - **RF1.1:** Registro de usuarios: Los usuarios deben poder registrarse proporcionando un nombre de usuario, correo electrónico, contraseña y seleccionando un rol.
  - **RF1.2:** Inicio de sesión: Los usuarios deben poder iniciar sesión con el nombre de usuario y la contraseña.
- **RF2:** Gestión de perfiles de usuario.
  - **RF2.1:** El administrador debe poder editar la información de los usuarios y habilitarlos.
  - **RF2.2:** Los usuarios deben poder cambiar su contraseña.
- **RF3:** Gestión de medicamentos.
  - **RF3.1:** Buscar medicamentos por nombre y ver la información sobre estos.
- **RF4:** Comparación de medicamentos.
  - **RF4.1:** El sistema debe permitir comparar medicamentos basándose en los principios activos.
  - **RF4.2:** Visualizar los resultados de la comparación de manera comprensible.
- **RF5:** Interfaz de usuario.
  - **RF5.1:** Desarrollar una interfaz web sencilla y fácil de usar para los usuarios.
  - **RF5.2:** La interfaz debe ser responsive y adaptarse a diferentes tamaños de dispositivos.
- **RF6:** Notificaciones y correos electrónicos.

- **RF6.1:** Permitir a los usuarios el restablecimiento de contraseñas a través de correo electrónico.
- **RF7:** Soporte en múltiples idiomas.
  - **RF7.1:** Soportar múltiples idiomas en la interfaz de usuario.
  - **RF7.2:** Permitir a los usuarios cambiar el idioma de la aplicación.
- **RF7:** Seguridad y permisos.
  - **RF7.1:** Implementar seguridad en el acceso a las funcionalidades según el rol que contenga el usuario.

## Requisitos no Funcionales

- **RNF1:** Diseño Responsive: Debe contener la funcionalidad responsive, es decir, que se adapte a los diferentes tamaños de los dispositivos donde se ejecute sin perder información de la aplicación a la hora de visualizarla.
- **RNF2:** Rendimiento: La aplicación debe proporcionar el resultado de las consultas de los usuarios en un tiempo razonable.
- **RNF3:** Seguridad: El sistema debe proteger los datos del usuario mediante cifrados o encriptaciones.
- **RNF4:** Compatibilidad: La aplicación debe ser compatible con los navegadores más utilizados (Chrome, Firefox, ...).
- **RNF5:** Internacionalización: La web debe estar disponible en un mínimo de dos idiomas.
- **RNF6:** Mantenimiento: El código debe estar bien documentado.
- **RNF7:** Usabilidad: La interfaz de usuario debe ser intuitiva y fácil de usar.
- **RNF8:** Modularidad: Utilizar tecnologías que permitan una expansión modular del sistema para su futuro crecimiento.

## B.4. Especificación de requisitos

En esta sección se van a desglosar los casos de uso:

CU-1	Registrar Usuario
<b>Versión</b>	1.0
<b>Autor</b>	Iñigo Sanz Delgado
<b>Requisitos asociados</b>	RF-1.1
<b>Descripción</b>	Permite a los nuevos usuarios registrarse proporcionando sus datos.
<b>Precondición</b>	Ninguna
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El Usuario accede a la página de registro.</li> <li>2. El Usuario introduce su nombre de usuario, correo electrónico, contraseña y selecciona un rol.</li> <li>3. El Usuario envía el formulario de registro.</li> <li>4. El Sistema valida los datos introducidos.</li> <li>5. El Sistema crea una nueva cuenta de usuario.</li> <li>6. El Sistema envía un correo de confirmación al usuario.</li> </ol>
<b>Postcondición</b>	El usuario está registrado y recibe un correo de confirmación.
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ El correo electrónico ya está registrado.</li> <li>■ La contraseña no cumple con los requisitos de seguridad.</li> </ul>
<b>Importancia</b>	Alta

Tabla B.1: CU-1 Registrar Usuario.

CU-2	Iniciar Sesión
<b>Versión</b>	1.0
<b>Autor</b>	Iñigo Sanz Delgado
<b>Requisitos asociados</b>	RF-1.2
<b>Descripción</b>	Permite a los usuarios autenticarse en el sistema.
<b>Precondición</b>	El usuario debe estar registrado.
<b>Acciones</b>	<ol style="list-style-type: none"><li>1. El Usuario accede a la página de inicio de sesión.</li><li>2. El Usuario introduce su nombre de usuario y contraseña.</li><li>3. El Usuario envía el formulario de inicio de sesión.</li><li>4. El Sistema valida las credenciales.</li><li>5. El Sistema redirige al usuario a la página principal.</li></ol>
<b>Postcondición</b>	El usuario está autenticado y puede acceder a sus funciones.
<b>Excepciones</b>	<ul style="list-style-type: none"><li>■ Las credenciales son incorrectas.</li></ul>
<b>Importancia</b>	Alta

Tabla B.2: CU-2 Iniciar Sesión.

<b>CU-3</b>	<b>Cambiar Contraseña</b>
<b>Versión</b>	1.0
<b>Autor</b>	Iñigo Sanz Delgado
<b>Requisitos asociados</b>	RF-2.2
<b>Descripción</b>	Permite a los usuarios cambiar su contraseña.
<b>Precondición</b>	El usuario debe estar autenticado.
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El Usuario accede a la sección de perfil.</li> <li>2. El Usuario selecciona la opción para cambiar la contraseña.</li> <li>3. El usuario proporciona su nombre de usuario para cambiar la contraseña.</li> <li>4. Al usuario le llega un correo electrónico para realizar el cambio de contraseña.</li> <li>5. El Sistema valida la contraseña actual y la nueva contraseña.</li> <li>6. El Sistema actualiza la contraseña del usuario.</li> </ol>
<b>Postcondición</b>	La contraseña del usuario ha sido cambiada.
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ La contraseña actual es incorrecta.</li> <li>■ La nueva contraseña no cumple con los requisitos de seguridad.</li> </ul>
<b>Importancia</b>	Media

Tabla B.3: CU-3 Cambiar Contraseña.



CU-4	Buscar Medicamentos
<b>Versión</b>	1.0
<b>Autor</b>	Iñigo Sanz Delgado
<b>Requisitos asociados</b>	RF-3.1
<b>Descripción</b>	Permite a los usuarios buscar medicamentos por nombre y ver su información.
<b>Precondición</b>	El usuario debe estar autenticado.
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El Usuario accede a la sección de búsqueda de medicamentos.</li> <li>2. El Usuario introduce el nombre del medicamento en el campo de búsqueda.</li> <li>3. El Usuario envía la solicitud de búsqueda.</li> <li>4. El Sistema busca medicamentos que coincidan con el nombre proporcionado.</li> <li>5. El Sistema muestra los resultados de la búsqueda al usuario.</li> </ol>
<b>Postcondición</b>	El usuario puede ver la información de los medicamentos buscados.
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ No se encuentran medicamentos que coincidan con la búsqueda.</li> </ul>
<b>Importancia</b>	Alta

Tabla B.4: CU-4 Buscar Medicamentos.

CU-5	Comparar Medicamentos
<b>Versión</b>	1.0
<b>Autor</b>	Iñigo Sanz Delgado
<b>Requisitos asociados</b>	RF-4.1
<b>Descripción</b>	Permite a los usuarios comparar medicamentos basándose en sus principios activos.
<b>Precondición</b>	El usuario debe estar autenticado.
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El Usuario accede a la sección de comparación de medicamentos.</li> <li>2. El Usuario selecciona los medicamentos que desea comparar.</li> <li>3. El Usuario envía la solicitud de comparación.</li> <li>4. El Sistema compara los principios activos de los medicamentos seleccionados.</li> <li>5. El Sistema muestra los resultados de la comparación al usuario.</li> </ol>
<b>Postcondición</b>	El usuario puede ver los resultados de la comparación de medicamentos.
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ No se encuentran principios activos que coincidan en los medicamentos seleccionados.</li> </ul>
<b>Importancia</b>	Alta

Tabla B.5: CU-5 Comparar Medicamentos.

