Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN ENXEÑARÍA INFORMÁTICA
MENCIÓN EN ENXEÑARÍA DO SOFTWARE

Sistema de gestión de pacientes en ginecología

Estudante: Sergio Cortizo De Dios
Dirección: Carlos Fernández Lozano

A Coruña, xuño de 2021.

Agradecimientos

A mi familia, a mis amigos y a mi tutor que me ha ayudado en el margen de lo posible.

Resumen

Esta aplicación web se ha desarrollado bajo la elaboración de un trabajo final de grado en Ingeniería de Software con duración de varias semanas de trabajo dentro de la Universidade da Coruña (UDC).

Dicha aplicación consiste en un sistema de pacientes para una clínica ginecológica la cual pretende ayudar principalmente en tareas de gestión, control y seguimiento de pacientes, añadiendo funcionalidades como consulta de historial de pacientes, intercambio de mensajes entre usuarios de la app, interconsultas, generación de informes, entre otros.

Las razones que han llevado a la elaboración de este proyecto son debido a la falta de un sistema de gestión de historia clínica, principalmente en clínicas de tamaño pequeño/mediano, abriendo la posibilidad de facilitar todos estos aspectos a través del uso de aplicaciones web que aporten todas estas características.

Abstract

This webapp has been developed within a project of Software Engineering with a length of multiple weeks of work in University of A Coruña (UDC).

This app consists in a patient management system for a ginecology which pretends helping in management, control and patient tracing, adding features like patient history, messaging between users, interconsulting, patient reporting, etc.

The reasons between the project's development are due to the lack of a system that manages clinic history, mainly in small/medium clinics, opening the possibility of ease every of the features previously named through the use of webapps which offers these features.

Palabras clave:

- Ginecología
- Facultativo
- Administrador
- Paciente
- Interconsulta
- Spring MVC
- Hibernate
- Thymeleaf
- JQuery
- Bootstrap
- MySQL
- Dockerización
- Ingeniería del Software
- Metodologías ágiles
- Scrum
- Universidade da Coruña

Keywords:

- Ginecology
- Facultative
- Administrator
- Patient
- Interconsultation
- Spring MVC
- Hibernate
- Thymeleaf
- JQuery
- Bootstrap
- MySQL
- Dockerization
- Software Engineering
- Agile methodologies
- Scrum
- University of A Coruña

Índice general

Índice de figuras

Capítulo 1

Introducción

Esta sección cumple con el fin de resumir en qué consiste la aplicación, explicando cuál ha sido la planificación que se ha realizado para el proyecto, los objetivos a cumplir y las razones que han llevado a realizarlo.

1.1 Motivación

La motivación que ha llevado a realizar este TFG es debido a las siguientes premisas que se han encontrado:

- En clínicas de pequeño y mediano tamaño no existe la posibilidad de disponer de un complejo sistema de gestión de historia clínica.
- En consecuencia, es necesario optar por pequeños desarrollos ágiles que permitan llevar control de los diferentes aspectos de seguimiento médico-clínico de las mismas.
- De acuerdo a la normativa vigente es necesario llevar un seguimiento en cuanto a qué facultativo y en qué momento ha realizado el seguimiento de un paciente, así como un control de quien accede a la información o genera informes de los mismos.
- En este tipo de clínicas suele haber varios especialistas usando la misma aplicación y compartiendo información sobre las pacientes que pasan por la clínica, pudiendo aparte de hacer seguimiento de dichas pacientes, deben poder intercambiar opiniones entre ellos, solicitar interconsultas para determinados casos o mantener un control sobre las pacientes que tienen especial interés para ellos.

Por lo tanto, en base a las premisas indicadas se llega a la propuesta de una aplicación web que siga las pautas requeridas para el seguimiento de pacientes en una consulta médica de ginecología.

1.2 Objetivos

La aplicación web a desarrollar consiste en un sistema de gestión que facilita el control de pacientes dentro de una clínica ginecológica en la cual se busca como objetivos ofrecer lo siguiente:

- Aplicación enfocada en la arquitectura MVC (Modelo-Vista-Controlador) con persistencia en BBDD para control y seguimiento de pacientes.
- Uso de librerías de código abierto apoyadas en APIs Java de más bajo nivel para agilizar su desarrollo, como Spring o Hibernate.
- Gestión de historia clínica e identificación de valoración por facultativo
- Sistema de intercambio de mensajes entre facultativos, tareas comunes y avisos
- Seguimiento de pacientes de interés y últimos pacientes vistos
- Incorporación de historial de informes externos/pruebas complementarias a la paciente
- Gestión de copias de seguridad y restauración de BBDD desde la aplicación
- Generación de informes de seguimiento de las pacientes
- Dockerización de la aplicación + BBDD. Despliegue conjunto con Docker-compose.

A mayores se ha buscado funcionalidades aportadas por otras aplicaciones que cumplen con la misma finalidad que la aplicación que se está a desarrollar, como por ejemplo almacenar medicamentos o especialidades existentes hasta el momento, manejo de agendas para los facultativos y la posibilidad de lanzar avisos para los usuarios registrados dentro del sistema. Una de las soluciones más destacadas como tal es la aplicación FENIX[1], la cual ha sido desarrollada por la empresa SIVSA, caracterizada por ser una empresa reconocida a nivel mundial y se caracteriza principalmente por ofrecer soluciones en el área de TI tanto para centros de salud como administraciones públicas, empresas privadas y soluciones en la nube.

Cabe destacar que también ha sido necesario comprobar qué aspectos normativos podrían afectar para que la aplicación web sea operativa, los cuales son la Agencia Española de Protección de datos (AEPD)[2], Ley Orgánica de Protección de Datos y Garantía de los Derechos Digitales 3/2018[3], Ley de Servicios de la Sociedad de la Información y del Comercio Electrónico 34/2002[4] y el Reglamento General de Protección de datos UE 2016/679[5]. En la siguiente sección se hará una descripción pormenorizada de los requisitos tanto funcionales como no funcionales que se deben cumplir dentro de la aplicación.

1.3 Estructura de la memoria

A continuación, se indica la estructura que se sigue para la estructura de la memoria presente:

- **Capítulo 2. Estado del arte** -> En este capítulo, se indica las aplicaciones que se han tenido en cuenta para obtener aspectos que pueda ofrecer la aplicación web pero que éstas no tengan.
- **Capítulo 3. Tecnologías y herramientas** -> En este capítulo, se indica las tecnologías y herramientas sobre las cuales el alumno se ha apoyado para el desarrollo del TFG.
- **Capítulo 4. Metodología** -> En este capítulo, se indica la metodología que se ha usado para poder gestionar las tareas a seguir dentro del TFG, explicando en qué consiste, conceptos clave, cómo se ha adaptado a las necesidades del TFG y cómo se han organizado las tareas a realizar.
- **Capítulo 5. Diseño y desarrollo de la aplicación** -> En este capítulo, se indica cómo se ha ido repartiendo el trabajo de desarrollo de la aplicación, explicando lo que se ha hecho o cómo se ha dividido los requisitos en tareas más pequeñas.
- **Capítulo 6. Estructura del proyecto** -> En este capítulo, se explica la estructura del proyecto resultante una vez se ha terminado el desarrollo de la aplicación web.
- **Capítulo 7. Conclusiones** -> En este capítulo, se indican las conclusiones a las que se ha llegado una vez se ha terminado el desarrollo del TFG.
- **Anexo A. Historias de usuario detalladas** -> En este anexo, se indica con mayor detalle las historias de usuario que se han obtenido a partir de los requisitos funcionales que se han obtenido en el Sprint 0.
- **Anexo B. Pasos seguidos para el despliegue de la aplicación** -> En este anexo, se indica los pasos que se han realizado para poder desplegar la aplicación dentro del entorno de producción para la aplicación web.
- **Anexo C. Capturas de pantalla de la aplicación** -> En este anexo, se muestran capturas de pantalla de la aplicación en las cuales se muestran aspectos importantes de la aplicación.

1.4 Plan de trabajo

El desarrollo del proyecto se ha dividido en 4 fases principales, las cuales son las siguientes:

1. Análisis de requisitos inicial
2. Desarrollo de la aplicación web
3. Despliegue de la aplicación y base de datos dockerizada en entorno de producción
4. Documentación y elaboración de la memoria de trabajo

Las 3 primeras fases mencionadas anteriormente se han realizado de forma secuencial, haciendo en paralelo la redacción de la memoria para el proyecto conforme se va desarrollando a lo largo de las semanas que dura. De esta forma, nos aseguramos de que tanto la memoria como las características de la aplicación web estén correlacionadas entre si.

Capítulo 2

Estado del arte

En esta sección, se hablará de aplicaciones enfocadas también a lo que son clínicas ginecológicas, indicando cuáles son sus características y qué es lo que ofrecen como tal a sus usuarios.

2.1 FENIX

FENIX es un software de gestión para clínicas que opera en la nube y el cual está diseñado para facultativos y clínicas de tamaño pequeño[1], la cual se ha creado mediante el uso de tecnologías de Microsoft. Esta solución aporta la posibilidad de controlar y agilizar los procesos tanto médicos como administrativos de las clínicas, desde llamadas a pacientes hasta facturación de servicios. Dentro de él se diferencian 2 tipos de usuarios, los cuales son los administrativos encargados de gestionar las citas y los procesos de facturación y los facultativos, pudiendo acceder tanto a las tareas anteriores como a otras exclusivas de su rol.

Entre las características principales que podemos encontrar podemos destacar las siguientes:

- **Gestión Administrativa** -> FENIX ofrece funcionalidades para la gestión de pacientes tales como datos del paciente, las citas asignadas, facturación de las citas, etc.
- **Historia Clínica Electrónica** -> FENIX ofrece para sus usuarios funcionalidades para gestionar el historial clínico de los pacientes registrados en el sistema, destacando aspectos como la solicitud de pruebas, prescripción de recetas, informes, disgnósticos, cuestionarios realizados a los pacientes, etc.
- **Configuración de la clínica** -> FENIX permite gestionar los usuarios que pueden usar la aplicación, aparte de que también permite cambiar las aseguradoras que se pueden asignar, especialidades, medicamentos, plantillas y cuestionarios personalizables, etc.

