MEMORIA ESCRITA DEL PROYECTO

CFGS Desarrollo de Aplicaciones Web

**Dentalium, gestor de pacientes en una clínica dental**

**Autor:** Roberto Moreno Muñoz

**Tutor:** Alberto Castedo Espeso

**Fecha de entrega:** 24/04/2024

**Convocatoria:** 2S2425

**Documentos del proyecto: Enlace a la carpeta del Drive**

Índice de contenidos

[**1. Introducción (3-4 Páginas) 3**](#_heading=h.1fob9te)

[**1.1. Motivación 3**](#_heading=h.3znysh7)

[**1.2. Abstract 3**](#_heading=h.2et92p0)

[**1.3. Objetivos propuestos (generales y específicos) 3**](#_heading=h.tyjcwt)

[**2. Estado del Arte (4-5 páginas) 4**](#_heading=h.3dy6vkm)

[**3. Metodología usada (2-4 páginas) 5**](#_heading=h.1t3h5sf)

[**4. Tecnologías y herramientas utilizadas en el proyecto  
(2-3 páginas) 6**](#_heading=h.4d34og8)

[**5. Planificación, Diagnóstico y Contexto Laboral (1-2 páginas) 7**](#_heading=h.2s8eyo1)

[**6. Análisis del proyecto (8-16 páginas) 8**](#_heading=h.17dp8vu)

[**7. Diseño del proyecto (6-14 páginas) 9**](#_heading=h.3rdcrjn)

[**8. Despliegue y pruebas (4-10 páginas) 10**](#_heading=h.26in1rg)

[**9. Conclusiones (1-2 páginas) 11**](#_heading=h.lnxbz9)

[**10. Vías futuras (1-2 páginas) 12**](#_heading=h.35nkun2)

[**11. Bibliografía/Webgrafía (1-2 páginas) 13**](#_heading=h.1ksv4uv)

[**12. Anexos 14**](#_heading=h.44sinio)

[**1.1. Manual de Usuario 14**](#_heading=h.2jxsxqh)

***En la normativa de proyectos vigente encontrarás una breve descripción de cada uno de estos apartados para saber qué información debes incluir en ellos***

# Introducción (3-4 Páginas)

## Motivación

## Abstract

## Objetivos propuestos (generales y específicos)

Lll

**Objetivo General:**

**OG-01**: Posibilitar a los profesionales de la odontología una correcta gestión de los pacientes de una clínica dental

**Objetivos Específicos:**

**OE-01**: Permitir a los usuarios (profesionales y pacientes) **acceder de forma diferenciada a la aplicación**

**OE-02**: Permitir al profesional **emitir presupuestos** que pueden ser consultados por el paciente

**OE-03**: Permitir al profesional **incluir estudios clínicos** (incluido el formato DICOM) que pueden ser obtenidos y exportados por el paciente

**OE-04**: Permitir al paciente **consultar sus citas**, **confirmar su asistencia** y **obtener justificante de asistencia firmado** tras la cita.

# Estado del Arte (4-5 páginas)

# Metodología usada (2-4 páginas)

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

Esquema de fases de la metodología en cascada sin retroalimentación (Company Ibañez, 2024

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

# Tecnologías y herramientas utilizadas en el proyecto (2-3 páginas)

Kkk

**Tipo de aplicación a realizar**: Se realizará una aplicación web ya que, debido a la temática del proyecto (en la que tanto profesional como paciente deben hacer uso de ella) parece más conveniente que el uso de una aplicación de escritorio (que obligaría a tenerla instalada en un ordenador) o el uso de una aplicación móvil (que no es práctica para el uso en el gabinete del dentista).

Siguiendo las actuales tendencias de bajo acoplamiento, la aplicación dispondrá de un frontend y de un backend siguiendo la **arquitectura cliente-servidor**.

**Tecnologías utilizadas en el backend**:

Se realizará una **API REST**, utilizando **Java (JDK 17)** utilizando el framework **Spring Boot** (versión 3.4) que posee embebido el contenedor de servlets **Tomcat**.

Como ORM de persistencia se empleará **Hibernate**, una implementación de JPA que viene incluida en el módulo de **Spring Data JPA**.

La gestión de la seguridad se realizará utilizando **tokens JWT** y securizando los endpoints mediante roles gestionados con **Spring Security**.

Se intentará evitar el código *boilerplate* provocado por la verbosa repetición de getters y setters mediante técnicas como el uso de records (estructuras implementadas a partir de Java 14) o la **librería Lombok**.

Se utilizarán informes **JaspersReports** para la generación de PDFs (presupuestos y justificantes)

Asimismo, se empleará la librería **dcm4che** para el tratamiento de ficheros médicos DICOM así y la librería **itextpdf** de Java para firmar los PDFs generados.

Podrá utilizarse **JUnit** para realizar algún test unitario (a modo ilustrativo) para la fase de pruebas.

De cara a la correcta documentación de los endpoints en la fase de diseño de la memoria final, podría utilizarse **Swagger API**.

**Tecnologías utilizadas en el frontend**:

Se empleará **Angular** (lenguajes **JavaScript** y **TypeScript**), así como los lenguajes de marcado **HTML** y **CSS**. Se implementarán estilos y estéticas acordes a las recomendaciones dadas por **Material Design** de Google.

Se utilizarán las librerías **Bootstrap** (para mejorar el estilo CSS, ayudando al comportamiento responsive de la página) y **JQuery** (cuando se considere necesario su uso en JavaScript).

Igualmente se emplearán peticiones **AJAX** para comunicarse con el backend sin necesidad de renderizar toda la página. No se descarta el uso (un uso completamente mínimo) de la librería de componentes **PrimeNG** si en algún caso se considerase necesario, aunque se intentará promover siempre la implementación nativa en Typescript. El despliegue de la aplicación de front me planteo realizarlo mediante algún servidor apropiado para aplicaciones SPA (como **Apache** o **Nginx**)

**Base de datos**: Se utilizará como motor de base de datos **MySQL** y como programa de gestión **MySQL Workbench**.

**IDEs y herramientas de desarrollo**: Se utilizarán productos de la **familia de JetBrains** como **IntelliJ IDEA Ultimate** (idealpara backend en Java) y **WebStorm** (ideal para JavaScript y TypeScript). Se empleará **Postman** para probar las peticiones a la API. De cara al futuro despliegue de la infraestructura, se utilizará **Docker** (ficheros Dockerfile y docker-compose.yml)

**Equipo**: Se empleará un ordenador portátil convencional equipado de SO Windows 11 y herramientas básicas de ofimática (**Office 365**, **Notepad++**, **Obsidian** y **OBS Studio**).