CU-6	Cambiar idioma de la aplicación
<b>Versión</b>	1.0
<b>Autor</b>	Iñigo Sanz Delgado
<b>Requisitos asociados</b>	RF7.2
<b>Descripción</b>	Permitir a los usuarios cambiar el idioma de la aplicación.
<b>Precondición</b>	Ninguna
<b>Acciones</b>	<ol style="list-style-type: none"><li>1. El usuario accede a la configuración de su perfil.</li><li>2. El usuario selecciona el idioma deseado de una lista de opciones.</li><li>3. El usuario guarda los cambios.</li><li>4. El sistema actualiza el idioma de la interfaz de usuario.</li></ol>
<b>Postcondición</b>	La interfaz de usuario se muestra en el idioma seleccionado.
<b>Excepciones</b>	Ninguna
<b>Importancia</b>	Baja

Tabla B.6: CU-6 Cambiar idioma de la aplicación.



## *Apéndice C*

---

# **Especificación de diseño**

---

### **C.1. Introducción**

En las especificaciones de diseño se van a documentar las decisiones que se han tomado para estructurar el proyecto.

### **C.2. Diseño de datos**

Para representar los datos utilizados en la aplicación se ha decidido utilizar un modelo relacional. La base de datos utilizada es MySQL, opción muy fiable y utilizada comúnmente en el desarrollo de aplicaciones web.

#### **Modelo Entidad-Relación**

Primero se muestra el diagrama de la relación entre Usuarios y Roles.

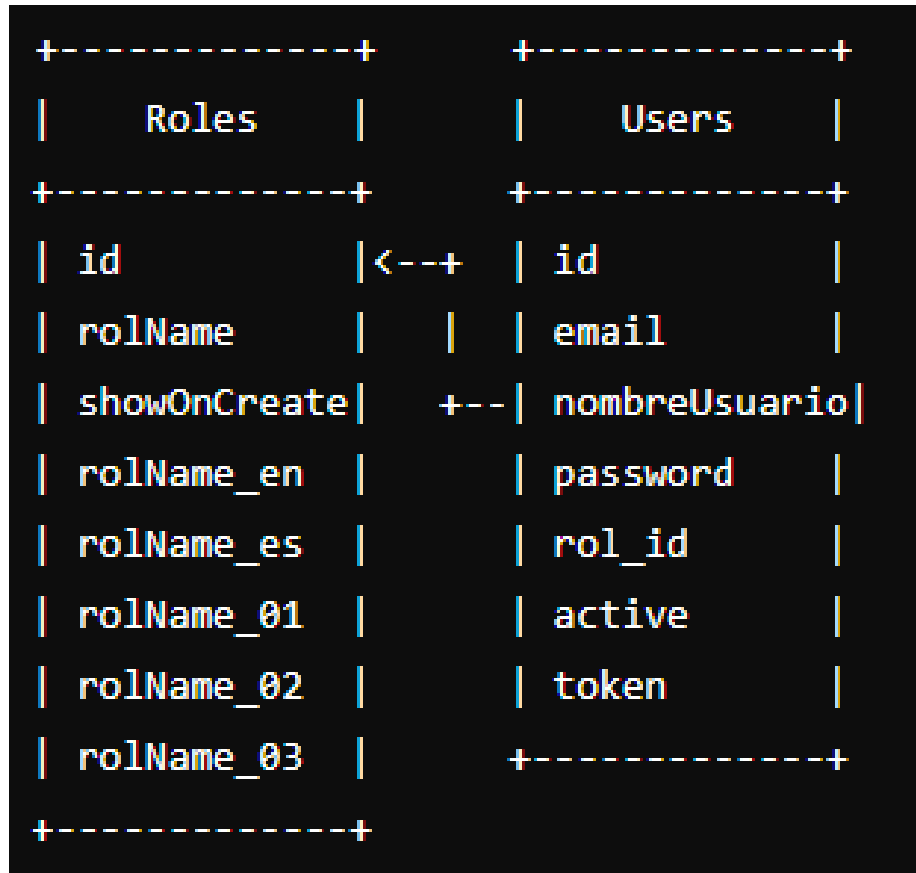


Figura C.1: Diagrama entidad-relación de Usuarios y Roles

- La tabla Users tiene una columna para la clave foránea (rol\_id), la cual referencia a la columna (id) de la tabla de Roles.
- La relación entre Users y Roles, es una relación de muchos a uno (ManyToOne).
- La relación entre Roles y Users, es una relación de uno a muchos (OneToMany).

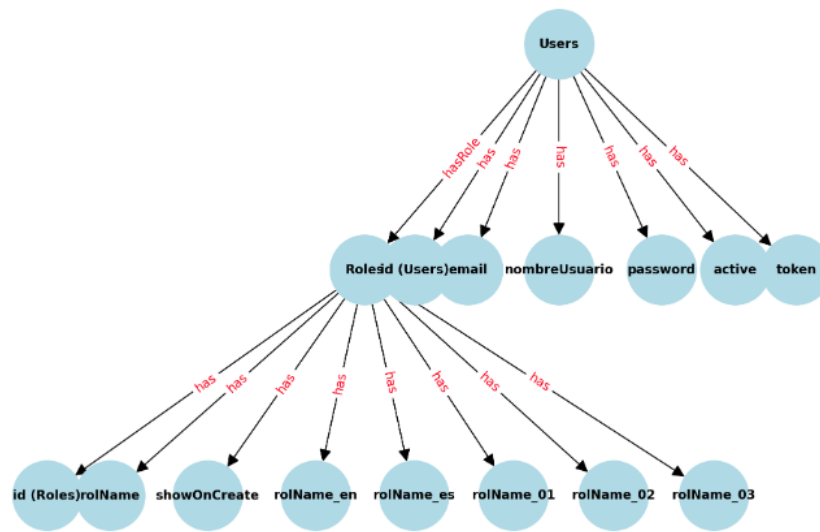


Figura C.2: Imagen de la relación de Usuarios y Roles

## Diagramas java

Los diagramas que se van a mostrar en las siguientes imágenes son los diagramas que el el programa IntelliJ proporciona de cada clase.