- **Otros servicios** -> FENIX también ofrece otros servicios para los usuarios como por ejemplo importación/exportación de datos, copias de seguridad automáticas, teleconsultas, copias de seguridad automatizadas, etc.

2.2 GINESALUS

GINESALUS[6] es un software popular el cual está especializado para clínicas, consultas ginecológicas y centros de fertilidad, el cual ha sido desarrollado por QSOFT desde 1995 y permite gestionar de manera integrada las 3 grandes áreas de Gestión:

- **Agendas** -> GINESALUS usa un sistema de agendas visual que imita el estilo de Microsoft Outlook estando configuradas a medida de los usuarios, resúmenes de actividades mensuales, recordatorios para clientes mediante servicios de mensajería como SMS o email, envío de mensajes entre profesionales del centro, etc.

- **Historia clínica** -> GINESALUS ofrece un acceso sencillo y rápido al historial de los pacientes, con una historia clínica creada por médicos especialistas y configurable a medida. Tiene otras características útiles como generación automática de documentos, inserción de dibujos e imágenes, etc.

- **Gestión económica** -> GINESALUS ofrece un sistema de gestión económica de la clínica ginecológica, controlando aspectos como por ejemplo facturación para particulares y mutuas, liquidación de comisiones, control de deudas de pacientes, control de stock y proveedores, etc.

Las razones pruincipales por las cuales se escoge GINESALUS son las siguientes:

1. Está basado en la experiencia que aportan 6.000 clientes del sector salud en 30 países.
2. Está diseñado por ginecólogos y directores de Clínicas Ginecológicas y de Fertilidad.
3. Es posible adaptarlo a las necesidades particualres de cada centro.
4. Aporta beneficios económicos directos, fruto de la eliminación de errores, ahorro de tiempo y optimización de la toma de decisiones.
5. Cumple con la normativa de la LOPD.
6. Es intuitivo y fácil de utilizar.
7. Incluye servicio de mantenimiento, con actualizaciones y soporte técnico continuado.

2.3 Conclusiones

Como conclusión, podemos comprobar que la aplicación web que se está desarrollando se apoya principalmente a través del uso de herramientas de cógido abierto, apoyándose principalmente en proyectos como Bootstrap, JQuery, Spring MVC, Hibernate, Thymeleaf, entre otros, pudiendo convertirse en una ventaja de cara a por ejemplo FENIX, la cual usa herramientas de Microsoft para ser operativa.

En cuanto a funcionalidades ofrecidas por la aplicación web, aparte de ofrecer más o menos las mismas características que las otras dos aplicaciones mencionadas anteriormente, podemos destacar como características principales un mayor enfoque en la comunicación entre usuarios del sistema, añadiendo un sistema de mensajería privada entre usuarios con posibilidad de responder a mensajes recibidos, gestión de interconsultas a partir de entradas de los historiales clínicos de las pacientes para resolver dudas, abrir tareas comunes entre usuarios para escribir mensajes en base a ellas como si se tratase de un chat grupal o de un foro de mensajería y avisos dirigidos a todos los usuarios del sistema.

Otro aspecto a destacar de la aplicación web es que ofrece más de una posibilidad para gestionar copias de seguridad en el sistema, pudiendo generar copias y restaurar la BBDD desde ellas a través de la propia aplicación y en caso de emergencia también se generan copias de seguridad periódicas por si se corre el riesgo de que por alguna razón no se pueda restaurar la BBDD desde la propia aplicación.

Por último, también se puede destacar la posibilidad de añadir múltiples roles para cada uno de los usuarios, pudiendo crear usuarios que pueden por ejemplo ejercer el rol de administrativo, de facultativo o ambos en caso de que sea necesario.

Capítulo 3

Tecnologías y herramientas

En este apartado, se indicará el software que se ha usado a lo largo del TFG para su realización, es decir, desde el sistema operativo hasta el o los lenguajes de programación que se han usado en conjunto con sus librerías y frameworks que se han usado como apoyo para el desarrollo de la aplicación web, incluyendo editores y entorno de virtualización para simular el entorno de producción sobre el cual la aplicación web será operativa.

3.1 Sistema operativo

El sistema operativo que se ha usado es Ubuntu 18.04.5 LTS[7].

Ubuntu es una distribución de Linux basada en Debian, la cual se caracteriza por ser de software libre y de código abierto y por tener la capacidad de correr tanto en computadores de escritorio como en servidores. Se orienta al usuario general enfocándose en la facilidad de uso y en mejorar la experiencia de usuario, estando compuesto por software distribuído bajo una licencia libre o de código abierto.

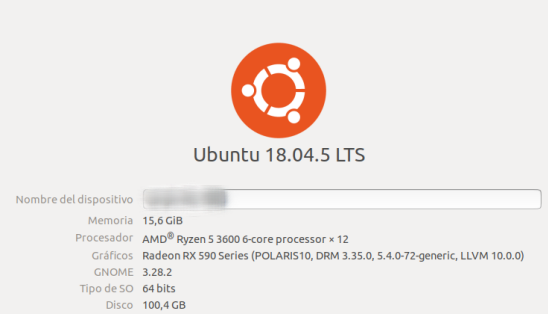


Figura 3.1: Captura de Ubuntu con información sobre hardware usado

3.2 Editor de código

El editor de código que se ha usado para el desarrollo de la aplicación es Eclipse[8], concretamente en su versión 2020-12. Es usado principalmente en proyectos donde se usa como lenguaje de programación Java, aunque también se puede incluír soporte para otros lenguajes de programación a través de plugins proporcionados en el marketplace de Eclipse o pudiendo instalarlos usando la funcionalidad para añadir nuevo software a través de enlaces externos.

El entorno de desarrollo usado por Eclipse se caracteriza por el empleo de plugins para proporcionar toda su funcionalidad, lo cual es una plataforma ligera para componentes de software. Esto no sólo permite añadir soporte para otros lenguajes de programaciópn sino que permite añadir otras funcionalidades como sistemas gestores de BBDD, generación de diagramas, debugging, generación de proyectos mediante el uso de herramientas como por ejemplo Maven, etc.

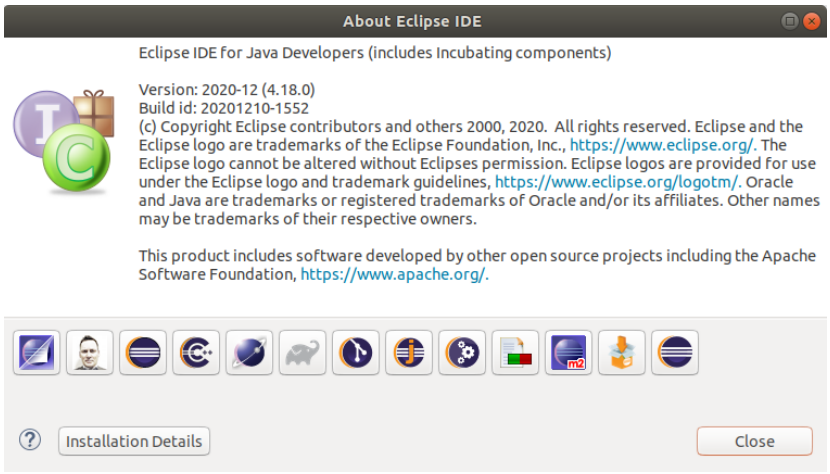


Figura 3.2: Información sobre Eclipse y la versión instalada

3.3 Control de versiones

Git es el software que se ha usado para el control de versiones y cambios realizados en la aplicación.

Git es un software de control de versiones el cual se caracteriza por ser gratuito, de código abierto y estar diseñado para manejar todas las características del proyecto de forma rápida y

eficiente, sea cual sea el tamaño del proyecto[9].

Entre sus características principales se pueden encontrar las siguientes:

1. **”Branching” y ”merging”** -> Es la característica principal que hace que sea diferente con respecto a otros productos de software con la misma finalidad, tales como por ejemplo Subversion. Esto te permite crear distintas ”branches” o versiones derivadas de otras versiones del proyecto, las cuales funcionan independientemente las unas de las otras. Dichas ”branches” se pueden crear, borrar, mezclarse entre ellas o borrarlas en cuestión de segundos. Esto también te permite trabajar sobre el proyecto de múltiples formas:

   - **Cambio de contexto sin fricciones** -> Git te permite crear una rama para probar una idea, actualizarla varias veces, volver a la ”branch” original, actualizarla, volver a la ”branch” en la cual se estaba probando la idea y combinar ambas ramas.
   - **Líneas de código basadas en roles** -> Es posible tener por ejemplo una ”branch” que siempre contenga sólo lo que va a producción, otra en la que se combine el trabajo para realizar pruebas y por último varias más pequeñas para el trabajo que se haga todos los días como correcciones de bugs o añadir nuevas funcionalidades.
   - **”Workflow” basado en funcionalidades** -> Es posible crear nuevas ramas para cada nueva característica en la que esté trabajando para poder alternar sin problemas entre ellas y luego eliminar cada ”branch” cuando esa funcionalidad se fusione en su ”branch” principal.
   - **Experimentación desechable** -> Es posible crear una rama para hacer algún experimento, ver que no va a funcionar y simplemente borrar esa ”branch”, abandonando el trabajo, sin que nadie más la vea (incluso si has actualizado otras ”branches”).

Figura 3.3: Ejemplo ilustrativo de cómo funcionaría el uso de ”branches”

2. **Software ligero y rápido** -> Con Git, casi todas las operaciones se realizan localmente, lo que le aporta ventajas en velocidad en sistemas centralizados que constantemente tienen que comunicarse con un servidor ubicado en otra parte. Está diseñado para trabajar con un Kernel de Linux, lo cual le permite manejar de forma efectiva grandes repositorios desde el 1er día. También se caracteriza por estar desarrollado mediante C, ahorrando costes en ejecución en relación con lenguajes de alto nivel, tales como Java, Python, PHP, C++, etc.