**Plataformas externas**: Se utilizará **Git** como mecanismo para el control de versiones, almacenando el código en un repositorio privado de **GitHub**. Finalmente, el proyecto será entregado mediante **Google Drive**.

# Planificación, Diagnóstico y Contexto Laboral (1-2 páginas)

Lll

Escala de tiempo

El contenido generado por IA puede ser incorrecto.

Gráfico, Gráfico en cascada

El contenido generado por IA puede ser incorrecto.

**DAFO**

# Análisis del proyecto (8-16 páginas)

## Requisitos funcionales y no funcionales

El análisis comienza acordando puntos que permitan lograr los objetivos generales y específicos que se han planteado. Así, se redacta un pequeño esbozo de una Especificación de Requisitos de Software (documento ERS) destacando los requisitos funcionales y no funcionales.

### Requisitos funcionales

|  |  |
| --- | --- |
| **Objetivos específicos** | **Requisitos funcionales** |
| **OE-01**: Permitir a los usuarios (profesionales y pacientes) acceder de forma diferenciada a la aplicación | **RF-01.0**, **RF-01.1**, **RF-01.2**, **RF-01.3**, **RF-01.4**, **RF-01.5**, **RF-01.6**, **RF-01.7, RF-01.8** |
| **OE-02**: Permitir al profesional emitir presupuestos que pueden ser consultados por el paciente | **RF-02.0**, **RF-02.1**, **RF-02.2**, **RF-02.3**, **RF-02.4**, **RF-02.5**, **RF-02.6** |
| **OE-03**: Permitir al profesional incluir estudios clínicos (incluido el formato DICOM) que pueden ser obtenidos y exportados por el paciente | **RF-03.0**, **RF-03.1**, **RF-03.2** |
| **OE-04**: Permitir al paciente consultar sus citas, confirmar su asistencia y obtener justificante de asistencia firmado tras la cita. | **RF-04.0**, **RF-04.1**, **RF-04.2**, **RF-04.3**, **RF-04.4** |

Se listan todos los requisitos funcionales, destacando palabras útiles en color azul para la construcción del modelo de datos en el apartado 6.2.

* **RF-01.0:** El sistema permite la existencia de usuarios con tres roles diferenciados: Administrador, Dentista y Paciente.
* **RF-01.1:** El usuario con rol de administrador (en adelante administrador) puede registrar usuarios con rol de dentista o con rol de paciente.
* **RF-01.2:** El administrador puede solicitar el envío de contraseña vía correo electrónico al resto de usuarios.
* **RF-01.3:** El administrador puede asignar un dentista a un paciente y modificar esta asignación posteriormente.
* **RF-01.4:** El administrador puede gestionar los tratamientos de la clínica, añadiendo tratamientos nuevos.
* **RF-01.5:** El administrador puede desactivar usuarios y tratamientos.
* **RF-01.6:** El administrador puede eliminar a un usuario lo cual tendrá como efecto su eliminación o, si existe material relacionado con él, su asignación como usuario no válido
* **RF-01.7:** Cualquier usuario puede solicitar la renovación de su contraseña vía correo electrónico
* **RF-01.8**: El sistema conoce los datos personales básicos de cualquier usuario tales como email, usuario, contraseña, nombre y apellidos, teléfono. Del paciente se requiere además conocer su DNI y dirección completa; del dentista basta con conocer su número de colegiado.
* **RF-02.0:** El usuario con rol de dentista (en adelante dentista) puede visualizar todos los datos de los pacientes a él asignados.
* **RF-02.1:** El dentista puede crear uno o varios presupuestos para cualquiera de sus pacientes
* **RF-02.2.:** El dentista puede modificar el estado de un presupuesto creado (siendo posible tener como estado: pendiente de emitir, emitido, tratamiento en curso, rechazado por paciente, anulado por doctor, tratamiento finalizado)
* **RF-02.3:** El dentista puede añadir tratamientos al presupuesto del paciente, indicando la cantidad, las piezas afectadas y el descuento si lo hubiera
* **RF-02.4:** El dentista puede eliminar tratamientos del presupuesto del paciente
* **RF-02.5:** El usuario con rol de paciente (en adelante paciente) puede consultar los los presupuestos que se han generado para él.
* **RF-02.6:** Tanto el dentista como el paciente pueden imprimir un documento firmado electrónicamente con el presupuesto realizado por el dentista en el que se muestra una fotografía de las piezas afectadas por los tratamientos en él incluidos.
* **RF-03.0:** El dentista puede adjuntar estudios clínicos fotográficos al historial del paciente diferenciando entre los siguientes tipos: Perfil, Frontal, Sonrisa, Oclusal superior, Intraoral derecha, Intraoral frontal, Intraoral izquierdo, Oclusal inferior. Tanto él como el paciente pueden visualizar y/o descargar las fotografías, estando sus rutas almacenadas en el servidor y considerándose la extensión y tipo de contenido del fichero.
* **RF-03.1:** El dentista puede adjuntar ficheros DICOM (extensión .dcm) pudiendo tanto él como el paciente visualizar su contenido y sus metadatos.
* **RF-03.2:** El dentista puede eliminar estudios clínicos del historial del paciente.
* **RF-04.0:** El dentista dispone de agenda con la que puede visualizar las citas distribuidas por días e igualmente dispone de visor en el historial del paciente para consultar sus citas próximas y anteriores.
* **RF-04.1:** El dentista puede añadir citas nuevas para un paciente tanto desde su agenda como desde el historial del propio paciente indicando la fecha y hora de inicio, fecha y hora de final y las observaciones que considere.
* **RF-04.2:** El dentista puede indicar el estado de la cita del paciente, existiendo los estados: Programado, Confirmado, Asistido, No asiste y Rechazado.
* **RF-04.3:** El paciente puede consultar sus citas próximas y anteriores.
* **RF-04.4:** Tanto el dentista como el paciente pueden imprimir un certificado de cita programada o de asistencia a consulta firmado electrónicamente.