## DTOs

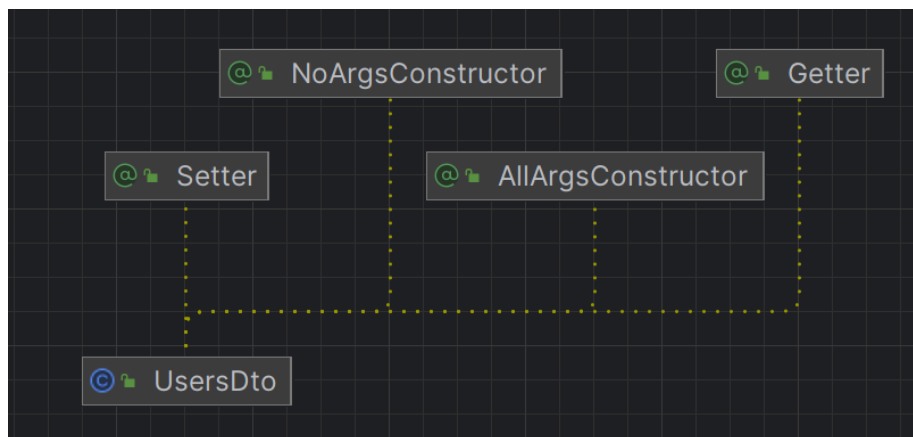


Figura C.3: Diagrama del DTO de Usuarios

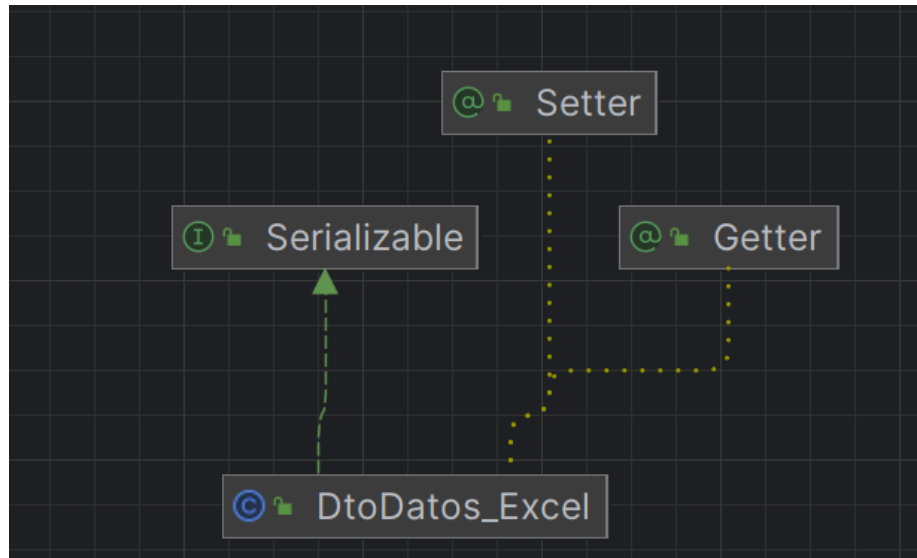


Figura C.4: Diagrama del DTO de los Datos Excel

Se pueden observar todos los diagramas de las clases de la siguiente manera:

1. En IntelliJ
2. Pulsamos en la clase que queremos visualizar
3. Click derecho encima de la clase
4. Opción Diagrams
5. Visualizamos el diagrama en cuestión

### C.3. Diseño procedimental

Este apartado explica el inicio de sesión y la lógica que hay detrás.

#### Inicio de Sesión

El inicio de sesión en la aplicación utiliza Spring Security para manejar la autenticación. El flujo del inicio de sesión es de la siguiente manera:



### **Formulario de Inicio de Sesión**

El usuario accede a una vista que contiene el inicio de sesión (`login.html`). El formulario envía los datos de inicio de sesión al servidor.

### **Controlador de Inicio de Sesión**

El controlador correspondiente maneja la solicitud para el inicio de sesión. Si las credenciales son correctas, se autentica al usuario y se le redirige a la página principal.

### **Gestión de Sesiones**

Spring Security gestiona las sesiones de usuario, utilizando cookies de sesión para mantener al usuario autenticado durante un tiempo en sus interacciones mientras utiliza la aplicación.

### **Registro de Usuarios**

El registro de nuevos usuarios sigue un flujo similar al del inicio de sesión, con algunas diferencias para crear estos usuarios.

### **Formulario de Registro**

El usuario accede a la vista con el formulario de registro (`register.html`). Este formulario envía los datos de registro al servidor utilizando un POST a la ruta `/users/registro`.

### **Controlador de Registro**

El controlador maneja la solicitud de registro. Si los datos son válidos y la contraseña cumple con la seguridad, se crea una nueva cuenta de usuario en la base de datos. Hay que especificar el Rol del usuario, que en este caso lo selecciona el mismo. Estas opciones se ocultarán para solo mostrar las necesarias.

### **Codificación de Contraseñas**

La contraseña del usuario se codifica utilizando la opción de `BCryptPasswordEncoder` antes de almacenarse en la base de datos para garantizar la seguridad en la aplicación.

## Recuperación de Contraseña

La recuperación de contraseña es una funcionalidad que puede salvar a cualquier usuario del olvido de su contraseña.

### Formulario de Solicitud de Restablecimiento de Contraseña

El usuario accede a la vista que contiene el formulario para solicitar el restablecimiento de la contraseña (`emailresetearpassword.html`). Este formulario envía los datos al servidor utilizando un método `POST` a la ruta `/users/hasOlvidadoTuPassword`.

### Controlador de Solicitud de Restablecimiento de Contraseña

El controlador genera un token para el restablecimiento y envía un correo electrónico al usuario con un enlace que le permite ir a la vista para rellenar el formulario para restablecer la contraseña.

## Gestión de Usuarios

La gestión de usuarios incluye funcionalidades para poder listar, habilitar/deshabilitar y eliminar los usuarios en el sistema.

### Listado de Usuarios

El listado de usuarios muestra una tabla en con los usuarios que hay exceptuando el usuario que esta loggeado en ese momento. El controlador correspondiente obtiene la lista de usuarios desde la base de datos y se la pasa a la vista.

### Habilitar/Deshabilitar Usuarios

La habilitación o des-habilitación de un usuario se realiza mediante un botón que hay en la tabla de la vista de usuarios. Al hacer clic en el botón, se envía una solicitud `POST` al servidor para el cambiar el estado a activo o desactivarlo.

### Eliminar Usuarios

Se realiza de la misma manera que el apartado anterior, se envía una solicitud `POST` para eliminar al usuario.

## Comparador de Medicamentos

El comparador de medicamentos permite a los usuarios ver la compatibilidad entre dos medicamentos basándose en sus principios activos.

### Formulario de Comparación

El usuario ingresa los nombres de dos medicamentos en un formulario (`comparador.html`). El formulario se comunica con la API para que realice el apartado lógico de la comparación, esta devuelve los resultados y se muestran en esa misma vista.

## Controlador de Comparación

El controlador correspondiente maneja la solicitud de comparación, de esta manera se comunica con la API o puede comunicarse con la API para enviar o recibir los datos.

La búsqueda de información sobre medicamentos se realiza de la misma manera.

## Gestión de Datos

La gestión de los datos que utiliza la aplicación se realiza mediante el uso de JPA (Java Persistence API) para así interactuar con la base de datos MySQL y poder mapear los datos a objetos DTO (Data Transfer Objects).

### Repositorio de Datos

El repositorio de datos utiliza JPA para comunicarse con la base de datos y obtener los datos que se necesitan.

### Servicios

Los servicios tienen la lógica de negocio y se ayudan de los repositorios para acceder a los datos que necesitan.

### Controladores

Los controladores manejan las solicitudes HTTP e interactúan con los servicios para obtener los datos que se necesitan y devolverlos a las vistas correspondientes.

## Proceso de Obtención de Datos

1. **Carga de los Datos:** Los datos se cargan desde la BBDD utilizando JPA y se mapean a objetos DTO utilizando los mappers.
2. **Procesamiento de los Datos:** Los datos se procesan en la capa de servicio que aplica la lógica de negocio que se necesita necesaria.
3. **Devolución de Datos:** Los datos, una vez procesados, se devuelven a los controladores, que los pasan a las vistas para su que se presenten.

Se adjuntará un documento JavaDoc que contendrá la información del código. [3]

## C.4. Diseño arquitectónico

En este apartado se concretan los elementos más relevantes sobre la arquitectura del proyecto.

### Gestión de Usuarios

Para la gestión de las operaciones que tienen relación con los usuarios, se ha implementado un motor que maneja las solicitudes CRUD de los usuarios de la aplicación. Se implementa utilizando Java y Spring Boot, siguiendo el patrón de diseño Modelo-Vista-Controlador.