3. **Software distribuído** -> Git es un sistema distribuído, lo cual quiere decir que en vez de hacer un ”checkout” del fragmento actual del código fuente, se está realizando un ”clonado” del repositorio actual. Esto implica que incluso si el equipo está trabajando sobre el mismo repositorio, cada usuario tiene una copia en local del repositorio principal, por lo que en caso de pérdida del servidor o de corrupción de datos, se puede usar alguna de dichas copias para poder restaurar el repositorio a no ser que dado el caso en el que ocurra no haya alguna copia existente de éste. Gracias a ello, junto con la funcionalidad de ”branching” que ofrece, abre muchas posibilidades a la hora de gestionar el flujo de trabajo entre los miembros del equipo, abriendo la posibilidad de trabajar en el repositorio de forma centralizada como por ejemplo en Subversion o haciendo que los cambios realizados por un miembro del equipo pase por un miembro intermedio para que lo supervise antes de subirlo al repositorio entre otros.

4. **Seguridad de datos** -> El modelo de datos usado por Git asegura la integridad en cada parte del proyecto a desarrollar. Cada fichero y commit realizado dentro del repositorio tiene un ”checksum”, con el cual dichos recursos son traídos de vuelta cada vez que se hace un ”checkout”. Esto imposibilita que puedas coger cualquier recurso ubicado fuera del proyecto y que no sea lo que esté subido en él. Tampoco es posible cambiar cualquier aspecto del repositorio en el cual se ubica el proyecto sin antes cambiar los IDs de lo que hay detrás de ellos, es decir, que si tienes un ID asociado a un ”commit”, tienes la seguridad de que el proyecto está en el mismo estado en el cual se ha realizado el ”commit”, sino que nada ha sido cambiado en su historial.

5. **”Staging area”** -> Git tiene una funcionalidad llamada ”staging area”, la cual es un área intermedia donde los ”commits” pueden ser formateados y revisados antes de realizar un ”commit”. Uno de los aspectos que lo diferencia con otras herramientas para gestionar repositorios es que se puede dejar preparados algunos ficheros y hacer un ”commit” de sólo algunos de ellos sin tener que hacer un ”commit” del resto.

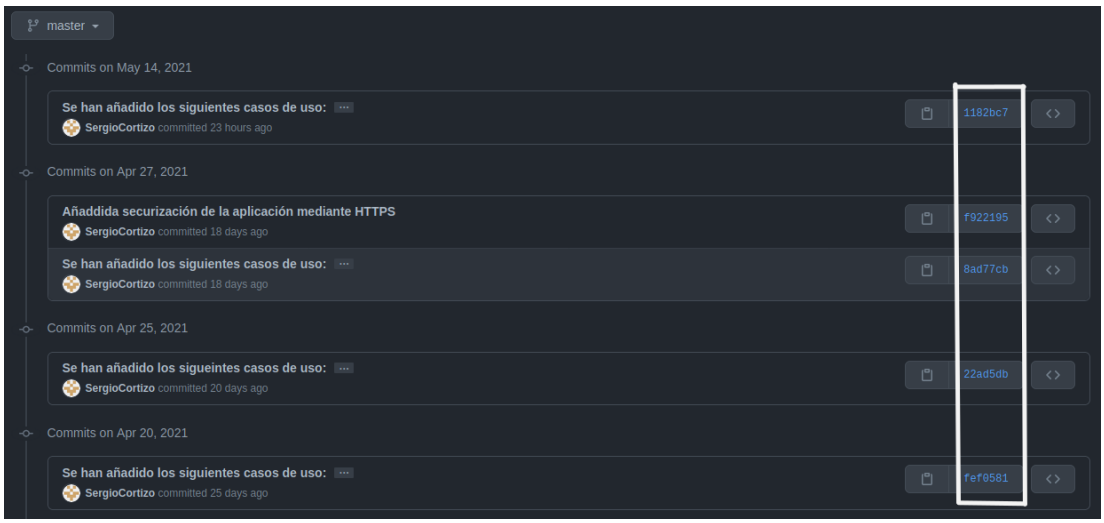


Figura 3.4: Muestra del ID de ”commits” en GitHub

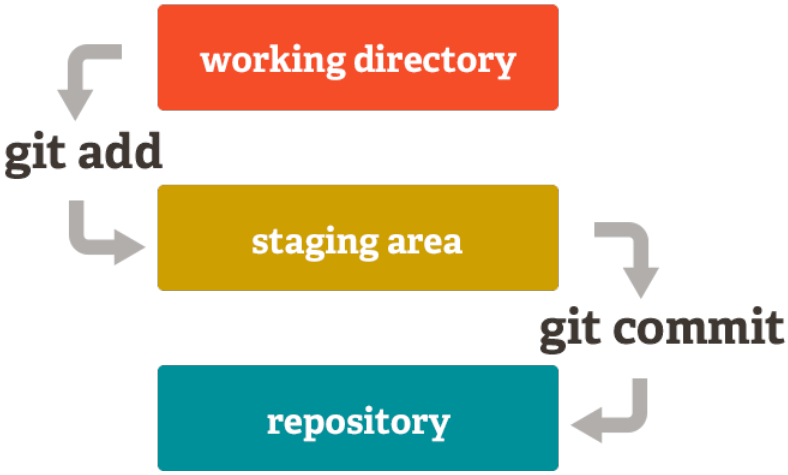


Figura 3.5: Esquema sobre la ”staging area”

6. **Gratis y de código abierto** -> Git está licenciado bajo GNU General Public License version 2.0, la cual es una licencia de código abierto, es decir, que gracias a ella es posible compartir y cambiar software gratuíto para asegurar que dicho software es libre para todos sus usuarios finales.

En este caso, para almacenar los cambios realizados dentro del proyecto en principio se ha creado un repositorio dentro del servidor de Git de la facultad, pero debido a que el año en el cual se ha realizado el TFG es el último en el cual se hace su mantenimiento, se ha tenido que migrar todo el repositorio a otro repositorio privado ubicado dentro de GitHub, conservando toda la información relacionada con él tales como ficheros o ”commits” realizados en el repositorio anterior.

3.4 Java

Java es un lenguaje de programación desarrollado por Sun Microsystems y comercializado desde 1995. Hoy en día es usado por muchas aplicaciones y sitios web, cuyo número va en aumento y está presente en muchos dispositivos electrónicos que se usan a diario, tales como ordenadores, centros de datos, móviles, etc., ya que una de sus características principales es que si es ejecutado en una plataforma, no tiene por qué ser ejecutado en otra[10].

Java es un lenguaje de programación creado con 5 objetivos principales:

1. Debería usar el paradigma de la programación orientada a objetos -> La Programación Orientada a Objetos es un método de programación en el cual se diseña el software de forma que tanto los datos junto con sus operaciones estén agrupados dentro de objetos, vistos como paquetes en lso que se guardan ”comportamiento” y ”estado”, de forma que aplicándolo a proyectos éstos sean más fáciles de gestionar y manejar, mejorando su calidad y reduciendo el número de proyectos fallidos, pudiendo también reutilizar dichos objetos creados para proyectos futuros.

2. Debería permitir la ejecución de un mismo programa en múltiples sistemas operativos -> significa que programas escritos en Java pueden ejecutarse igualmente en cualquier tipo de hardware, por lo que si se escribe una vez un programa con Java éste se puede utilizar en otro hardware. Para ello, primero se compila el código Java para generar lo que se conoce como ”bytecode”, el cual es ejecutado posteriormente por una Java Virtual Machine, que es un programa escrito en código nativo para la plataforma a la que va destinado y es la que interpreta y ejecuta el código. Esta característica hace que sea una alternativa popular en aplicaciones de entorno servidor, como por ejemplo servicios web.

3. Debería incluir por defecto soporte para trabajo en red
4. Debería diseñarse para ejecutar código en sistemas remotos de forma segura
5. Debería ser fácil de usar y tomar lo mejor de otros lenguajes orientados a objetos

Entre los entornos en los cuales se aplica el uso de Java se encuentran los siguientes:

- Dispositivos móviles y sistemas embebidos -> Hay microprocesadores diseñados para ejecutar bytecode Java y software Java, ente los cuales podemos encontrar tarjetas inteligentes, teléfonos móviles, buscapersonas o sintonizadores de TV.
- Navegador web -> Es posible desarrollar pequeñas aplicaciones para incrustarlas en una página HTML para que sean descargadas y ejecutadas por el navegador web, siendo ejecutadas por una Java Virtual Machine acoplada como un plugin dentro del navegador.
- Sistemas de servidor -> donde más se usa actualmente debido al uso de servlets y Java Server Pages, justificando su uso para la implementación de la aplicación web. Esto supuso un avance importante debido a los siguientes factores:
  - El API de programación es muy sencilla, flexible y extensible.
  - Los servlets no son procesos independientes, por lo que se ejecutan dentro del mismo proceso que la JVM, siendo más eficientes en rendimiento, carga computacional y memoria usadas.
  - Las JSP son páginas que se compilan dinámicamente de modo que el código que se consigue supone una ventaja en rendimiento.
- Aplicaciones de escritorio -> Hoy en día existen aplicaciones gráficas de usuario basadas en Java, cuyo entorno de ejecución Java (JRE) se ha convertido en un componente habitual en PCs con alguno de los sistemas operativos más usados del momento. Además, muchas aplicaciones Java lo incluyen dentro del propio paquete de la aplicación de modo que se ejecuten en cualquier PC.

Para el desarrollo de la aplicación mediante el uso de Java, se han usado librerías para agilizar el desarrollo de sus funcionalidades, entre lo que se incluye pruebas para dichas funcionalidades o interacción con la BBDD para leer o escribir datos dentro de ella.

3.4.1 Hibernate

Hibernate es una herramienta de mapeo objeto-relacional para Java y .Net (NHibernate) que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) o anotaciones en los ”beans”

de las entidades que permiten establecer estas relaciones, es decir, las clases que representan las tablas en la BBDD y sus relaciones entre ellas[11].

Con Hibernate, se busca relacionar los 2 modelos de datos dentro de la aplicación, es decir, el modelo usado dentro del servicio web el cual está elaborado usando programación orientada a objetos y el modelo relacional usado dentro de la BBDD. Para ello, se detalla el modelo de datos, as relaciones que existen y que forma tienen, permitiendo a la aplicación manipular los datos de ésta en base a los objetos proporcionados aplicando conversiones de datos entre los tipos del lenguaje utilizado por la aplicación y los tipos en SQL y librando al desarrollador de tener que interactuar con dichos datos a mano, manteniendo la portabilidad entre todos los motores de bases de datos con un ligero incremento en el tiempo de ejecución. También permite formar consultas a BBDD personalizadas mediante el uso de un lenguaje de consulta de datos llamado HQL en conjunto con una API para construir las consultas programáticamente. Dentro de Java se usa en aplicaciones Java independientes o en aplicaciones Java EE mediante el componente Hibernate Annotations que implementa el estándar JPA.