### Requisitos no funcionales

Estos requisitos harán posible conseguir el cumplimiento de todos los objetivos específicos. Por eso, en lugar de hacer una tabla de correspondencias, se mencionan directamente:

* **RNF-01.0**: La aplicación web funcionará de forma ligera, teniendo buenos resultados en cuanto a performance en la mayoría de dispositivos.
* **RNF-02.0**: La aplicación web tendrá un diseño mediante mecanismos que posibiliten aspirar a un diseño responsive en versiones futuras.
* **RNF-03.0**: Todos los componentes de la aplicación web serán fácilmente adaptables a un sistema de despliegue llevado a cabo mediante mecanismos de virtualización y/o CI/CD.
* **RNF-04.1**: Los mecanismos de autenticación y autorización son suficientemente seguros para evitar intrusión en el sistema o escalada de privilegios.
* **RNF-05.1**: La aplicación tendrá una interfaz intuitiva que permita la correcta usabilidad.

## Diagrama de entidad-relación

La amplia mayoría de los requisitos funcionales no podrían ser llevados a cabo sin mecanismos de persistencia de datos (existencia de usuarios, tratamientos, presupuestos, informes médicos, citas, etc.). A partir de los mencionados requisitos del diseño conceptual se puede obtener el siguiente diagrama de relación (puede verse ampliado en el anexo para no incumplir la normativa del proyecto que restringe las imágenes a 1/3 de página):

Diagrama

El contenido generado por IA puede ser incorrecto.

El diagrama se obtiene mediante el siguiente proceso:

Identificación de entidades y relaciones; de claves y de atributos

|  |  |
| --- | --- |
| **ENTIDADES** | **Atributos de entidad** |
| **USUARIO** | id, email, usuario, contraseña, nombre, primer\_apellido, segundo\_apellido, teléfono, activo, valido |
| **DENTISTA** | numero\_colegiado |
| **ADMINISTRADOR** | (Sin atributos, bastaría con indicar su rol al usuario) |
| **PACIENTE** | dni, direccion, provincia, municipio, código\_postal |
| **PRESUPUESTO** | id, fecha |
| **ESTADO\_PRESUPUESTO** | id, nombre |
| **TRATAMIENTO** | id, nombre, precio\_unitario, activo |
| **ESTADO\_CITA** | id, nombre |
| **TIPO\_ESTUDIO** | id, nombre, tipo\_contenido, extension, id |

\*Se introduce el atributo id para un mejor control de los registros, evitando así en algunos casos el uso de claves primarias compuestas.

|  |  |
| --- | --- |
| **RELACIONES** | **Atributos de relación** |
| usuario **ES** administrador, paciente, dentista |  |
| dentista **TIENE ASIGNADO** paciente | desde, hasta |
| dentista **TIENE CITA** con paciente | id, fecha\_inicio, fecha\_fin, observaciones |
| dentista **REGISTRA** estudio médico de paciente | fecha, ruta |
| estudio médico **ES** de un tipo\_estudio | -- |
| cita **SE ENCUENTRA EN** un estado\_cita | -- |
| paciente **RECIBE** presupuesto | -- |
| presupuesto **SE ENCUENTRA EN** un estado\_presupuesto | -- |
| presupuesto **CONTIENE** tratamiento | -- |

Cardinalidad de entidades y relaciones

Una vez determinadas las entidades y relaciones implicadas, se determinan sus cardinalidades efectuando razonamientos tales como:

|  |
| --- |
| dentista TIENE CITA con paciente (**TENER**)   * Un DENTISTA, ¿Con cuántos pacientes puede tener citas? 🡪 Cardinalidad de entidad “dentista” (0,n) * Un PACIENTE, ¿Cuántos dentistas pueden tener citas con él? 🡪 Cardinalidad de entidad “paciente” (0,n)   Cardinalidad de relación “tener”: **N:M.** |

No se desarrollan el resto de razonamientos de cardinalidad por motivos de extensión del proyecto y porque no es algo que sea expresamente requerido por la normativa. Las cardinalidades pueden verse tanto en el diagrama de entidad relación mostrado arriba, como en el diagrama relacionan mostrado abajo (ya que MySQL Workbench lo elabora mediante notación de pie de gallo -crow’s foot notation- lo que también permite visualizar fácilmente las cardinalidades)

Diagrama relacional

Efectuado el diseño conceptual se puede avanzar hacia el diseño lógico de la base de datos mediante el modelo de datos relacional. Para ello, se siguen las reglas de transformaciones de entidades y relaciones. A continuación se adjunta captura del diagrama relacional de MySQL Workbench (puede consultarse en formato ampliado en el anexo para no incumplir la norma de ocupar 1/3 de la página).

En este diagrama de MySQL Workbench se observa que los ids de las tablas son generados mediante un número secuencial (almacenado en tablas role\_seq, user\_seq, etc.) que ha sido creado por el propio MySQL al definir las tablas con id autogenerado.

Diagrama, Escala de tiempo

El contenido generado por IA puede ser incorrecto.

## Diagrama de casos de uso junto con cada una de sus especificaciones

De todos los requisitos funcionales y no funcionales, se puede extraer las reglas de negocio necesarias para representa los casos de uso.

Se identifican los siguientes actores:

* Administrador
* Dentista
* Paciente

Diagrama de casos de uso frontera:

UC\_1 – Gestionar tratamientos

UC\_2 – Gestionar usuarios

UC\_3 – Gestionar citas

UC\_4 – Gestionar estudios médicos

UC\_5 – Gestionar presupuestos

UC\_6 – Visualizar citas

UC\_7 – Visualizar estudios médicos

UC\_8 – Visualizar presupuestos

**CULMINAR**

Diagrama

El contenido generado por IA puede ser incorrecto.

Procedemos ahora a detallar las **especificaciones de cada caso de uso** en las páginas siguientes.

Los casos de uso UC\_A1 y UC\_A2 los dejaremos para el final por ser los de mayor envergadura, abordando primero el resto de casos.