### Patrón MVC

En este proyecto el patrón MVC se representa de la siguiente manera:

1. Modelo: El modelo es representado por las entidades JPA que se utilizan para mapear las tablas de la BBDD.
2. Vista: La vista se construye utilizando Thymeleaf para generar las interfaces en HTML.
3. Controlador: El controlador se implementa en Java utilizando Spring Boot para manejar las solicitudes HTTP que se realizan.

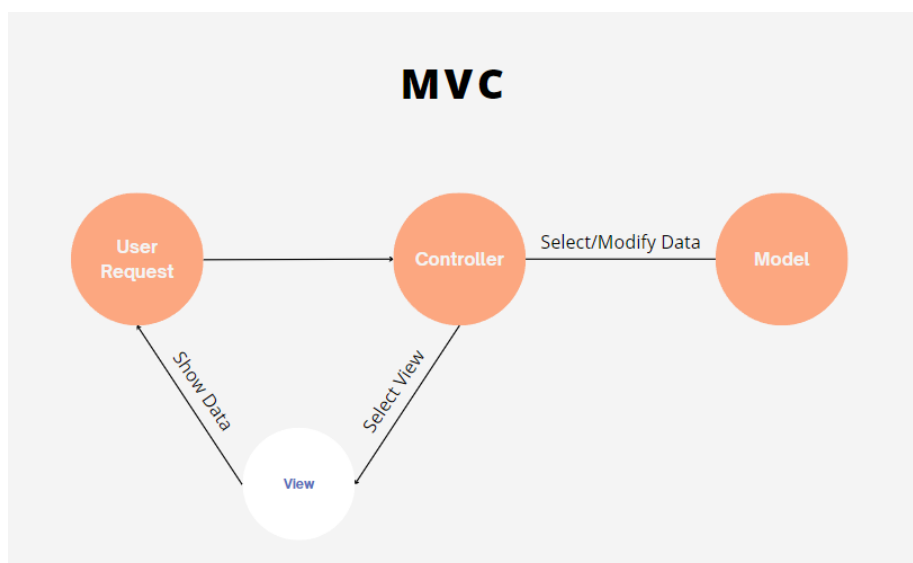


Figura C.5: Diagrama MVC, hecho con Canva

## Integración con la API en Python

El proyecto integra una API que realiza diferentes operaciones con los datos. La integración también se logra mediante solicitudes HTTP desde el backend, utilizando la librería RestTemplate que proporciona Spring.

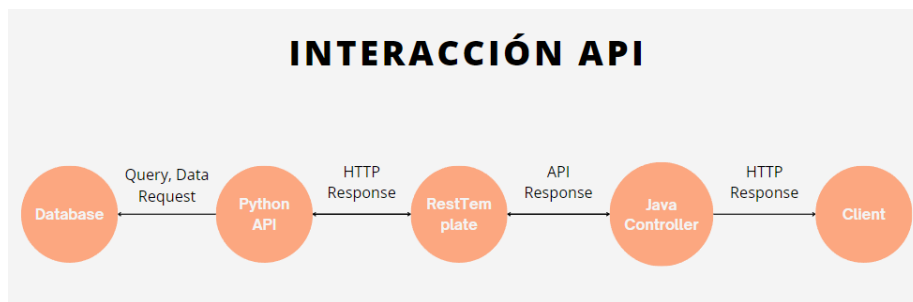


Figura C.6: Diagrama Interacción API, hecho con Canva

## Arquitectura de la Base de Datos

La base de datos que se ha utilizado es MySQL y las operaciones para la persistencia de los datos se realizan mediante JPA.

1. Las entidades se mapean a tablas de la BBDD.

2. Los repositorios se utilizan para realizar las operaciones CRUD.

Cabe destacar que al utilizar Spring Boot, parte de la gestión y arquitectura de la base de datos la realiza el mismo.

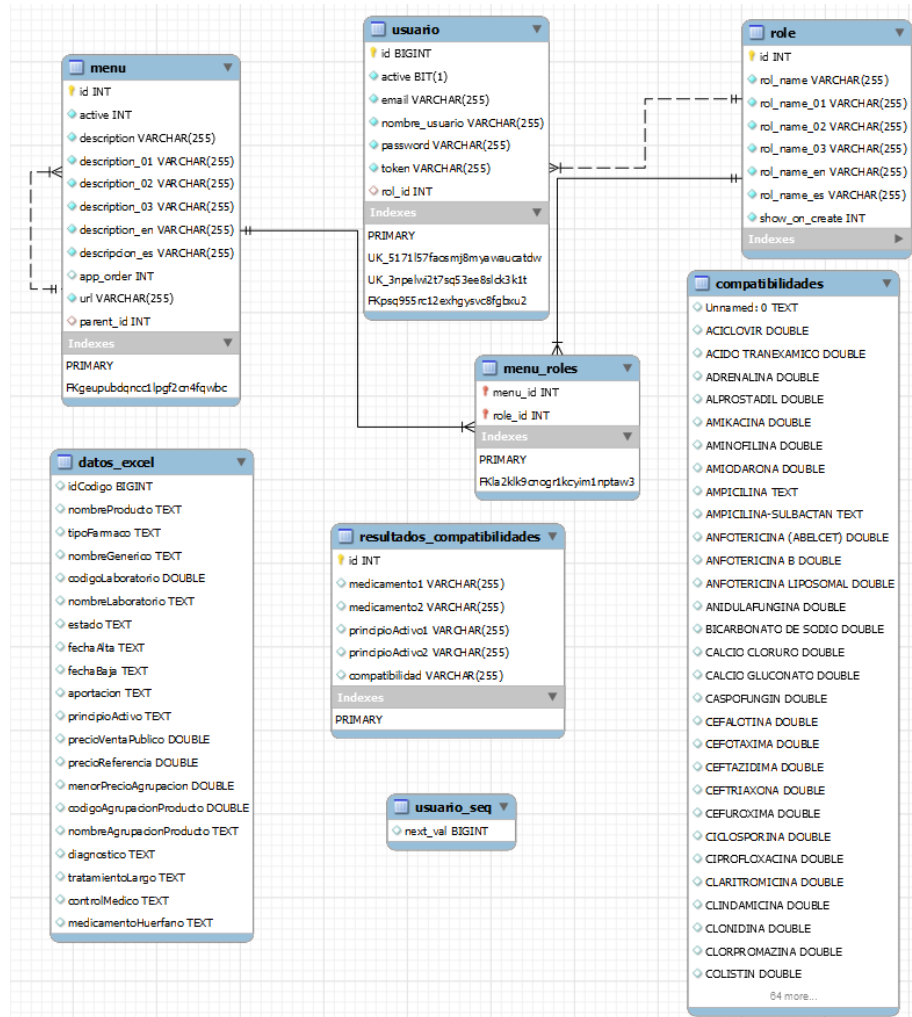


Figura C.7: Reverse Engineering MySQL Workbench

## C.5. Guía de Estilos

En esta sección se van a nombrar los colores principales de la aplicación, el logo de esta y otros aspectos sobre los estilos.

## Colores

La paleta de colores utilizada en la aplicación es principalmente la que incluía la plantilla Bootstrap que se ha utilizado. Principalmente son colores Grises, Azules y blancos. Muchos de los colores que aparecen no se han utilizado.