Dentro de la implementación de la aplicación, el uso de dicha herramienta nos permite tener que ahorrarnos el trabajo extra que implica crear operaciones de altas, bajas, modificaciones o consultas sobre cada una de las entidades, las cuales suelen ser siempre las mismas salvo alguna excepción, pudiendo enfocarnos en consultas más complejas[12].

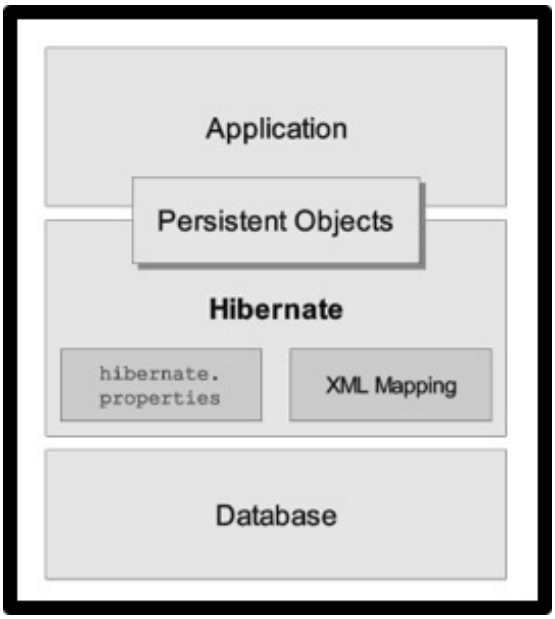


Figura 3.6: Conexión entre la BBDD y la aplicación a través de Hibernate

3.4.2 Spring Framework

Spring es un framework aplicado en el desarrollo de aplicaciones, de código abierto y para la plataforma Java[13]. Dicho framework comprende varios módulos para distintos fines, entre los que se incluyen por ejemplo:

- **Contenedor de inversión de control** -> permite la configuración de los componentes de la aplicación y la administración del ciclo de vida de los objetos hechos con Java a través de inyección de dependencias.
- **Programación orientada a aspectos** -> habilitando la implementación de rutinas transversales. Particualrmente, se usa dentro de la aplicación web para la implementación de logging dentro de la capa de servicios, pudiendo ver por ejemplo cuándo se ha hecho determinada acción o cuándo se ha lanzado determinada excepción.
- **Acceso a datos** -> se trabaja con RDBMS en la plataforma java, usando JDBC y herramientas de Mapeo objeto relacional.
- **Gestión de transacciones** -> unifica distintas APIs de gestión y coordina las transacciones para los objetos Java durante la ejecución de la aplicación.
- **Modelo vista controlador (MVC)** -> Un framework basado en HTTP y servlets, que provee herramientas para la extensión y personalización de aplicaciones web y servicios web REST. Dentro de la aplicación web se usa en conjunto con Thymeleaf para el renderizado de plantillas en HTML, del cual se hablará más tarde.
- **Procesamiento por lotes** -> un framework para procesamiento de mucho volumen que como características incluye funciones de registro/trazado, manejo de transacciones, estadísticas de procesamiento de tareas, reinicio de tareas, y manejo de recursos.
- **Autenticación y Autorización** -> procesos de seguridad configurables que soportan un rango de estándares, protocolos, herramientas y prácticas a través del subproyecto Spring Security.
- **Testing** -> Soporte de clases para desarrollo de unidades de prueba e integración.

Para la aplicación web, no es necesario utilizar todas las característocas que aporta, sólo las que se consideren necesarias para que pueda cubrir los servicios que ofrece, como por ejemplo MVC, Autenticación y autorización a recursos o el acceso a los datos. A continuación se indicará las características que se han usado por parte de Spring, las cuales se acoplan a través de dependencias en el proyecto usando Maven.

3.4.3 Maven

Maven es una herramienta de software para la gestión y construcción de proyectos desarrollados en Java cuyo modelo de construcción está basado en XML[14].

Para el proyecto, se usa un fichero en formato XML denominado POM (Project Object Model)[15], dentro del cual se indica toda la información relacionada con el proyecto, como la descripción del proyecto, sus dependencias con otros módulos, el orden de costrucción de los elementos, etc. De base ya viene con comandos definidos y que se consideran comunes, tales como instalación, configuración o empaquetado de la aplicación. Maven se caracteriza por estar diseñado para poder usarse en red, es decir, que puede descargar plugins de diversos repositorios, sea del repositorio oficial de Maven, de otras organizaciones o de la propia organización en la cual se esté realizando el proyecto.

Las características principales de Maven para el uso en proyectos Java son las siguientes:

- **Convención sobre configuración** -> Maven sigue el principio de Convención sobre configuración, el cual es un paradigma de programación de software que busca minimizar el número de decisiones a tomar, ganando simplicidad sin perder flexibilidad en el proceso y pudiendo utilizar modelos existentes en el desarrollo de software.
- **Reutilización** -> Maven está construido alrededor de la idea de reutilización, en particular aplicado a la lógica de construcción. La principal idea es no reutilizar el código sino simplemente cambiar la configuración.
- **Ciclo de vida** -> Las partes del ciclo de vida principal del proyecto Maven son:
  1. compile -> Genera los ficheros .class compilando los fuentes .java
  2. test -> Ejecuta los test automáticos de JUnit existentes, abortando el proceso si alguno de ellos falla.
  3. package -> Genera el fichero .jar con los .class compilados
  4. install -> Copia el fichero .jar a un directorio de nuestro ordenador donde maven deja todos los .jar. De esta forma esos .jar pueden utilizarse en otros proyectos maven en el mismo ordenador.
  5. deploy -> Copia el fichero .jar a un servidor remoto, poniéndolo disponible para cualquier proyecto maven con acceso a ese servidor remoto.

  A la hora de ejecutar alguno de estos comandos, primero se ejecuta de forma secuencial los comandos anteriores al indicado antes de ejecutar dicho comando. Es decir, si por

ejemplo en un terminal ejecutamos ”mvn install”, el orden secuencial sería compile > test > package > install. Fuera del ciclo de vida, existen más comandos definidos pero no están dentro de él debido a que Maven no los considera como parte del ciclo de vida por defecto. Dichos comandos se pueden añadir modificando el ciclo de vida a través del POM. Algunos ejemplos de ellos son:

  - clean -> Elimina todos los .class y .jar generados. Después de este comando se puede comenzar un compilado desde cero.
  - assembly -> Genera un fichero .zip con todo lo necesario para instalar nuestro programa java. Se debe configurar previamente en un fichero xml qué se debe incluir en ese zip.
  - site -> Genera un sitio web con la información de nuestro proyecto. Dicha información debe escribirse en el fichero pom.xml y ficheros .apt separados.
  - site-deploy -> Sube el sitio web al servidor que hayamos configurado.

- **POM** -> El POM es la unidad base de trabajo en Maven. Tal y como se ha mencionado anteriormente, es un documento en XML que contiene información sobre el proyecto y configuraciones para poder montarlo, proporcionando de base valores por defecto para la mayoría de los proyectos que lo usan. A la hora de ejecutar un comanodo en Maven, primero se comprueba que está el POM, se lee la información almacenada dentro de él y se ejecuta el comando. A continuación, se muestra un ejemplo de cómo se vería el contenido del POM:

```
<project>
  <modelVersion>4.0.0</modelVersion>

  <parent>
    <groupId>com.mycompany.app</groupId>
    <artifactId>my-app</artifactId>
    <version>1</version>
    <relativePath>../parent/pom.xml</relativePath>
  </parent>

  <artifactId>my-module</artifactId>

  <properties>
    <mavenVersion>3.0</mavenVersion>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.apache.maven</groupId>
```

```
20          <artifactId>maven-artifact</artifactId>
21          <version>${mavenVersion}</version>
22        </dependency>
23        <dependency>
24          <groupId>org.apache.maven</groupId>
25          <artifactId>maven-core</artifactId>
26          <version>${mavenVersion}</version>
27        </dependency>
28      </dependencies>
29    </project>
30
```

- **Integración con IDE** -> Maven es un sustituto del IDE que se esté usando en ese momento, por lo que se debe integrar con ellos. Es posible hacerlo mediante el uso de plugins para la configuración de archivos del IDE a partir del uso de POMs, pudiendo aplicarse a IDEs como Eclipse, IntelliJ o Netbeans. En este caso, se ha incluído un plugin con la configuración necesaria para su integración con Eclipse, el cual es tal y como se ha indicado con anterioridad el IDE seleccionado para el desarrollo de la aplicación web.

Dentro del POM proporcionado para la construcción del proyecto, se han indicado un conjunto de dependencias y plugins. Las dependencias que se han añadido son las siguientes:

- **spring-boot-starter-test**[16] -> Característica de Spring Framework para el testeo de aplicaciones desarrolladas en Spring mediante el uso de JUnit Jupiter, Hamcrest y Mockito.
- **spring-boot-starter-data-jpa**[17] -> Característica de Spring Framework que permite el uso del framework en conjunto con Hibernate para al interacción con la BBDD.
- **spring-boot-starter-web**[18] -> Característica de Spring Framework que ayuda en el desarrollo de aplicaciones web a través del uso de Spring MVC.
- **spring-boot-starter-thymeleaf**[19] -> Característica de Spring Framework que ayuda en el desarrollo de aplicaciones web siguiendo el patrón Modelo-Vista-Controlador mediante el uso de Thymeleaf.
- **spring-boot-devtools**[20] -> Herramienta de Spring que nos permite reiniciar de forma automática la aplicación web mientras estamos añadiendo cambios sobre ella en el entorno de desarrollo.
- **thymeleaf-extras-springsecurity5**[21] -> Módulo que forma parte del entorno de Thymemeaf Extras el cual nos permite añadir características de securización a las plantillas HTML que usen Thymeleaf para el renderizado de contenido sobre ellas. No es

parte de la base de Thymeleaf, pero tiene soporte completo por parte de su equipo de desarrollo.