Tabla 6. Especificación del caso de uso UC\_A3 (Elaboración propia, 2024)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador del caso de uso** | | | **UC\_A3** | |
| **Nombre del caso de uso** | | | Localizar tiendas de supermercados cercanas | |
| **Descripción breve** | | | Mostrar en un mapa supermercados cercanos a la ubicación del usuario siempre que este tenga conexión a internet y haya concedido permisos de ubicación a CompraPlus | |
| **Actor** | | | Usuario registrado y logueado | |
| **Precondiciones** | | | El usuario está registrado con una cuenta válida y se ha logueado en la aplicación  La aplicación tiene concedidos permisos de Internet (dados en la instalación) | |
| **Poscondiciones** | | | Se muestra el mapa con las tiendas de supermercados cercanas, mensaje informativo sobre la circunstancia que lo ha impedido o se deriva a otra pantalla. | |
| **Flujo de eventos: a.Secuencia normal** | | | | |
|  | | **Actor** | **Sistema** | |
| **1** | | Usuario entra en la vista “Tiendas cercanas” |  | |
| **2a** | |  | CompraPlus comprueba que el usuario ha concedido permisos de localización, siendo cierto | |
| **3a** | |  | CompraPlus comprueba que el dispositivo posea conexión a Internet, siendo cierto | |
| **4a** | |  | Se consultan a Google Places API los datos de ubicación de los supermercados Día, Eroski y Mercadona próximos en un radio de 2 kilómetros | |
| **5a** | |  | Se le muestra al usuario al usuario un mapa con su posición actual y los supermercados Día, Eroski y Mercadona situados en un radio de 2 kilómetros. Estos supermercados son marcados en el mapa con el logo del supermercado correspondiente | |
| **6a** | | El usuario pincha en alguno de los marcadores |  | |
| **7a** | |  | Aparece un pequeño cuadro de diálogo sobre el marcador indicando el nombre de la tienda. Por ejemplo.: “La Plaza Día” o “Eroski City” | |
| **8a** | |  | Si el usuario permanece en esta vista, CompraPlus comprueba la ubicación de usuario cada 5 segundos | |
| **9a** | | El usuario se mueve de su posición inicial una distancia mayor a 500 metros |  | |
| **10a** | |  | Se repiten las acciones desde el paso 2a | |
| **Flujo de eventos alternativo: b.No se han dado permisos de ubicación** | | | | |
| 2b | |  | CompraPlus comprueba que el usuario ha concedido permisos de localización, siendo falso | |
| 3b | |  | CompraPlus solicita al usuario que conceda permisos de ubicación mediante cuadro de diálogo proporcionado por el sistema | |
| 4b | | El usuario deniega el permiso |  | |
| 5b | |  | CompraPlus muestra al usuario cuadro de diálogo informándole de que debe ir a los Ajustes del dispositivo para conceder permiso de ubicación. Si el usuario acude allí, se producirá el flujo de ejecución normal. | |
| 6b | | El usuario sigue sin conceder permisos de ubicación | CompraPlus cambia a la pantalla de “Mis listas” | |
| **Flujo de eventos alternativo: c.No hay conexión a Internet** | | | | |
| 3c |  | | | CompraPlus comprueba que el dispositivo posea conexión a Internet, siendo falso |
| 3d |  | | | Se muestra aviso informativo al usuario. El mapa centrado en la posición del usuario |

Tabla 7. Especificación del caso de uso UC\_A4 (Elaboración propia, 2024)

|  |  |  |
| --- | --- | --- |
| **Identificador del caso de uso** | | **UC\_A4** |
| **Nombre del caso de uso** | | Indicar preferencias de modo oscuro |
| **Descripción breve** | | Permitir elegir el tema de la aplicación (modo claro u oscuro), almacenándolo como preferencia del usuario |
| **Actor** | | Usuario registrado y logueado por primera vez.  Usuario posee modo claro como tema por defecto en su dispositivo. |
| **Precondiciones** | | El usuario está registrado con una cuenta válida y se ha logueado en la aplicación por primera vez |
| **Poscondiciones** | | El usuario ve la aplicación según su preferencia de tema (claro u oscuro) |
| **Flujo de eventos: Secuencia normal** | | |
|  | **Actor** | **Sistema** |
| **1** | Usuario entra en la aplicación |  |
| **2** |  | CompraPlus muestra el tema por defecto del dispositivo (Modo claro) |
| **3** |  | CompraPlus almacena el tema por defecto del dispositivo como preferencia del usuario respecto al tema de la aplicación |
| **4** | Usuario pincha en el menú de la barra superior y marca la opción “Cambiar modo” |  |
| **5** |  | CompraPlus cambia el tema de la aplicación (Modo oscuro) |
| **6** | Usuario cierra sesión |  |
| **7** | Usuario inicia sesión con credenciales válidas |  |
| **8** |  | CompraPlus recupera la preferencia del usuario y muestra el tema de la aplicación según esta (Modo oscuro) |
| **9** | El usuario puede cambiar el tema de la aplicación cada vez que desee repitiendo el punto 4 |  |

Tabla 8. Especificación del caso de uso UC\_A5 (Elaboración propia, 2024)

|  |  |  |
| --- | --- | --- |
| **Identificador del caso de uso** | | **UC\_A5** |
| **Nombre del caso de uso** | | Ver información acerca de la aplicación |
| **Descripción breve** | | Mostrar información respecto al autor y propósito de la aplicación |
| **Actor** | | Usuario registrado y logueado |
| **Precondiciones** | | El usuario está registrado con una cuenta válida y se ha logueado en la aplicación |
| **Poscondiciones** | | El usuario ve la información sobre la aplicación y su autor |
| **Flujo de eventos: Secuencia normal** | | |
|  | **Actor** | **Sistema** |
| **1** | Usuario pincha en el menú de la barra superior y marca la opción “Acerca de” |  |
| **2** |  | Se muestra un cuadro de diálogo indicando el propósito de la aplicación y posibilitando al usuario aceptar el mensaje o dirigirse al perfil de Github de Roberto Moreno |
| **3** | Usuario pincha en “Ver Github” |  |
| **4** |  | Se redirige al repositorio de GitHub de Roberto Moreno abriéndose el navegador o la aplicación de Github, si el usuario la tiene instalada |

Tabla 9. Especificación del caso de uso UC\_A6 (Elaboración propia, 2024)

|  |  |  |
| --- | --- | --- |
| **Identificador del caso de uso** | | **UC\_A6** |
| **Nombre del caso de uso** | | Compartir en redes sociales |
| **Descripción breve** | | Permitir compartir listas de la compra y productos por aplicaciones de redes sociales instaladas en el dispositivo |
| **Actor** | | Usuario registrado y logueado |
| **Precondiciones** | | El usuario está registrado con una cuenta válida y se ha logueado en la aplicación.  El usuario se encuentra ante un listado de listas de la compra o listado de productos. |
| **Postcondiciones** | | El elemento que el usuario deseaba compartir (lista de la compra o producto) se comparte de forma exitosa en la aplicación deseada |
| **Flujo de eventos: Secuencia normal** | | |
|  | **Actor** | **Sistema** |
| **1** | El usuario marca la opción de “Compartir” dada sobre una lista de la compra o sobre un producto concreto en el listado de productos de supermercado |  |
| **2** |  | CompraPlus muestra cuadro de diálogo del sistema para que el usuario elija en qué app desea compartir la información |
| **3** | El usuario elije una de las aplicaciones |  |
| **4** |  | CompraPlus transmite la información a la aplicación elegida, tomando esta el control para realizar la acción de compartir |