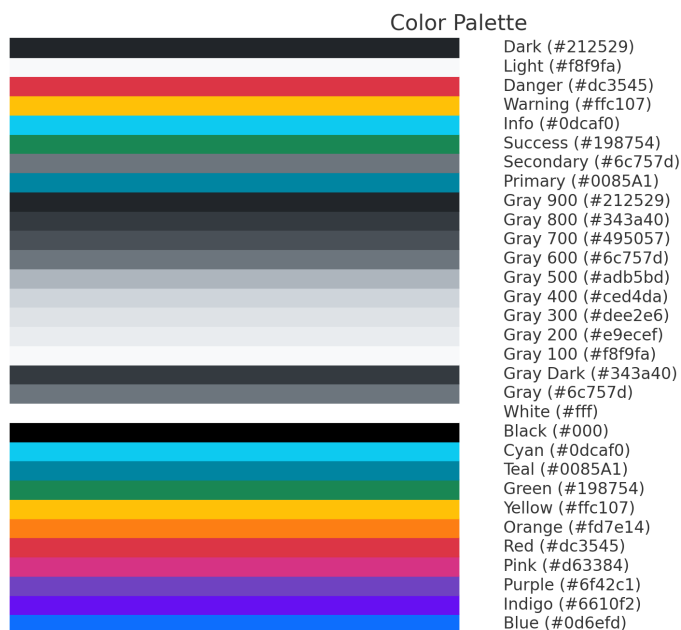


Figura C.8: Paleta de Colores de la plantilla Bootstrap

## Logo y Nombre

El nombre Activus nace de las interacciones de los medicamentos, al basarse para realizar las comparaciones en sus principios activos.

El logo se ha generado mediante Inteligencia Artificial, utilizando también colores azules y un estilo que se podría asemejar al ámbito de la investigación farmacéutica.

Creación del icono del Logo: <https://www.dall-e-free.com/>

Tipografía: <https://www.canva.com/>



Figura C.9: Logo Activus

Las imágenes utilizadas para las cabeceras y para otros apartados semejantes también se han generado con IA.

## Tipografía y Botones

La tipografía que se ha utilizado es la que se incluía en la plantilla. Es la siguiente: `font-family: .open Sans", -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, "Helvetica Neue", Arial, sans-serif, .apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol", "Noto Color Emoji";`

Para los botones y enlaces se han heredado los estilos del fichero CSS pero se han implementado cambios estéticos, como redondear border, añadir efectos de hover o sus tamaños y separaciones.



Figura C.10: Ejemplo de los botones

## Más Estilos

Para la gran mayoría de contenidos se han utilizado elementos Card para mostrar el texto o formularios, consiguiendo que la interfaz adquiriera una apariencia atractiva.



## REGISTRO

¡Crea una nueva cuenta para continuar!

Nombre de Usuario

---

Correo Electrónico

---

Contraseña

---

Roles:

☐ Administrator

☐ Anonymous

☐ Standard user

REGISTRARSE

INICIAR SESIÓN

Figura C.11: Vista de los contenidos dentro de un estilo que implementa Card



## *Apéndice D*

---

# Documentación técnica de programación

---

## D.1. Introducción

El proyecto contiene diferentes servicios que se van a ir desglosando en este apartado.

## D.2. Estructura de directorios

La estructura de directorios de la aplicación podemos dividirla en 3 secciones:

1. Aplicación Java o Backend
2. Thymeleaf para la interfaz gráfica o Frontend
3. API para la comunicación con servicios externos

### Backend Java

Todo el código esta dentro de la carpeta raíz Proyecto Revisar, se va a mantener este nombre ya que la aplicación no esta terminada.

Las siguientes carpetas se encuentran dentro de `src/main/java/_proyect/jpa_code`:

- **api/**: Contiene el fichero que se encarga de manejar las peticiones HTTP relacionadas con el comparador de medicamentos.

- **config/**: Contiene los ficheros relacionados con la configuración de Spring Boot, la seguridad y usuarios.
- **dto/**: Contiene los ficheros que representan los objetos de transferencia de la aplicación.
- **model/**: Contiene los ficheros que representan los modelos de datos que se utilizan en la aplicación.
- **repository/**: Contiene los ficheros que representan las interfaces de repositorio que se utilizan en la aplicación.
- **service/**: Contiene los ficheros que proporcionan los servicios para realizar diferentes acciones en la aplicación.
- **service/mapper/**: Contiene los ficheros para poder mapear los objetos y las entidades de la aplicación.
- **util/**: Contiene diferentes ficheros, cada uno proporciona una utilidad en la aplicación, como pueden ser la validación de contraseñas o la generación de cadenas aleatorias para los tokens.
- **web/controller/**: Contiene los controladores que se necesitan en la aplicación.

Al mismo nivel que todas las carpetas anteriormente mencionadas está el fichero `JpaCodeApplication`, que es la clase principal de la aplicación. Dentro de `jpa_code` estaría el archivo `pom.xml` con las dependencias de Maven.

## Frontend

La siguiente estructura de carpeta estaría dentro de `src/main/resources`:

- **static/**: Contiene las carpetas y ficheros descargados de la plantilla Bootstrap.
  - **assets/img/**: Contiene las imágenes utilizadas en el frontend de la aplicación.
  - **css/**: Contiene el archivo de estilos utilizados en la aplicación.
  - **js/**: Contiene scripts de la aplicación.

- **templates/**: Contiene las carpetas y ficheros que contienen las vistas de la aplicación.
  - **aplicacion/**: Contiene las vistas de las funcionalidades para los medicos que utilicen la aplicación.
  - **fragments/**: Contiene scripts que estan en archivos HTML, se utilizan como fragmentos de código para incorporarlos en diferentes vistas.
  - **gestion/**: Contiene las vistas para la gestión de usuarios, algunas en desarrollo.
  - **layout/**: Contiene el layout de la aplicación.
  - **users/**: Contiene diferentes vistas de la aplicación relacionadas a las funcionalidades de la aplicación, como el login o el reseteo de contraseña.

Al mismo nivel que las carpetas anteriormente nombradas se encuentran los ficheros que contienen los mensajes utilizados en la aplicación para conseguir el cambio de idioma. También se encuentra uno de los ficheros más importantes, el archivo `application.properties`, que contiene diferentes propiedades de la aplicación, como la codificación, la configuración para el envío de correos electrónicos o el usuario y contraseña para la conexión con la BBDD.

## API

Todo el contenido esta dentro de la carpeta FICHEROS API, la estructura de carpetas de la API es la siguiente:

- **API/app/**: Contiene los ficheros que conforman la API. Dentro también se encuentran los archivos excel de datos y la tabla de compatibilidades usada en la aplicación.
  - **python\_modules/**: Contiene los archivos necesarios para crear el entorno para desarrollar y levantar la API.

## D.3. Manual del programador

En este apartado se detallan varios aspectos relevantes para el programador.

## Variables de Entorno

Las variables de entorno se utilizan para configurar diferentes valores (se les llama valores secretos) sin necesidad de incluirlos en el código fuente.

### Variables de Entorno Flask y SQLAlchemy

El API utiliza un entorno virtual que se ejecuta en la raíz de los ficheros del API, se puede ejecutar también en la carpeta `app/`.

### Variables de Entorno Thymeleaf y Spring

Utiliza el archivo `application.properties` que está en el directorio `src/main/resources` para la configuración de las variables de entorno, son configuraciones del servidor y de la base de datos.

- **spring.datasource.url:** Contiene la URL de la conexión a la base de datos.
- **spring.datasource.username:** Contiene el nombre de usuario de la base de datos.
- **spring.datasource.password:** Contiene la contraseña de la base de datos.
- **spring.jpa.hibernate.ddl-auto:** Contiene la configuración de Hibernate para la gestión de las tablas de la base de datos.
- **server.port:** Contiene el puerto en el que se ejecuta el servidor.

Hay más variables de entorno en el archivo `application.properties`, como podría ser la codificación o configuración del email.