- **mysql-connector-java**[22] -> Controlador de JDBC que posibilita la conexión entre la BBDD y la aplicación web desarrollada con Java.
- **spring-security-crypto**[23] -> Módulo de Spring proporcionado para la encriptación simétrica, generación de claves y encriptado de contraseñas. Dentro de la aplicación, se usa para el encriptado de las contraseñas para los usuarios de la aplicación web que se vayan añadiendo.
- **spring-security-config**[24] -> Característica de Spring Framework que sirve de ayuda en la configuración de aspectos de seguridad dentro de la aplicación web.
- **spring-security-web**[25] -> Característica de Spring Framework que ayuda en aspectos de la aplicación web tales como autenticación y autorización, aportando integración de forma opcional con Spring MVC.
- **spring-boot-configuration-processor**[26] -> Procesador de anotaciones que genera metadatos sobre las clases dentro de la aplicación que estén anotadas con @ConfigurationProperties, los cuales son usados por el IDE para completar y documentar automáticamente propiedades definidas en ficheros de configuración como por ejemplo ”application.properties”.
- **validation-api**[27] -> Módulo que añade la validación de Java Beans dentro de la aplicación mediante el uso de anotaciones.
- **tika-core**[28] -> Módulo que proporciona lo necesario para la detección y extracción de metadatos sobre ficheros en diferentes formatos como PDF o TXT. Dentro de la aplicación se usa por ejemplo para poder manejar el cambio de logotipo por parte del administrador o administradores.
- **commons-codec**[29] -> Módulo que contiene codificación/decodificación de caracteres para varios formatos, entre los que se incluyen el formato hexadecimal o Base64.
- **flying-saucer-pdf**[30] -> M´odulo que permite el renderizado de archivos en formato PDF mediante el uso de XML o XHTML en conjunto con CSS 2.1. En este caso se usa para la generación de informes de distinta índole dentro de la aplicación.
- **lombok**[31] -> Librería para Java que permite reducir la escritura de código mediante el uso de anotaciones en Java, permitiendo ahorrar tiempo y mejorando la legibilidad del mismo.

- **spring-boot-starter-log4j2**[32] -> Característica de Spring Framework que permite el uso de Log4j2 para la funcionalidad de logging dentro de la aplicación web. En este caso, se usa para la edición de la tabla de registros dentro de la BBDD y del fichero de texto en donde se guardan los logs de la aplicación en caso de ser necesario consultarlo.
- **spring-boot-starter-aop**[33] -> Característica de Spring Framework que permite el uso de la programación orientada a aspectos a través de Spring AOP y AspectJ. Se usa en este caso para el uso de logging dentro de la capa de servicios de la aplicación web.
- **mysql-backup4j**[34] -> Librería de Java que permite la generación y el uso de backups desde la aplicación web, generando un fichero en formato sql que es comprimido en un fichero en formato .zip para ser proporcionado a través de email, Google Drive o cualquier otro medio de preferencia.

En cuanto a plugins usados dentro del proyecto, se han incluído los siguientes:

- **spring-boot-maven-plugin**[35] -> Plugin que proporciona soporte para Spring Boot dentro del proyecto Maven, pudiendo empaquetar archivos ejecutables en .jar o en .war, ejecutar aplicaciones desarrolladas en Spring boot, generar información de compilación y ejecutar la aplicación hecha en Spring Boot antes de inicializar las pruebas de integración.
- **sql-maven-plugin**[36] -> Plugin que permite integrar SQL dentro del proyecto Maven, permitiendo ejecutar sentencias en SQL sean cadenas de texto, una lista o un conjunto de ficheros. Dentro del proyecto se usa para crear una BBDD que se usa dentro del entorno de desarrollo y otra para realizar las pruebas.

3.5 MySQL

MySQL[37] es un sistema gestor de BBDD que se caracteriza por ser de código abierto y el cual es desarrollado, distribuído y soportado por Oracle. Se usa dentro del proyecto para proporcionar a la aplicación web un almacenamiento de los datos que debe manejar, eligiéndolo como gestor de BBDD de preferencia por diversas razones que se adaptan a las necesidades particualres del proyecto, entre las cuales se encuentran las siguientes:

- **BBDD Relacional** -> Una BBDD Relacional es una BBDD en la cual se clasifican los datos en tablas, pudiendo establecer reglas y relaciones entre ellas usando por ejemplo claves primarias, claves foráneas, columnas con datos únicos, relaciones 1:1, 1:N, M:N entre otros tipos de restricciones, permitiendo un buen diseño dentro de las tablas de la BBDD y en consecuencia se solventan problemas como inconsistencias, datos duplicados, que están obsoletos, etc., justificando su uso como algo esencial para un sistema

en el cual es fundamental que haya una consistencia entre los datos que se estén manejando.

- **MySQL es de código abierto** -> Esto quiere decir que cualquiera puede usar y modificar el software que se ha proporcionado, es decir, que puede ser descargado por cualquiera y ser usado gratuitamente, pudiendo incluso modificar el código fuente en caso de tener que adaptarlo a necesidades particulares.

- **MySQL es rápido, escalable, fiable y accesible** -> MySQL puede funcionar perfectamente en un PC de sobremesa o portátil en conjunto con otras aplicaciones, servidores web, etc. con poca o nula atención. En caso de dedicar una máquina para la BBDD, es posible aplicar ajustes para aprovechar memoria, procesador y capacidad de E/S disponible, pudiendo también usarse en conjunto con máquinas o clústeres interconectadas entre si mediante red.

- **MySQL trabaja en entornos cliente/servidor o sistemas embebidos** -> Es un sistema cliente/servidor formado por un servidor SQL multihilo que ofrece soporte para diferentes entornos ”backend”, entornos cliente y librerías, herramientas administrativas, APIs, etc.

- **MySQL dispone de software desarrollado por la comunidad** -> dispone de funcionalidades desarrolladas por otros usuarios del sistema, por lo que es posible que el lenguaje de programación o aplicación que se esté usando en el momento tengoa soporte para MySQL.

Las características principales que ofrece MySQL para su funcionamiento son las siguientes:

- **”Internals” y portabilidad** -> MySQL es un sistema escrito en C/C++, el cual ha sido probado con distintos compiladores y optimizado para trabajo mediante multihilo y/o consultas que implica recuperar datos de varias tablas, como por ejemplo consultas que implican el uso e JOIN.

- **Tipos de datos** -> desde números hasta cadenas de caracteres de tamaño fijo o variable.

- **Declaraciones y funciones** -> MySQL soporta distintos tipos de consultas, sean de selección de columnas, filtrado, agrupamiento de datos, operaciones sobre columnas, inserciones, actualizaciones, borrado, etc.

- **Seguridad** -> MySQL tiene un sistema de seguridad basado en privilegios y contraseñas el cual es flexible y seguro, permitiendo verificación basada en ”host”. Las contraseñas

están seguras mediante encriptación del tráfico sobre ellas a la hora de conectarse a un servidor.

- **Escalabilidad y límites** -> MySQL está diseñado para soportar BBDD grandes, con múltiples tablas en las que se guardan miles de registros a diario y con soporte para añadir hasta 64 índices por tabla.

- **Conectividad** -> Los clientes se pueden conectar a MySQL a través de distintos protocolos de red y/o lenguajes de programación usando librerías, APIs o conectores pensados para ello.

- **Localización** -> MySQL tiene soporte para distintos idiomas al original y soporte completo para diferentes conjuntos de caracteres, tales como latin1 o UTF-8.

- **Clientes y herramientas** -> MySQL tiene distintas herramientas y utilidades para fines como copias de seguridad, optimización de datos, administración o reparación.

3.6 Bootstrap

Bootstrap[38] es una biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web enfocada en la parte cliente de la aplicación, enfocándose en concreto en componentes que se usan con frecuencia por parte del lado cliente, incluyendo básicamente plantillas de diseño para elementos HTML y CSS, tales como botones, tablas, enlaces, text, formularios, listas, menús desplegables, etc.

Bootstrap es un proyecto mantenido por un pequeño equipo de desarrollo en GitHub, siendo originalmente creado por un par de empleados en Twitter, siendo desde entonces uno de los frameworks para ”frontend” y de código abierto más populares del momento.

Dentro del proyecto, se usa Bootstrap básicamente para estilizar todos los elementos de la parte cliente de la aplicación web en conjunto con Thymeleaf, el cual como se ha mencionado anteriormente se encarge de renderizar el contenido de las plantillas mediante el uso de HTML, aplicándose tanto en tablas como en formularios, botones, enlaces, menús desplegables, etc.

3.7 JQuery

JQuery[39] es una biblioteca multiplataforma de JavaScript que permite simplificar tareas como la funcionalidad de las plantillas HTML, manejo de eventos en la aplicación, animaciones o interacción mediante el uso de AJAX, permitiendo obtener los mismos resultados que usando código en JavaScript nativo pero ahorrando tiempo y esfuerzo.

JQuery es software libre y de código abierto con doble licencia bajo la Licencia MIT y la Licencia GNU v2, permitiendo su uso en proyectos libres y privados. También proporciona medios para el desarrollo de plugins, pudiendo crear abstracciones para interacción con elementos y animaciones de bajo nivel, efectos avanzados, entre otros.

JQuery es una herramienta para poder manipular el DOM (Data Object Model) de forma sencilla. Dentro de una plantilla HTML, el DOM representa en forma de árbol todos los elementos que forman la plantilla, pudiendo por ejemplo seleccionar un grupo de elementos con características en común para cambiar uno o varios de sus atributos o hacer que respondan ante eventos recibitos por el navegador, como por ejemplu un clic, la presión de una tecla, etc.

Los principios de desarrollo con JQuery son los siguientes:

- **Separación entre JS y HTML** -> JQuery proporciona una sintaxis simple para añadir elementos que manipulen el DOM usando JavaScript en vez de tener que añadir eventos a atributos HTML para llamar a atributos de JavaScript, facilitando el poder separar el código en JavaScript del código realizado en HTML.
- **Brevedad y claridad** -> JQuery promueve la brevedad y la claridad del código a desarrollar.
- **Eliminación de incompatibilidades entre navegadores** -> LOs motores de JavaScript fincionan de distinta forma entre si, por lo que en el caso de desarrollar funcionalidades para plantillas HTML en JavaScript nativo habría que adaptarlo de forma que se adapte al motor de JavaScript para cada uno de ellos en el margen de lo posible.
- **Extensibilidad** -> Los nuevos eventos, elementos y métodos pueden agregarse fácilmente y luego reutilizarse como si fueran un plugin en su conjunto.