Tabla 10. Especificación del caso de uso UC\_B1 (Elaboración propia, 2024)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador del caso de uso** | | | | **UC\_B1** |
| **Nombre del caso de uso** | | | | Registrarse |
| **Descripción breve** | | | | Posibilitar el registro del usuario |
| **Actor** | | | | Usuario no registrado |
| **Precondiciones** | | | | El usuario no está registrado |
| **Poscondiciones** | | | | El usuario queda parcialmente registrado, a la espera de validar su correo o recibe mensaje informativo sobre el por qué no ha sido posible |
| **Flujo de eventos: a.Secuencia normal** | | | | |
|  | | **Actor** | | **Sistema** |
| **1** | | El usuario rellena en la pestaña de registro los datos que se le solicitan (email y contraseña) | |  |
| **2a** | |  | | CompraPlus comprueba que todos los datos hayan sido completados, siendo cierto |
| **3a** | |  | | CompraPlus comprueba que las contraseñas sean iguales, siendo cierto |
| **4a** | |  | | CompraPlus comprueba que eixs |
| **5a** | |  | | CompraPlus solicita a Firebase la creación del usuario, devolviendo respuesta exitosa |
| **6a** | |  | | Se muestra al usuario mensaje indicándole que debe validar su correo electrónico para finalizar el registro |
| **Flujo de eventos alternativo: b.Datos de registro incompletos** | | | | |
| **2b** | |  | | CompraPlus comprueba que todos los datos hayan sido completados, siendo falso |
| **3b** | |  | | Se muestra al usuario mensaje indicándole que debe completar todos los campos |
| **Flujo de eventos alternativo: c.Las contraseñas no coinciden** | | | | |
| **3c** | |  | | CompraPlus comprueba que las contraseñas sean iguales, siendo falso |
| **4c** | |  | | Se muestra al usuario mensaje indicándole que las contraseñas no coinciden |
| **Flujo de eventos alternativo: d.No hay conexión a Internet** | | | | |
| **4d** | |  | | CompraPlus comprueba que el dispositivo posea conexión a Internet, siendo cierto |
| **5d** | |  | | Se muestra al usuario mensaje indicándole que no hay conexión. |
| **Flujo de eventos alternativo: e.Firebase devuelve error en el registro** | | | | |
| **5ª** | |  | | CompraPlus solicita a Firebase la creación del usuario, devolviendo respuesta errónea |
| **6ª** | |  | | Se muestra al usuario mensaje con el motivo. |
| **Identificador del caso de uso** | | | **UC\_B2** | | |
| **Nombre del caso de uso** | | | Validar cuenta de correo | | |
| **Descripción breve** | | | Finalizar el registro del usuario, validando su cuenta de correo electrónico | | |
| **Actor** | | | Usuario no registrado | | |
| **Precondiciones** | | | El usuario está parcialmente registrado: Se ha registrado en CompraPlus pero aún no ha validado su correo.  El correo aportado es un correo real. | | |
| **Postcondiciones** | | | El usuario queda completamente registrado | | |
| **Flujo de eventos: Secuencia normal** | | | | | |
|  | **Actor** | | **Firebase** | | |
| **1** |  | | Envía mensaje al correo electrónico del usuario con un link para que valide su registro | | |
| **2** | El usuario abre el mensaje y pincha en el link | |  | | |
| **3** |  | | Firebase le comunica que la validación de su registro ha sido exitosa | | |

|  |  |  |
| --- | --- | --- |
| **Identificador del caso de uso** | | **UC\_A1** |
| **Nombre del caso de uso** | | Gestionar listas de la compra |
| **Descripción breve** | | Crear listas de la compra incluyendo los productos preferidos por el usuario y el supermercado en el que podrá encontrarlos |
| **Actor** | | Usuario registrado y logueado |
| **Precondiciones** | | El usuario está registrado con una cuenta válida y se ha logueado en la aplicación |
| **Postcondiciones** | | La lista de la compra queda gestionada con la acción que se deseaba realizar: Ha sido creada, completada, compartida, eliminada o programado un recordatorio sobre ella; pudiéndose también acceder a sus productos agrupados por supermercado. |
| **Flujo de eventos: A1.I. Gestionar descripciones de la lista de la compra** | | |
|  | **Actor** | **Sistema** |
| **1** | Usuario entra en la vista “Mis listas” |  |
| **2** |  | CompraPlus devuelve todas las listas de la compra pertenecientes a dicho usuario |
| **3** | Usuario pulsa en añadir una nueva lista |  |
| **4** |  | CompraPlus le muestra formulario en el que puede elegir el nombre de la lista de la compra y la fecha de compra. |
|  | El usuario rellena los datos de la lista de la compra que desea crear |  |
| **5** |  | CompraPlus comprueba que los datos estén completos y crea la lista de la compra. Si algún dato está incompleto, se lo indica al usuario mediante cuadro de diálogo. |
| **6** | El usuario visualiza las listas creadas pudiendo **compartirlas** en redes sociales (Punto de extensión UC\_A6), **eliminarlas** previo mensaje de confirmación o **acceder a los productos** que contienen que le aparecerán agrupados según el supermercado |  |
| **Flujo de eventos: A1.II. Añadir producto manualmente** | | |
| **7(II)** | Tras elegir una lista de la compra, el usuario pulsa en la opción “Producto manual” |  |
| **8(II)** |  | CompraPlus muestra un cuadro de diálogo en el que el usuario puede introducir el nombre del producto, el supermercado (Día, Eroski o Mercadona) y la cantidad que desea adquirir |
| **9(II)** | El usuario rellena dichos datos |  |
| **10(II)** |  | CompraPlus comprueba que los datos estén completos y crea el producto. Si algún dato está incompleto, se lo indica al usuario mediante cuadro de diálogo. |
| **Flujo de eventos: A1.III. Añadir producto de supermercado** | | |
| **7(III)** | Tras elegir una lista de la compra, el usuario pulsa en la opción “Producto supermercado” o simplemente en la opción de “Añadir” |  |
| **8(III)** |  | CompraPlus muestra al usuario una barra de búsqueda para buscar el producto deseado |
| **9(III)** | El usuario introduce el producto deseado y le da a buscar |  |
| **10(III)** |  | CompraPlus muestra productos de los tres supermercados ya mencionados, ordenados de mayor a menor precio (Inclusión de UC\_A2.I, Listar productos de supermercados) |
| **11(III)** | El usuario elige el producto deseado, que se añade a la lista |  |
| **Flujo de eventos: A1.IV. Eliminar producto** | | |
| **7(IV)** | El usuario marca sobre cierto producto para eliminarlo o darlo por adquirido |  |
| **8(IV)** |  | CompraPlus, según se desee eliminar o dar por adquirido el producto, mostrará un mensaje para confirmar la eliminación o tachará el producto de la lista |
| **Flujo de eventos: A1.V. Establecer recordatorio** | | |
| **7(V)** | El usuario indica que desea “Establecer recordatorio” en una lista de la compra |  |
| **8(V)** |  | CompraPlus muestra un cuadro de diálogo con un reloj para indicar la hora de la fecha de compra en la que desea establecerse el recordatorio |
| **9(V)** | El usuario introduce la hora |  |
| **10(V)** |  | CompraPlus indica al teléfono que debe notificar al usuario |
| **11(V)** |  | El día y hora indicados, el dispositivo envía una notificación al usuario |