## Configuración Servicio de Correo

Aquí se detalla la configuración del servicio de correo y del dominio en la siguiente serie de capturas de pantalla:

Se debe entrar en el proveedor del hosting del dominio y en la página oficial de Mailtrap, tener un usuario en ambas páginas: <https://mailtrap.io/>

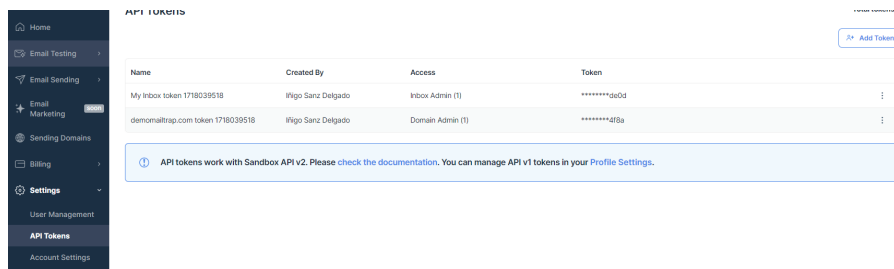


Figura D.1: Captura donde se muestra el token de Mailtrap que hay que poner en la configuración del fichero `application.properties`

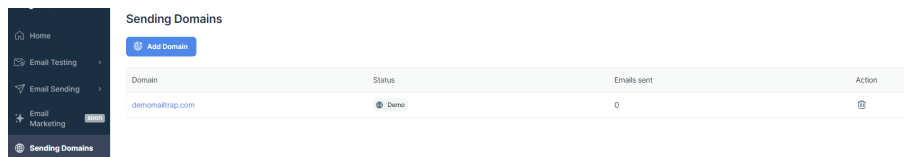


Figura D.2: Captura donde hay que añadir el dominio requerido en Mailtrap

### Dominio

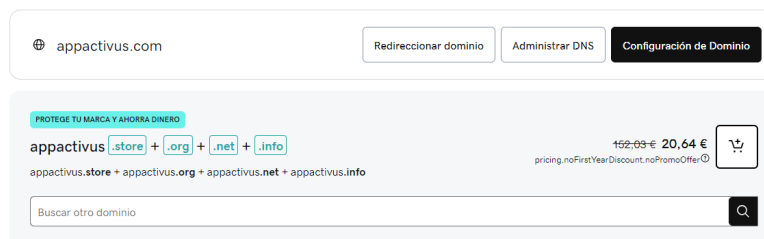


Figura D.3: Captura del panel del hosting del dominio, para ir a la siguiente imagen hay que pulsar en administrar DNS

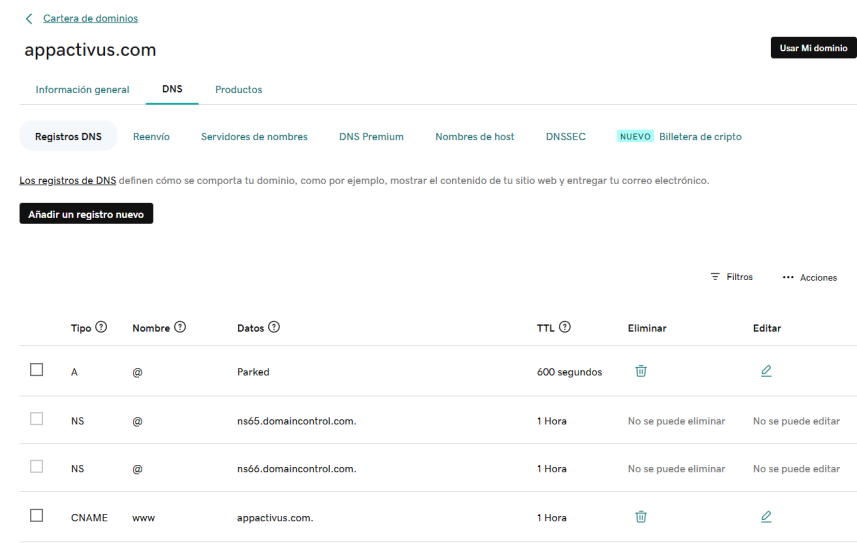


Figura D.4: En esta captura se muestra la configuración de los registros de DNS de nuestro dominio, para ir a la siguiente página hay que pulsar en añadir un registro nuevo

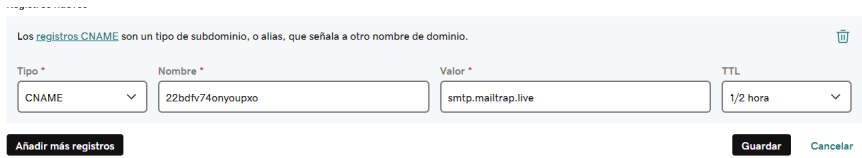


Figura D.5: Ene sta captura se indican los valores del nuevo registro, los cuales los tenemos que copiar de Mailtrap, siguiente imagen



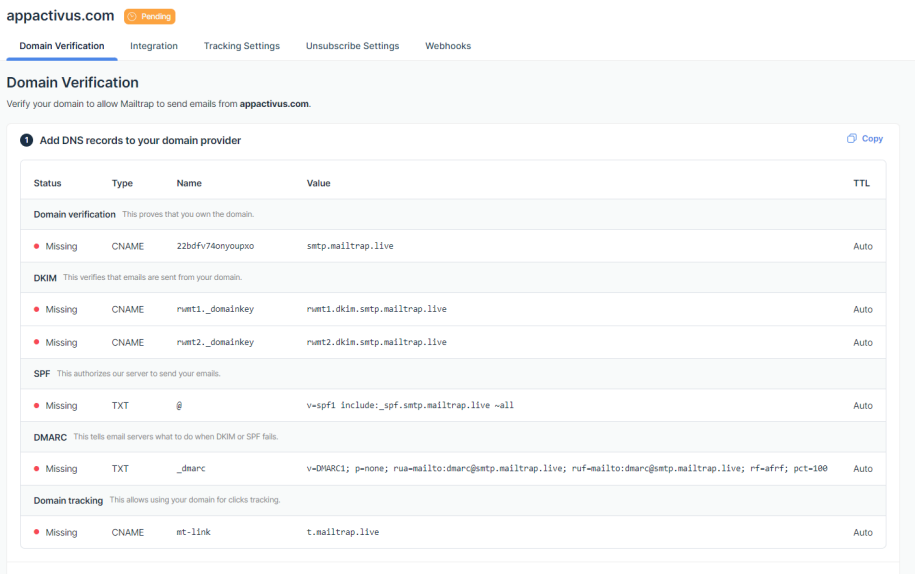


Figura D.6: Captura Mailtrap donde se muestran todos los registros que tenemos que añadir en nuestro dominio

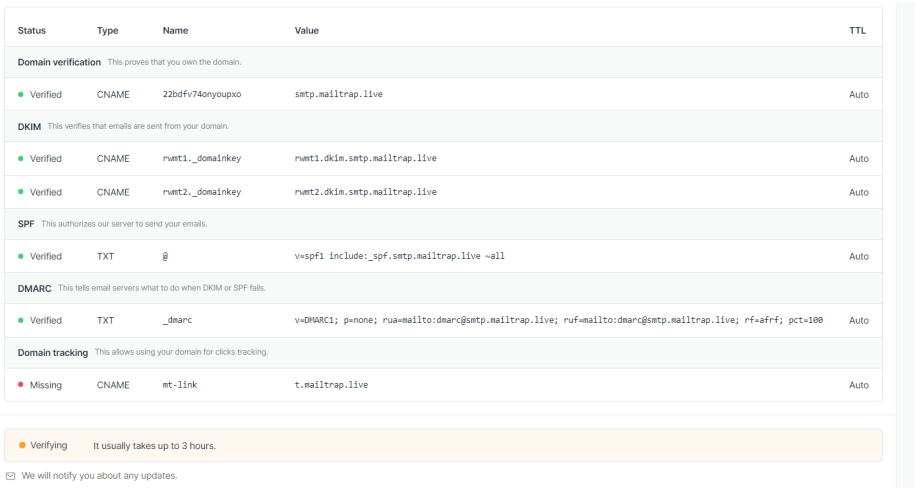


Figura D.7: Captura donde se muestra como van cambiando los estados de los registros según se vayan añadiendo al dominio