Dentro de la aplicación, se usa JQuery para añadir interacción a alas plantillas HTML generadas mediante Thymeleaf sin tener que pensar en la compatibilidad con otros navegadores

web ni la forma de manejar animaciones a través de JavaScript ya que están definidos por el propio JQuery, usando en conjunto otros plugins desarrollados por la comunidad para aspectos como por ejemplo la validación de datos en el cliente mediante el uso de formularios o listas dinámicas ya montadas. Dichos plugins se detallarán más adelante.

Las características principales de JQuery son las siguientes:

- Selección de elementos DOM utilizando el motor de selección de código abierto de múltiples navegadores Sizzle, un motor de selección de elementos en una plantilla HTML a partir de atributos en común de la forma más óptima posible.
- Interactividad y modificaciones del árbol DOM.
- Eventos.
- Manipulación de la hoja de estilos CSS.
- Efectos y animaciones.
- Animaciones personalizadas.
- AJAX.
- Procesamiento asíncrono.
- Soporte para el uso de plugins.
- Utilidades varias como obtener información del navegador, operar con objetos y vectores, funciones para rutinas comunes, etc.
- Soporte para navegadores.

Los plugins que se han utilizado en conjunto con JQuery dentro de la aplicación son los siguientes:

- **Bootstrap Multiselect**[40] -> Plugin basado en JQuery que proporciona una interfaz de usuario intuitiva para usar un selector múltiple, el cual es un selector con ”checkboxes” en forma de menú desplegable.
- **JQuery Ensure Max Length**[41] -> Plugin modificable que sirve para limitar el número de caracteres usados dentro de entradas de texto.

- **jQuery Validation Plugin**[42] -> Plugin para facilitar la validación de formularios del lado del cliente ofreciendo múltiples opciones de edición de validación. De base ya viene con distintos tipos de validación mientras se provee una API para crear métodos propios de validación, incluyendo mensajes de error con el ingles como idioma por defecto y soporte para hasta 37 idiomas. Dentgro de la aplicación se usa para validar campos como números de teléfono, DNIs o números de historia del SERGAS, añadiendo métodos propios de validación conforme sea necesario.

- **DataTables**[43] -> Plugin que facilita añadir interacción a tablas HTML mediante el uso de JQuery, añadiendo opciones de personalización, paginación, criterios de búsqueda, internacionalización, orden de datos usando múltiples columnas, etc. bootstrap-datepicker

- **Bootstrap Datepicker**[44] -> Plugin que permite añadir campos de entrada de fechas usando Bootstrap para el estilizado.

- **jquery-hunterTimePicker**[45] -> Plugin que permite habilitar un selector de hora en formato de 24 horas para cualquier campo de entrada que se elija.

3.8 TinyMCE

TinyMCE[46] es una librería en JavaScript que permite añadir un procesador de texto dentro de una plantilla HTML, funcionando como un campo de entrada de texto en el cual dicho contenido se puede editar como si fuera un documento de texto en aplicaciones como Microsoft Word o LibreOffice, asociándolo al acrónimo de WYSIWYG.

WYSIWYG es un acrónimo el cual quiere decir que **W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et, lo cual traducido al español quiere decir que lo que ves es lo que obtienes. Es una frase que se aplica a editores de texto con formato que permiten escribir un documento mostrando directamente el resultado final. En el caso de editores de HTML, este concepto se aplica a los que permiten escribir la página sobre una vista preliminar similar a la de un procesador de textos, ocupándose en este caso el programa de generar el código fuente en HTML. Por lo tanto, esto permite añadir contenido sobre una parte en concreto de la aplicación web, pudiendo añadir listas, subrayar texto o ponerlo en negrita, añadir tablas o colores, etc.

En el caso de la aplicación web, se usa para funcionalidades como por ejemplo comentarios para una entrada del curso clínico de una paciente en concreto, aclaraciones a la hora de

prescribir una receta o escribir mensajes de cualquier índole dentro de la aplicación, como por ejemplo mensajes privados, interconsultas o tareas comunes.

3.9 Docker

Docker[47] es un software de Tecnologías de la Información de creación de contenedores que permite la creación y el uso de componentes de Linux para distintos fines que se pueden conectar entre si para ofrecer servicios en su conjunto, pudiendo instalar contenedores para dar funciones de BBDD, servicios, web, copias de seguridad, etc., pudiendo usarlos como si fueran contenedores de máquinas virtuales caracterizadas por ser livianas y siendo flexibles en su uso, pudiendo desde copiarlas, crearlas, modificarlas, borrarlas, moverlas de un servidor a otro en caso de necesidad, etc.

Docker funciona usando el kernel de Linux en conjunto con éste para la segregación de los procesos, de modo que éstos puedan ejecutarse de manera independiente. Los contenedores que usa Docker se han diseñado con el fin de poder cumplir esa independencia, es decir, que se pueda ejecutar varios procesos y aplicaciones por separado para poder aprovechar mejor la infraestructura y aportando la seguridad que se obtiene al separarlos.

Dichos contenedores se implementan dentro de Docker siguiendo un modelo basado en imágenes, haciendo posible compartir una aplicación con todas sus dependencias en varios entornos, automatizando también en el proceso la implementación de dichas aplicaciones en los contenedores que sea necesario crear.

Gracias a ello, Docker ofrece las siguientes ventajas a la hora de aplicar su modelo basado en imágenes y contenedores:

- **Modularidad** -> Docker se enfoca en la capacidad de tomar una parte de una aplicación, para actualizarla o repararla, sin necesidad de tomar la aplicación completa.
- **Control de versiones de imágenes y capas** -> Cada archivo de imagen de Docker se compone de una serie de capas, combinándose en una sola imagen. Una capa se crea cuando la imagen cambia. Cada vez que un usuario especifica un comando, como ejecutar o copiar, se crea una nueva capa. Docker reutiliza estas capas para construir nuevos contenedores, agilizando de esta forma el proceso de construcción de éstas. Los cambios intermedios se comparten entre imágenes, mejorando aún más la velocidad, el tamaño y la eficiencia. El control de versiones es inherente a la creación de capas, por lo

que cada vez que se produce un cambio nuevo éste se guarda en un registro de cambios incorporado.

- **Restauración** -> Docker permite restaurar imágenes a una versión anterior si se da el caso de que la versión que se esté usando actualmente no esté acorde a las necesidades particulares del proyecto, hacíendolo compatible con un enfoque al desarrollo ágil y permite aplicar Integración Contínua y Despliegue Contínuo desde el punto de vista de las herramientas.

- **Implementación rápida** -> Es posible reducir el tiempo de implementación de las aplicaciones web a través de Docker a pocos segundos. Esto es debido a que los contenedores pueden compartir rápidamente los procesos similares con nuevas aplicaciones, aparte de que un SO en Docker no necesita iniciarse para agregar o mover un contenedor, reduciendo los tiempos de implementación, por lo que es posible crear y destuír la información de los contenedores sin preocupaciones, de forma fácil y rentable.

Para el proyecto, se han creado 3 contenedores para que la aplicación web sea operativa, los cuales son los siguientes:

1. **Aplicación web** -> Contenedor que representa la aplicación web encargada de ofrecer los servicios a los clientes. Para crear el contenedor, se usa la imagen oficial de Maven, la cual se puede encontrar dentro de Docker Hub[48].

2. **BBDD** -> Contenedor encargado de guardar y mandar los datos que vaya dando o pidiendo la aplicación web. Dicho contenedor usa un volumen ubicado dentro de la máquina que se encarga de ejecutar los contenedores de Docker, ya que cada vez que se reinicia o se apaga, a no ser que se le indique una ubicación para guardar por ejemplo los registros de la BBDD, dichos datos no persisten, desapareciendo en el proceso. Para crear el contenedor se usa la imagen oficial de mysql, la cual se puede encontrar en Docker Hub[49].

3. **Copias de seguridad** -> Contenedor encargado de generar copias de seguridad de forma periódica y restauración de la BBDD a partir de una de ellas, usando un volumen para guardar las copias de seguridad por las mismas razones que el contenedor de BBDD. Este contenedor existe para cubrir el caso en el cual no sea posible restaurar la BBDD a partir de una copia de seguridad generada desde la aplicación pero que al menos exista la posibilidad de recuperarla usando dicho contenedor. Para crear el contenedor se usa una imagen de Docker llamada ”mysql-cron-backup”, la cual usa ”mysqldump” para realizar copias de seguridad de forma periódica usando crontab. Dicha imagen también se puede encontrar en Docker Hub[50].

Cabe destacar que para tanto el contenedor de la aplicación web como de la BBDD se ha creado un Dockerfile para la creación de éstos, en concreto para configurarlas de modo que tengan la codificación de caracteres en UTF-8. Esto es debido a que las imágenes que se están usando de base no disponen de codificación de caracteres en UTF-8, lo cual puede complicar casos como leer datos desde la aplicación que contengan caracteres especiales que se usan habitualmente en el español, como por ejemplo las tildes.

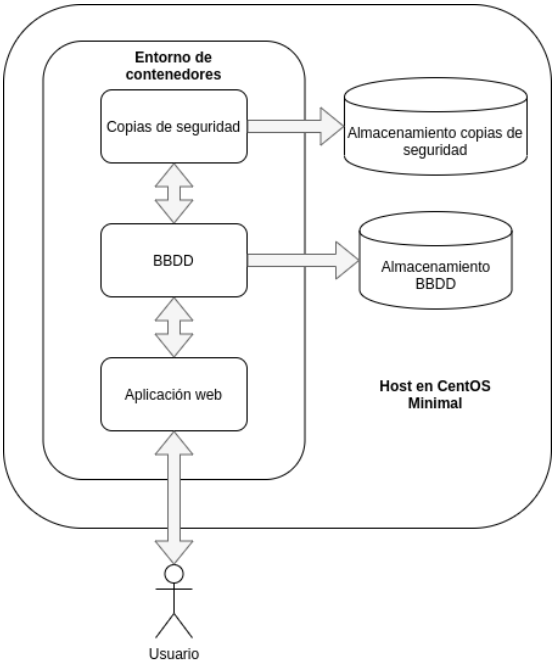


Figura 3.7: Diagrama sobre los contenedores implementados

Para el despliegue conjunto de los contenedores, se ha usado docker-compose[51], la cual es una herramienta que permite definir aplicaciones que se caracterizan por estar compuestas por múltiples contenedores de Docker con distintos fines, usando un fichero en formato YAML para configurar los servicios y pudiendo inicializarlos a la vez usando solo un comando. Es posible usar docker-compose para distintos casos de uso, como por ejemplo entornos de desarrollo, de producción, de pruebas, etc., usando comandos para iniciar/parar los contenedores, comprobar el estado de los contenedores, entre otros.