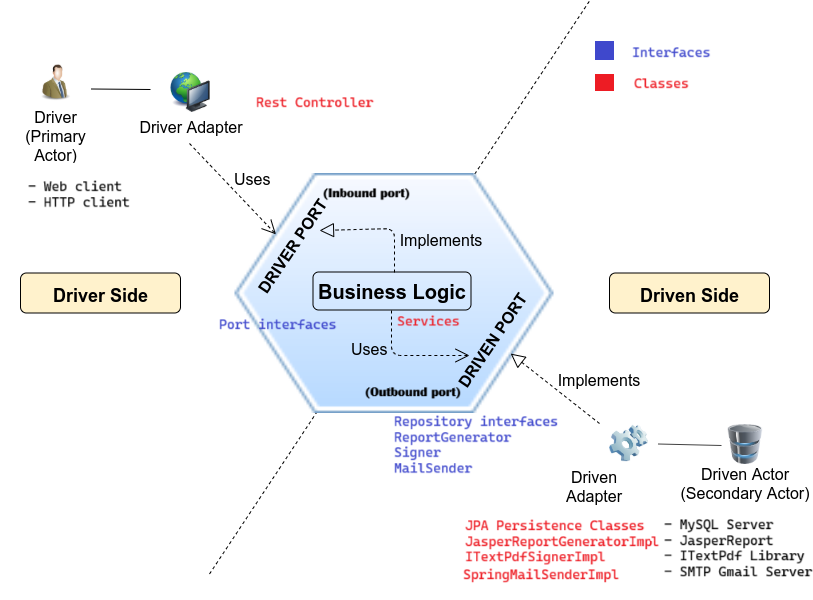
Tabla 12. Especificación de caso de uso UC\_A2 (Elaboración propia, 2024)

|  |  |  |
| --- | --- | --- |
| **Identificador del caso de uso** | | **UC\_A2** |
| **Nombre del caso de uso** | | Comparar precios de supermercados |
| **Descripción breve** | | Permitir al usuario buscar productos en los principales supermercados españoles y conocer su precio actual |
| **Actor** | | Usuario registrado y logueado |
| **Precondición** | | El usuario está registrado con una cuenta válida y se ha logueado |
| **Flujo de eventos: A2.I. Listar productos de supermercados** | | |
|  | **Actor** | **Sistema** |
| **1** |  | CompraPlus muestra al usuario una barra de búsqueda para buscar el producto deseado |
| **2** | El usuario introduce el producto deseado y le da a buscar |  |
| **3** |  | CompraPlus muestra productos de los tres supermercados, ordenados de mayor a menor precio y mostrando su fotografía en miniatura (inclusión de A2.II) |
| **Flujo de eventos: A2.II. Ver fotografías de los productos de los supermercados** | | |
| **4(II)** | El usuario pincha sobre la fotografía de los productos |  |
| **5(II)** |  | CompraPlus amplía dicha fotografía y la muestra en un cuadro de diálogo. |

## Diagrama de clases

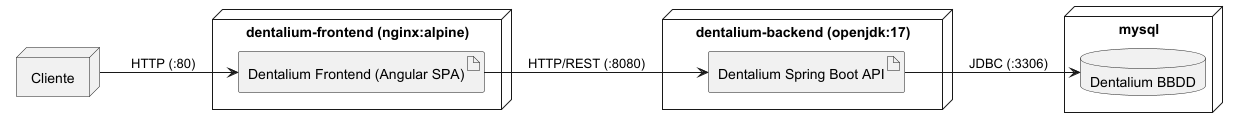
## Diagrama de clases

# Diseño del proyecto (6-14 páginas)



Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.



# Despliegue y pruebas (4-10 páginas)

La aplicación se compone de tres elementos:

* Frontend: Realizado en Angular 19.1.0, cuyo contenido es servido por un **servidor Nginx**
* Backend: Realizado en Spring Boot 3.4.3. con gestión de dependencias mediante Maven 3.9.9. y cuyo jar es ejecutado sobre **JDK 17**.
* Base de datos: Gestionada en **MySQL Server 8**.

Se plantean dos formas de llevar a cabo el despliegue:

**1.- Despliegue mediante Docker (con contenedores) -Opción recomendada-**

Esta es la opción recomendada por los siguientes motivos:

- El desarrollo será sea verdaderamente reproducible: Cualquier persona podrá ejecutarlo con los mismos resultados.

- Se pueden llevar a cabo tareas automatizadas del proceso de CI/CD mediante sistemas como Jenkins, GitHub Actions, etc. que permiten despliegues ágiles, escalado horizontal (se podrá conseguir alta disponibilidad del producto ejecutando múltiples instancias o contenedores de la misma aplicación para repartir la carga entre ellos en momentos de mayor demanda).

Texto

El contenido generado por IA puede ser incorrecto.

# Conclusiones (1-2 páginas)

La realización de este gestor de pacientes dentales me resulta motivador a nivel personal ya que me permite aplicar los conocimientos sobre desarrollo a realidades cotidianas. Recientemente he tenido la experiencia de visitar diversas clínicas y me ha sorprendido la escasa disponibilidad de software que interconecte y mejore la relación profesional-paciente. He querido hacer hincapié en funcionalidades que, por mi experiencia profesional como desarrollador, sé que son funcionalidades sencillas de implementar y de las que los software de gestión odontológica presentan carencia o es necesario un elevado precio para obtenerla (confirmación de citas, disponibilidad del estudio médico o del fichero de datos DICOM de las radiografías por parte del paciente; disponibilidad de justificantes de asistencia para el trabajo parecidos a los que son emitidos en los exámenes de Ilerna, etc.)