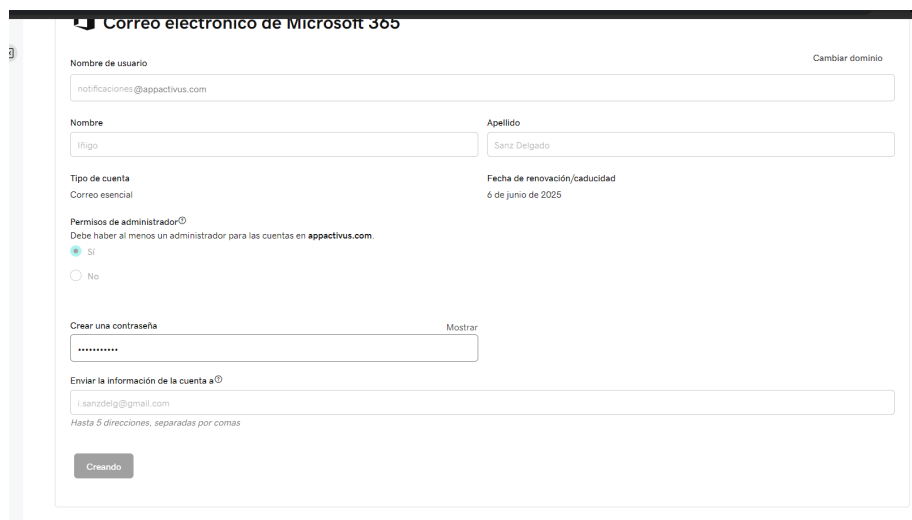
The image shows a web interface for creating a Microsoft 365 email account. At the top, it says 'Correo electrónico de Microsoft 365'. Below this, there are several input fields: 'Nombre de usuario' (Username) with the value 'notificaciones@appactivus.com', 'Nombre' (First Name) with 'Ifigo', and 'Apellido' (Last Name) with 'Sanz Delgado'. There is a 'Cambiar dominio' link next to the username field. Below these, 'Tipo de cuenta' (Account type) is set to 'Correo esencial' (Essential mail), and 'Fecha de renovación/caducidad' (Renewal/expiration date) is '6 de junio de 2025'. A section for 'Permisos de administrador' (Administrator permissions) asks if there should be at least one administrator for the accounts in 'appactivus.com', with 'Sí' (Yes) selected. There is a 'Crear una contraseña' (Create a password) field with a 'Mostrar' (Show) link. At the bottom, there is a field to 'Enviar la información de la cuenta a' (Send account information to) with the email 'i.sanzdelg@gmail.com' and a note 'Hasta 5 direcciones, separadas por comas' (Up to 5 addresses, separated by commas). A 'Creando' (Creating) button is at the bottom left.

Figura D.8: Captrua donde debemos crear el correo de donde saldrán las notificaciones, la contraseña habrá que añadirla también en la configuración de la aplicación

## D.4. Compilación, instalación y ejecución del proyecto

En esta sección se va a explicar lo necesario para la ejecución del proyecto.

### Instalación

Para ejecutar cada una de las partes que componen la aplicación haremos uso de diferentes programas:

La aplicación está pensada para ejecutarse en un entorno con el sistema operativo Windows, por lo que los siguientes apartados harán referencia a este SO. La estructura o ubicación de las instalaciones dependerá de cada usuario.

### WAMP

1. Descargamos el instalador de WAMP, en este caso desde la página: <https://wampserver.uptodown.com/>
2. Ejecutamos el instalador.

A pesar de utilizar WAMP, se podría utilizar XAMPP de la misma manera. En WAMP no se ha realizado ningún cambio, se ha utilizado la configuración por defecto que viene.

### MySQL Workbench

Para la visualización y gestión de la base de datos se ha utilizado MySQL Workbench.

1. Descargamos el instalador correspondiente, página oficial:  
<https://www.mysql.com/>
2. Ejecutamos el instalador.

En MySQL Workbench se ha creado una conexión con el servidor WAMP utilizando un usuario **root** y la dirección localhost/3306 utilizando la contraseña **proyectoFinal**.

## MySQL Connections



Figura D.9: Conexión con WAMP Server

Luego se ha creado el esquema **bbdd\_proyecto** de la siguiente manera:

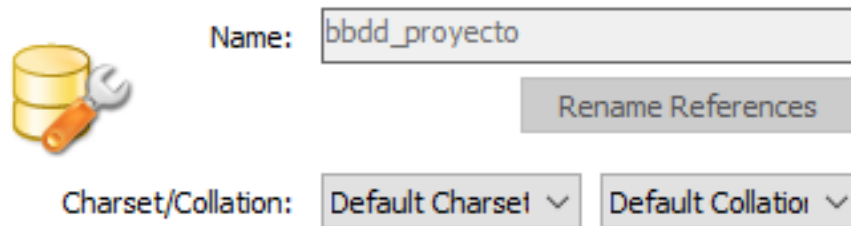


Figura D.10: Esquema utilizado para el proyecto

## IntelliJ

Para la parte de backend de la aplicación se ha utilizado IntelliJ, pero se podría usar sin ningún problema VS Code o Eclipse por ejemplo.

1. Descargamos el instalador correspondiente, página oficial JetBrains:  
<https://www.jetbrains.com/>
2. Ejecutamos el instalador.

En este IDE no se ha hecho ninguna instalación específica, si que es necesario descargarse e implementar el JDK de Java que vayamos a utilizar por ejemplo. De todas formas, lo bueno de un IDE como IntelliJ es que nos irá diciendo lo necesario para ir implementando lo necesario.

Las dependencias que se necesitan están en el archivo pom.xml de Maven que se incluye en el proyecto.