Las características principales de docker-compose son las sigueintes:

- **Múltiples entornos aislados en un simple host** -> Docker-compose usa nombres de proyecto para aislar los entornos entre si y pudiendo aplicarse a distintos contextos.
- **Persistencia de datos al crear contenedores** -> Docker-compose guarda toda la in-

formación usada por los contenedores, siempre y cuando éstos se guarden dento de algún almacenamiento asociado con el host real encargado de ejecutarlos. En caso de que haya datos existentes de ejecuciones anteriores, dichos datos se copian a los contenedores a los cuales van asociados, asegurando que esos datos no se pierden en caso de caída.

- **Sólo se recrean contenedores con cambios importantes** -> Docker-compose cachea las configuraciones de los contenedores, de modo que cuando se vuelve a arrancar un contenedor en el cual su configuración no ha cambiado, se reutiliza dicho contenedor, de forma que se puede aplicar cambios muy rápido.

- **Variables de entorno y traslado de composiciones de contenedores entre entornos** -> Docker-compose tiene soporte para usar variables de entorno dentro del fichero en YAML, pudiendo configurarlo para ser usado para distintos entornos o usuarios.

Un ejemplo de cómo sería la estructura de un fichero YAML en docker-compose sería el siguiente:

```
version: "3.9"
services:
  web:
    build: .
    ports:
      - "5000:5000"
    volumes:
      - .:/code
      - logvolume01:/var/log
    links:
      - redis
  redis:
    image: redis
volumes:
  logvolume01: {}
```

3.10 Oracle VM VirtualBox

VirtualBox[52] es una aplicación multiplataforma enfocada en la virtualización de sistemas operativos, permitiendo ejecutar múltiples máquinas dentro de un PC con distintos sistemas operativos al mismo tiempo. Esto quiere decir que si por ejemplo tienes un ordenador con Windows 10 como sistema operativo, es posible tener otras máquinas ejecutándose a la vez con por ejemplo otra versión de Windows o bien con distintas distribuciones de Linux, como por ejemplo Ubuntu, Fedora, Debian, CentOS, etc., pudiendo instalar las máquinas virtuales

que se quiera siempre y cuando haya sufieciente espacio en la unidad de almacenamiento que se esté usando.

La virtualización es una solución que se puede usar para muchos casos, entre los cuales están los siguientes:

- **Se necesita arrancar múltiples sistemas operativos a la vez** -> Con la virtualización es posible correr varios sistemas operativos a la vez, pudiendo cambiar de un sistema operativo a otro sin tener que reiniciar el ordenador real.

- **Instalaciones de software más sencillas** -> Es posible usar máquinas virtuales para exportar configuraciones de por ejemplo servidores, es decir, que si por ejemplo se llega a dar el caso de tener que instalar un servidor LDAP o un servidor de correo, en vez de tener que instalarlo en una máquina real con todo el proceso que conlleva, se crea una máquina virtual con todo lo necesario y se instala en el entorno sobre el cual vaya a operar.

- **Entorno de pruebas y recuperación ante desastres** -> La máquina virtual junto con sus unidades de almacenamiento se pueden considerar como un contenedor que puede ser congelado, copiado, crear copias de seguridad a partir de él, etc.

- **Ventajas en infraestructura** -> La virtualización permite ahorrar costes en infraestructura, ya que gracias a ello se reducen los costes en dinero y electricidad que ello conlleva, pudiendo usar una única máquina que permita correr las máquinas virtuales necesarias en vez de tener múltiples máquinas reales cumpliendo sólo un propósito dentro del entorno de producción.

Las características principales que ofrece VirtualBox como herramienta de virtualización son las siguientes:

- **Portabilidad** -> VirtualBox funciona en múltiples sistemas de 64 bits, funcionando como un ”hipervisor de Tipo 2”, es decir, que a diferencia de los hipervisores de Tipo 1 los cuales funcionan directamente sobre el hardware, los hipervisores de tipo 2 como VirtualBox necesitan un sistema operativo compatible para que puedan funcionar pero operando de la misma forma en los sistemas operativos compatibles con el mismo formato para los ficheros de máquiinas virtuales.

- **Oracle VM VirtualBox Guest Additions** -> Es un conjunto de paquetes de software que se pueden instalar en sistemas anfitrión para mejorar el rendimiento e integración y comunicación con el sistema real.

- **Soporte para hardware** -> VirtualBox soporta características como multiprocesamiento, soporte para unidades USB, resolución multipantalla, compatibilidad con tarjetas de sonido o unidades de almacenamiento IDE/SATA, etc.
- **Snapshots** -> VirtualBox puede generar snapshots de forma arbitraria en base al estado actual de la máquina virtual, pudiendo volver a dicho estado en caso de ser necesario revirtiendo todo lo ocurrido en la máquina virtual después de la snapshot.
- **Grupos de máquinas virtuales** -> VirtualBox permite agrupar máquinas virtuales a grupos para poder manejarlas de forma colectiva o individual, pudiendo añadir una máquina virtual a múltiples grupos o incluso aplicar una jerarquía entre ellos haciendo grupos de grupos.
- **Arquitectura limpia y modularidad** -> VirtualBox usa un diseño modular que permite diseñar interfaces limpias y fáciles de usar, separando código tanto por parte del lado cliente como del lado servidor, haciéndolo sencillo de manejar desde distintas interfaces.
- **Uso remoto** -> Es posible añadir una extensíon para VirtualBox para poder manejar las máquinas virtuales remotamente denominada VirtualBox Remote Desktop Extension.

Para el proyecto, se ha creado una máquina virtual mediante el uso de VirtualBox la cual usa como sistema operativo CentOs 8, creando la máquina virtual con las siguientes características:

- **Memoria base** -> 2048 MB
- **Almacenamiento**
  - Sistema operativo -> 20 GB
  - Base de datos -> 20 GB
  - Copias de seguridad periódicas -> 20 GB
- **Adaptadores de red**
  - Adaptador sólo anfitrión
  - NAT

Tanto la memoria base como las unidades de almacenamiento se pueden ampliar en el caso del entorno de producción de ser necesario. Se ha añadido un adaptador de red sólo anfitrión para poder conectar con la máquina virtual a través de él para operar sobre ella mediante el uso de SSH o conectar a la aplicación web iniciada en la máquina virtual a través de un navegador web, indicando en ambos casos la dirección IP que se le ha asignado a esa interfaz de red.

Capítulo 4

Metodología

En este apartado, se indicará la metodología que se ha usado para la organización de las tareas realizadas dentro del TFG, así como la adaptación que se ha hecho para poder aplicarla adecuadamente junto con la planificación que se ha realizado.

4.1 Scrum

La metodología que se ha decidido usar para la organización de las tareas del proyecto es Scrum. Scrum[53] es un proceso en el cual se aplican buenas prácticas para trabajar en equipo y apoyándose las unas sobre las otras para dar resultados de la forma más ágil posible. Se usa normalmente para proyectos que se van a aplicar en entornos complejos en los cuales hay que dar resultados rápido o bien los requisitos son poco definidos o están cambiando y añadiéndose constantemente.

Se ha elegido Scrum para organizar los requisitos que debe cumplir el proyecto porque es una metodología que permite organizar rápidamente las necesidades que se deben cumplir para el producto sin hacer muchas complicaciones a través de reuniones que se hacen de forma periódica, acordando los requisitos que no queden claros cómo resolverlos y tratando de llegar a una solución lógica, pudiendo también entregar resultados rápido y aprovechando también cada sprint para adaptar, mejorar o corregir los requisitos según sea necesario o se vayan encontrando errores por el camino.

4.2 Definiciones de Scrum

A continuación, se muestra un conjunto de conceptos que son necesarios para entender Scrum en su conjunto[54][55]:

4.2.1 Conceptos clave de Scrum

- **Sprint** -> Período de tiempo de 1-4 semanas de duración durante el cual se debe abordar las tareas planificadas.
- **Planificación del Sprint** -> Reuniones de equipo que sirven para determinar qué tareas se realizarán y se entregarán en el próximo sprint.
- **Reunión diaria** -> Reunión de 15 minutos que se hace todos los días donde todos los miembros cubren e forma rápida y transparente lo que ha hecho el día anterior, lo que va a hacer el mismo día y los problemas que se está encontrando.
- **Revisión del Sprint** -> Evento en el que el equipo presenta el trabajo completado durante el Sprint al product owner, quien comprueba el trabajo y lo acepta o rechaza según lo que se haya acordado en el proceso como ”Hecho”, dando feedback por parte de los clientes para asegurar que las tareas que se crean cumplan con las necesidades del negocio.
- **Retrospectiva** -> Reunión que se hace al final de un Sprint para ayudar a determinar lo que fue bien, lo que no y en qué puede mejorar el equipo, pudiendo identificar estrategias y en qué se puede mejorar.
- **Product Backlog** -> Lista de tareas que describen los requisitos que debe cumplir el proyecto, ordenándolos en base al valor que tienen dentro del negocio.
- **Sprint Backlog** -> Lista específica de elementos cogidos del product backlog que se deben completar en un Sprint.
- **Incremento** -> Es la suma de todas las tareas que se han completado desde la última versión de producto entregada, siendo responsabilidad del equipo asegurarse de que todo lo que hay en un incremento funciona como es debido y se puede entregar/poner en producción, pero esa decisión depende del Product Owner.