Igualmente también supone un impulso a mis objetivos profesionales en cuanto a aprendizaje. Me considero un desarrollador bastante encasillado en un perfil backend. En mi día a día trabajo como desarrollador Java y, aunque a nivel de back utilizo Spring Boot, a nivel de front me mantengo utilizando tecnologías obsoletas como JSF con muy escasa práctica de HTML y CSS y nulo uso de frameworks modernos de JavaScript como Angular. Este proyecto final me permite aproximarme más a un perfil full-stack y poner en práctica todos los conocimientos que están ahí pero necesitan ser periódicamente refrescados.

Respecto a los objetivos del ciclo, este proyecto consolida mi camino de aprendizaje en ILERNA. Con él demuestro la capacidad de desarrollar aplicaciones web usando tecnologías específicas, garantizando el acceso seguro a los datos y cumpliendo los mínimos de usabilidad, calidad y accesibilidad tal y como recoge el Real Decreto 686/2010, de 20 de mayo, por el que se establece el título de Técnico Superior en Desarrollo de Aplicaciones Web y se fijan sus enseñanzas mínimas

# Vías futuras (1-2 páginas)

# Bibliografía/Webgrafía (1-2 páginas)

# Anexos

# Manual de usuario

Manual de usuario del proyecto: 1. Introducción: Pequeña descripción de la aplicación (para qué es, funcionalidades de cada rol en la aplicación….. 2. Propósito general de la aplicación: Que soluciona tu aplicación. 3. Guía de Instalación: Paso a paso de como instalar la aplicación y cómo ejecutarla 4. Guía de Uso: por de cada rol junto con capturas de pantalla 5. Mensajes de error: Pantallazos y descripción de los posibles errores de la aplicación. 6. Preguntas o dudas más frecuentes: Apartado de preguntas y respuestas, similar a las FAQ

# Manual de instalación

La aplicación *Dentalium* no conlleva un proceso de instalación para el usuario ya que, al tratarse de una aplicación web, este únicamente debe acceder por medio de su navegador a la dirección que se le indique.

El proceso de “instalación” o, más correctamente expresado, el **proceso de despliegue** de la aplicación es tarea de la persona que ejerza las funciones de Administrador de Sistemas Informáticos en Red.

La aplicación se compone de tres elementos:

* Base de datos: Gestionada en **MySQL Server 8**.
* Backend: Realizado en Spring Boot 3.4.3. con gestión de sus dependencias mediante Maven 3.9.9. y cuyo JAR es ejecutado sobre **JDK 17**.
* Frontend: Realizado en Angular 19.1.0, cuyo contenido es servido por un **servidor Nginx 1.27.4**

Se plantean dos formas de llevar a cabo el despliegue:

**1.- Despliegue mediante Docker (con contenedores) -Opción recomendada-**

Esta es la opción recomendada para que el desarrollo sea reproducible (cualquier persona obtendrá los mismos resultados ya que nos abstraeremos de las configuraciones y parametrizaciones del host anfitrión). Para ello deben seguirse los siguientes pasos:

**Paso 1:** Es necesario **tener instalado y activo el servicio de Docker en el host anfitrión** (Windows requerirá para ello la instalación del Subsistema de Windows para Linux -WSL-). En este caso concreto las pruebas han sido realizadas en un sistema operativo Windows con WSL 2.3.24 y Docker Desktop 4.17, aunque precisamente la bondad de Docker es que es completamente irrelevante el sistema operativo y la versión que se utilice.

**Paso 2**: Se requiere **crear** **en el directorio raíz donde se ubica docker-compose.yml una serie de directorios y ficheros**. **De cara a la evaluación del proyecto se dispone ya de todos ellos, salvo del certificado digital**. Este, que no es imprescindible para la ejecución del proyecto, no se suministra debido a motivos de seguridad aunque podría utilizarse cualquier certificado p12 válido (como el certificado personal expedido por la FNMT).

Los ficheros y directorios son los siguientes:

* Directorio **data/sql** en el que se almacenará la copia de la BBDD
* Fichero **my.cnf** para autorizar el plugin de autenticación de MySQL con password nativo
* Fichero **nginx.conf** como configuración del servidor Nginx del frontend (escuchar en el puerto 80, redigir todas las rutas a index.html, etc.)
* Fichero **dentalium.sql** con las sentencias DML para crear y poblar la base de datos
* Directorio **documents** que contendrá el certificado digital para firmar (certificate.p12) y el directorio medical\_studies donde se alojarán los documentos de los estudios médicos.

**Paso 3**: Antes de realizar el despliegue, se deben **parametrizar las variables de entorno en el fichero docker-compose.yml.** **De cara a la evaluación del proyecto ya vienen todas indicadas salvo la contraseña del certificado digital** (CERTIFICATE\_PASSWORD), al no estar este incluido por motivos de seguridad.

En el servicio mysql:

* **MYSQL\_DATABASE**: Nombre de la base de datos (dentalium)
* **MYSQL\_ROOT\_PASSWORD**: Password del usuario root
* **MYSQL\_USER**: Password de un usuario no root
* **MYSQL\_PASSWORD**: Password de un usuario no root

En el servicio backend:

* **DB\_URL**: jdbc:mysql://mysql:3306/dentalium (URL de la base de datos en la red de Docker)
* **DB\_USER**: Usuario de la BBDD (puede ser root)
* **DB\_PASSWORD**: Password del usuario (si es root, es el mismo valor que en MYSQL\_ROOT\_PASSWORD)
* **CERTIFICATE\_PASSWORD**: Password el certificado digital que debe estar situado en ./documents/\*.p12
* **JWT\_SECRET**: JWT generado (Puede generarse en https://jwtsecret.com/generate)
* **MAIL\_USERNAME**: Email del SMTP para enviar emails
* **MAIL\_PASSWORD**: Password del SMTP para enviar emails
* **FRONTEND\_URL**: Dominio de frontend que realizará las peticiones (para autorizar el CORS, ya que son peticiones con autorización (Basic, Bearer,…) y el navegador exige que esté validado el CORS). En este valor se ha puesto http://localhost para el caso en el que se ejecuta la aplicación web desde un navegador ubicado en la máquina anfitrión.
* **FRONTEND\_URL\_2**: Segundo dominio de frontend para autorizarlo en el CORS. En este valor puede ponerse por ejemplo un dominio o la IP de la interfaz de red para permitir acceder desde un navegador externo al host anfitrión (se pondría aquí la IP que se encontrase en la interfaz de red consultándola con *ipconfig* o *ifconfig*).