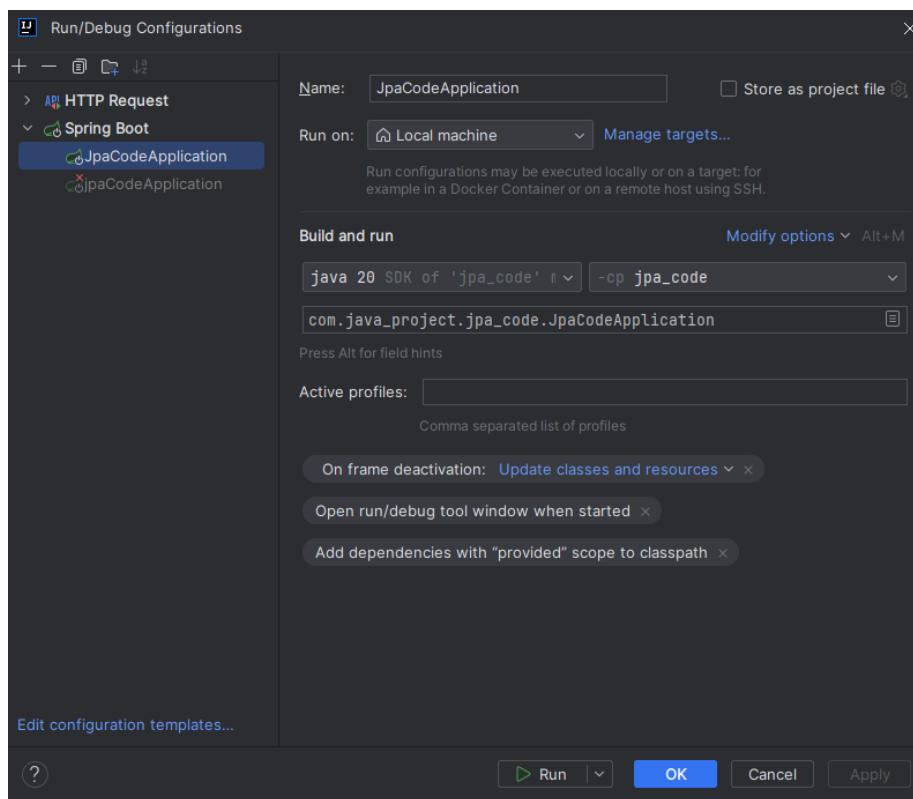


Figura D.11: Ejecución Spring donde se indica el JDK de Java utilizado

## Visual Studio Code

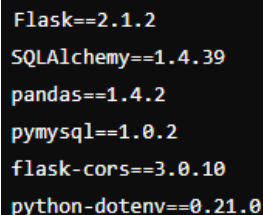
Este IDE se ha utilizado para desarrollar la API, pero podría valer al igual que en el apartado anterior, otros IDEs como Eclipse.

1. Descargamos el instalador correspondiente, página oficial:  
<https://code.visualstudio.com/>
2. Ejecutamos el instalador.

En el caso de este entorno que se ha configurado para desarrollar la API si que hay que descargar e instalar diferentes librerías.

- Para la instalación y configuración del entorno se han seguido los pasos dictados en el siguiente vídeo: <https://www.youtube.com/>
- Descargar Python desde la web oficial <https://www.python.org/>, versión 3.0 o superior recomendable.

- Para la instalación de las dependencias se podría crear un archivo `requirements.txt` y poner el contenido en el, y luego utilizar el comando:  
`pip install -r requirements.txt`

A screenshot of a dark-themed text editor showing the contents of a requirements.txt file. The text is as follows:

```
Flask==2.1.2
SQLAlchemy==1.4.39
pandas==1.4.2
pymysql==1.0.2
flask-cors==3.0.10
python-dotenv==0.21.0
```

Figura D.12: Archivo `requirements.txt` para la instalación de dependencias

- La otra manera sería ir ejecutando comando a comando cada una:
  - `pip install Flask`
  - `pip install SQLAlchemy`
  - `pip install pandas`
  - `pip install pymysql`
  - `pip install flask-cors`
  - `pip install python-dotenv` (opcional)

Al igual que IntelliJ, VS Code nos avisará de que necesitamos instalar las dependencias, por lo que no debería haber ningún problema.

## POSTMAN

Postman es una aplicación que permite realizar pruebas con APIs, realizando peticiones a servicios web y pudiendo visualizar las respuestas.

1. Descargamos el instalador correspondiente, página oficial:  
<https://www.postman.com/>
2. Ejecutamos el instalador.

## D.5. Ejecución de la Aplicación

Para hacer una demostración de la ejecución de la aplicación se ha grabado un vídeo [4].

## D.6. Pruebas del sistema

En este apartado se indican las herramientas utilizadas para realizar pruebas durante el desarrollo.

### POSTMAN

Es la herramienta que se ha utilizado para realizar las pruebas de los puntos de la API.

De esta manera se ha podido comprobar el funcionamiento y ejecución de cada una de las funcionalidades. Desde levantar la API, cargar los datos necesarios, hasta hacerle consultas sobre los medicamentos.

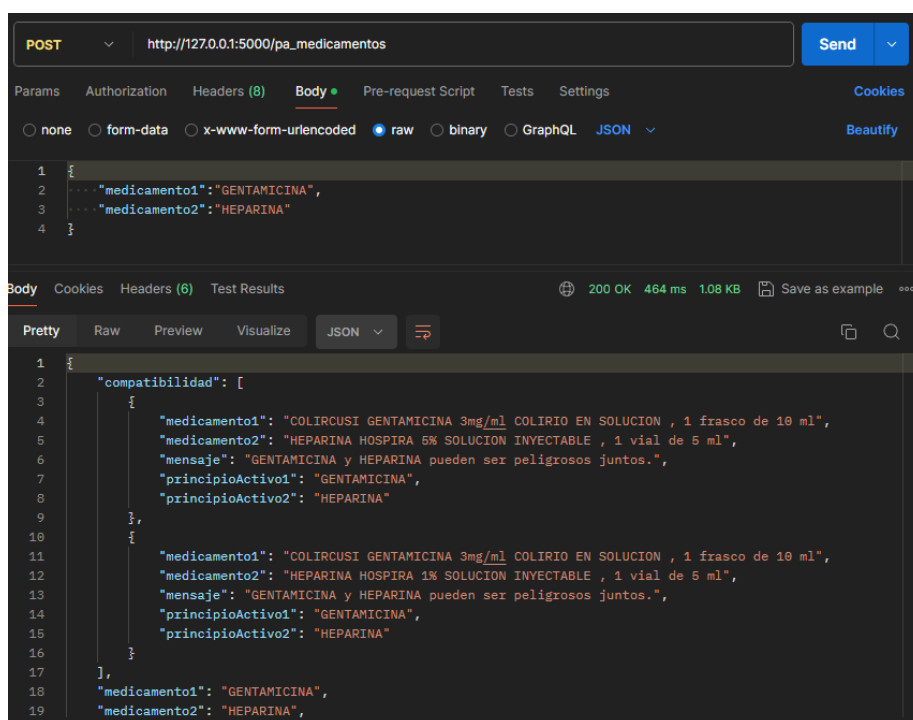


Figura D.13: Ejemplo del uso de Postman para consultar la compatibilidad de dos medicamentos





## *Apéndice E*

---

# Documentación de usuario

---

### E.1. Introducción

En este apartado se recogen varios aspectos de cara a los usuarios de la aplicación.

### E.2. Requisitos de usuarios

Los requisitos son realmente básicos, entro otros:

- Disponer de un ordenador, no es necesario que sea un equipo de alta gama, al ser una aplicación web no requerirá de muchos recursos locales.
- Utilizar un navegador compatible como Google Chrome o Mozilla Firefox.
- Disponer de conexión a internet (futuro).

### E.3. Manual del usuario

Se adjunta un vídeo junto con la documentación donde se muestran las opciones que tiene un usuario utilizando la aplicación [5].



---

## Bibliografía

---

- [1] Gobierno de España. Ley orgánica 3/2018, de 5 de diciembre, de protección de datos personales y garantía de los derechos digitales. <https://www.boe.es/buscar/act.php?id=BOE-A-2018-16673>, 2018.
- [2] Gobierno de España. Código sanitario. [https://www.boe.es/biblioteca\\_juridica/codigos/codigo.php?id=84&modo=2&nota=0&tab=2](https://www.boe.es/biblioteca_juridica/codigos/codigo.php?id=84&modo=2&nota=0&tab=2), Rango fechas.
- [3] Iñigo Sanz Delgado. Documentación javadoc del proyecto. Archivo local: doc/javadoc/index.html, 2024.
- [4] Iñigo Sanz Delgado. Video ejecución aplicación. Archivo local: videoEjecucion.mp4, 2024.
- [5] Iñigo Sanz Delgado. Video para usuarios. Archivo local: videoUsuarios.mp4, 2024.