**Paso 4:** **Si se quisiese que la URL del backend sea distinta a la dada por defecto** (<http://localhost:8080/dentalium/api>), **debe parametrizarse** la deseada en el fichero front/dentalium/src/environment/environment.ts . Esto puede ser necesario si, por ejemplo, se desea acceder desde un navegador externo al host anfitrión (se pondría aquí la IP que se encontrase en la interfaz de red de *ipconfig* o *ifconfig*). **De cara a la evaluación del proyecto no es necesario modificarlo.**

**Paso 4:** Levantar los contenedores.

Esto puede hacerse en el terminal mediante la ejecución del docker-compose.yml con el comando:

**docker-compose up**

De forma más cómoda también puede hacerse mediante el uso de IntelliJ IDEA Ultimate o de IntelliJ IDEA Community (instalando el plugin de Docker). En el proyecto se han guardado ya diversas configuraciones que permiten la ejecución de este Docker Compose en distintas modalidades (*Deploy* (hace docker compose up), *Deploy Always* (obliga a reconstruir siempre la imagen aunque ya se disponga de ella, docker compose up --build) y *Deploy Maven* (hace mvn clean install previamente al docker compose up --build).

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

**Paso 5:** Comprobar que las aplicaciones están correctamente levantadas accediendo a:

**Frontend: localhost**

**Backend: http://localhost:8080/dentalium/api/swagger-ui/index.html**

Referente a cómo funciona el Docker Compose cabe mencionar que los directorios backend y frontend disponen de un Dockerfile que, respectivamente:

* Genera la imagen de JDK y toma del directorio *target* el comprimido JAR, indicandole al contenedor el comando encargado de ejecutarlo. (¡Ojo!: Previamente habrá que haber generado el target mediante el comando de Maven mvn clean install . Esta labor NO es realizada por este Dockerfile luego cambios en el código no se verán reflejados si simplemente se regenera la imagen, sin recompilar el proyecto).
* Genera la imagen de Nginx, haciendo previamente *build* de la distribución del frontend e incluyéndola después en el directorio encargado de servir el contenido web en Nginx.

El fichero Docker Compose organiza los tres servicios (mysql, backend y frontend) ejecutando las imágenes (y, si no las tiene aún, generándolas mediante el Dockerfile que hemos mencionado) y montando los volúmenes en nuestro host dentalium.sql , /data/mysql , /documents/medical\_studies y /nginx.conf

Los tres servicios están conectados mediante una red virtual interna denominada *dentalius*, en modo bridge (los contenedores se comunican entre sí y, se comunican con la red externa por los puentes virtuales -mapeo de puertos- establecidos entre la máquina anfitriona y el contenedor). Esto debe ser considerado, como hemos mencionado más arriba, a la hora de definir la variable de entorno DB\_URL (Por eso es mysql://mysql:3306/dentalium)

**2.- Despliegue tradicional (sin contenedores)**

La máquina o máquinas en las que serán desplegados los componentes de la aplicación requieren:

* Base de datos: Tener instalado MySQL Server 8.
* Backend: Tener instalado JDK 17.
* Frontend: Tener instalado servidor Nginx o Apache.

Se da por hecho que se conoce cómo instalar dichas aplicaciones. Además, cabe mencionar que los hosts deberán estar conectados en red (tener visibilidad a nivel de IP y de puertos) entre ellos. El navegador web o móvil que consume la aplicación deberá poder hacer llamadas tanto al frontend, como al backend.

Para la **base de datos** debe ejecutarse el script dentalium.sql en MySQL Server. Esto puede hacerse con el comando source dentalium.sql . También puede accederse mediante un cliente (como MySQL Workbench) y ejecutarlo desde allí.

Para el **backend** es necesario:

* 1. En el lugar donde se encuentra el JAR (directorio de trabajo), disponer del directorio documents que contendrá:
  + certificate.p12: Certificado digital con el que se emiten las firmas de los documentos (No es imprescindible para el correcto funcionamiento de la aplicación)
  + medical\_studies: Carpeta que contendrá los estudios médicos guardados.
* 2. Tomar el comprimido JAR y levantar la aplicación llamando a la JVM. En este comando, deben declararse las mismas variables de entorno que serían declaradas en el docker-compose resultando algo similar a esto:

$env:DB\_URL="jdbc:mysql://localhost:3306/dentalium"

$env:DB\_USER="root"

$env:DB\_PASSWORD="root"

$env:CERTIFICATE\_PASSWORD="passworddelcertificaado"

$env:JWT\_SECRET="secretodeljwwt"

$env:MAIL\_USERNAME="dentalium.adm@gmail.com"

$env:MAIL\_PASSWORD="passworddelmail"

$env:FRONTEND\_URL="http://localhost"

$env:FRONTEND\_URL\_2="http://otra\_ip"

java -jar back\dentalium\target\dentalium.jar

Se aportan valores válidos en el fichero manual\_deployment.txt

Para el **frontend** se debe tomar la distribución del proyecto de Angular que se ha generado y depositarla en el directorio del servidor destinado a servir el contenido web (Ejs.: Por defecto en Linux para Nginx es /usr/share/nginx/html y para Apache /var/www/html). La distribución está disponible en el directorio **DIST**. Si quiere regenerarse la distribución a partir del código fuente se ejecutaría el comando *ng build -prod* o *npm run build –prod.*

**Antes de eso, si se quiere que la URL del backend sea distinta a la dada por defecto** (<http://localhost:8080/dentalium/api>), **debería parametrizarse** la deseada en el fichero front/dentalium/src/environment/environment.ts . Esto puede ser necesario si, por ejemplo, se desea acceder desde un navegador externo al host anfitrión (dado que ahí no puede accederse a “localhost” se pondría aquí la IP que se encontrase en la interfaz de red de *ipconfig* o *ifconfig*).

**3.- Despliegue mediante IDE**

Se aportan configuraciones que permiten desplegar el proyecto en modo de desarrollo (apunta al fichero de variables de entorno environment.env para cubrir lo descrito anteriormente)

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

# Glosario

CI/CD

SPA

AJAX

REST

SOAP

Diagrama

El contenido generado por IA puede ser incorrecto.

Diagrama, Escala de tiempo

El contenido generado por IA puede ser incorrecto.

Diagrama

El contenido generado por IA puede ser incorrecto.