4.2.2 Participantes de Scrum

Dentro del equipo que compone Scrum, podemos distinguir 3 roles, los cuales son:

- **Scrum Master**[56] -> El Scrum Master es un mienbro del equipo en Scrum encargado de coordinar y enseñar al resto de los participantes a entender Scrum tanto en la teoría como en la práctica, tanto a nivel de equipo como a nivel de organización. En concreto, el Scrum Master ayuda a la organización de 3 formas distintas:

1. Equipo Scrum -> El Scrum Master ayuda al Equipo de Scrum por ejemplo entrenando a los miembros del equipo de forma que se autogestionen, enfocándolos a crear incrementos en el proyecto de forma que coincida con el concepto que se tenga en el momento de ”Hecho”, solventando los problemas que haya entre los miembros los cuales puedan afectar al progreso y asegurando que todos los incrementos que se hagan sean positivos, productivos y se hagan en tiempo.
2. Product Owner -> El Scrum Master ayuda al propietario del producto o Product Owner por ejemplo ayudándole a buscar un objetivo claro para el producto y manejar el Product Backlog del proyecto de forma adecuada, ayudando a establecer una planificación del producto de forma empírica o facilitando la colaboración de las partes interesadas del proyecto según se vaya solicitando o necesitando.
3. La propia organización -> El Scrum Master ayuda a la organización en aspectos como por ejemplo orientándola en la adopción de Scrum, planificando y advirtiendo implementaciones de Scrum dentro de ella, ayudando a los empleados y partes interesadas, ayudando a todo el mundo en la organización a entender y promulgar una aproximación empírica para entornos complejos y quitando barreras entre las partes interesadas y los equipos de Scrum.

- **Product Owner**[57] -> El Product Owner es el miembro del Equipo Scrum encargado de maximizar el valor del producto final resultante del trabajo en el Equipo Scrum. También es el encargado dentro del equipo de manejar el Product Backlog, por lo que eso implica que:

  1. Desarrolla y comunica el objetivo del producto.
  2. Crear y comunicar requisitos para el Product Backlog.
  3. Ordenar los requisitos del Product Backlog.
  4. Asegurar que el Product Backlog es visible, se entiende y es visible.

  El propio Product Owner se puede encargar de ello o bien se lo puede delegar a otros, pero todas las decisiones que tome el Product Owner deben ser respetadas por el resto del equipo.

- **Equipo de desarrollo**[58] -> El equipo de desarrollo dentro del Equipo Scrum se encarga de cerar cualquier aspecto que se indique dentro de los incrementos de cada sprint. Dentro del equipo, se encargan principalmente de lo siguiente:

  1. Crear un plan para el Sprint a realizar (Sprint Backlog).
  2. Añadir calidad a cada uno de los Sprints dando una definición de lo que se puee considerar como ”Hecho”.

3. Adaptar a diario el plan que se tenga para cumplir el objetivo del Sprint.
4. Representar a cada mienbro del equipo como profesionales.

4.3 Adaptación de la metodología

En esta sección, se indica qué es lo que se ha hecho para poder adaptar la metodología de desarrollo para el proyecto, ya que a pesar de que Scrum es una buena metodología de desarrollo para trabajar en equipo, dadas las circunstancias del proyecto no es posible seguir al pie de la letra todo lo que propone.

4.3.1 Roles

Como se ha mencionado anteriormente, en Scrum disponemos de tres roles, los cuales son el Scrum Master, el Product Owner y el Equipo de Desarrollo. Por una parte, el alumno se ha encargado de ejercer el rol de Equipo de Desarrollo, por lo que el propio alumno ha tenido que realizar tanto el análisis como el diseño, la implementación y las pruebas de toda la aplicación web, aplicando también el rol de Product Owner según sea necesario para poder definir los requisitos que debe cumplir el sistema y del Scrum Master, comprobando que se entegan todos los requisitos que se deban cumplir de forma periódica y en el tiempo estipulado.

Cabe destacar que el director del proyecto también ha ejercido el rol de Product Owner, aportando ideas nuevas al proyecto y sugiriendo soluciones para resolver algunos requisitos en los que hubieran dudas de por medio.

4.3.2 Reuniones

En cuanto a las reuniones que caracterizan a Scrum, lo primero que se ha hecho es una primera reunión antes de iniciar el proyecto para dar una perspectiva general del proyecto, indicando una vista general de los requisitos que se deben cumplir dentro de la aplicación y dónde se puede empezar a buscar ideas para el desarrollo de ésta, como por ejemplo aplicaciones similares a partir de las cuales se puedan sacar dichas ideas o ver qué se puede ofrecer como nuevo en comparación a ellas.

A partir de este punto, se han realizado reuniones de forma periódica a través de videollamadas en Microsoft Teams para indicar avances en el proyecto, ver qué requisitos están ya terminados o ver qué más se puede mejorar, complementando también mediante correos electrónicos escritos mediante Microsoft Outlook para dar una retroalimentación más rápida

entre ambas partes, indicando también si se está avanzando en la dirección correcta o hay algo que se pueda mejorar, ya que en ambas partes hay circunstancias que podían complicar la realización de dichas videollamadas debido a la situación de pandemia que se está viviendo al momento de escribir la memoria del proyecto.

Cabe destacar que se han omitido las reuniones diarias debido a que en el equipo de desarrollo sólo hay un miembro del equipo para la realización del proyecto, ya que la finalidad de éstas es la de fomentar el apoyo dentro de los participantes del proyecto.

4.4 Organización del proyecto

Para la organización del proyecto, tal y como se ha explicado anteriormente, se ha hecho de forma secuencial el análisis de requisitos junto con las historias de usuario, el desarrollo de la aplicación web y el paso al entorno de producción, realizando parte de la memoria en paralelo para luego dejar margen de tiempo para detectar errores tanto en ésta como en la aplicación web.

En total, se ha dedicado 3 semanas a lo largo del mes de Febrero para analizar y agrupar las historias de usuario para la aplicación, ordenando dichas agrupaciones según el requisito que se deba cumplir, alternando entre medias con períodos de formación autodidacta para profundizar más en alguna librería o framework en la cual se requiera su uso pero era necesario ampliar conocimientos o para investigar y comparar herramientas que puedan agilizar el desarrollo de los requisitos, llegando incluso a realizar pequeñas demos o prototipos para comprobar si compensa más usar una herramienta u otra a la hora de resolver el problema o buen para aprender buenas prácticas usando dichas herramientas. También se ha aprovechado para configurar y preparar todo lo necesario para tener el entorno de trabajo listo para realizar el proyecto, dejando listo y configurado tanto el entorno Maven del proyecto como el repositorio Git donde se guardará.

En cuanto al desarrollo de la aplicación, se ha desarrollado a lo largo de Marzo y Abril, dedicando un total de 6 semanas para dejar 2 semanas para preparar el entorno de producción, en las cuales se ha tratado de llegar a la meta de 10 historias de usuario por semana, pero debido a que en algunas se encontraron problemas a la hora de desarrollarlas o no quedaba del todo claro cómo resolverlas, haciendo que tanto el director del proyecto como el alumno pensasen en alternativas para ello y dedicando más tiempo de lo esperado, acabando con un total de 8 semanas.

Para el paso al entorno de producción, se han dedicado 2 semanas para la preparación y configuración de la máquina virtual que se ha usado para poner operativo el proyecto, corrigiendo fallos como por ejemplo la codificación de caracteres o algún requisito que ha quedado por comprobar entre medias, asegurando que todo debe funcionar como es debido en dicho entorno. Finalmente, se ha dedicado el tiempo restante hasta el día de depósito del proyecto para redactar las partes que hayan quedado por añadir a la memoria del proyecto y para comprobar algún detalle que haya quedado entre medias y que haya sido necesario aclarar entre el director del proyecto y el alumno. Cabe destacar que en cada día dedicado al proyecto se ha invertido una media de 4-8 horas de esfuerzo.

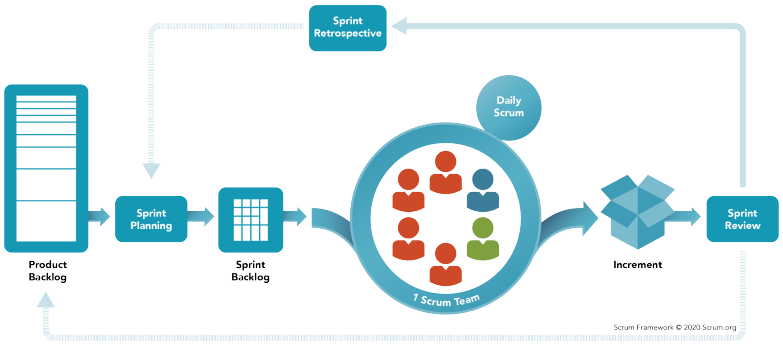


Figura 4.1: Ejemplo de Sprint en Scrum

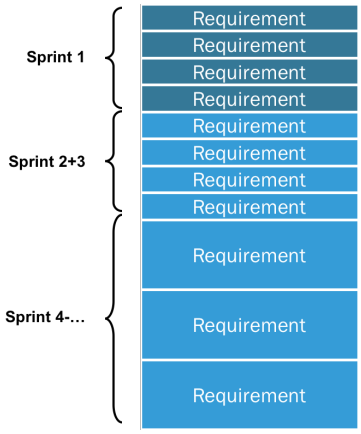


Figura 4.2: Esquema de muestra de Product Backlog

A continuación se indica el esfuerzo y el coste que se ha tenido que invertir para el proyecto, teniendo en cuenta los siguientes datos para el cálculo del coste monetario en €[59][60]:

- **Analista**[61] -> 1890 €/mes netos, 11.81 €/h

- **Programador**[62] -> 1610 €/mes netos, 10.06 €/h
- **Administrador de sistemas**[63] -> 2460.71 €/mes brutos, 1901.64 €/mes netos, 11.89 €/h

Tarea	Esfuerzo (h.h)	Coste (€)
Análisis de requisitos	12	141.72
Sprint para el requisito 1	4	40.24
Sprint para el requisito 2	4	40.24
Sprint para el requisito 3	10	100.60
Sprint para el requisito 4	6	60.36
Sprint para el requisito 5	8	80.48
Sprint para el requisito 6	8	80.48
Sprint para el requisito 7	4	40.24
Sprint para el requisito 8	4	40.24
Sprint para el requisito 9	4	40.24
Sprint para el requisito 10	8	80.48
Sprint para el requisito 11	12	120,72
Sprint para el requisito 12	6	60.36
Sprint para el requisito 13	20	201.20
Sprint para el requisito 14	6	60.36
Sprint para el requisito 15	40	402.40
Sprint para el requisito 16	6	60.36
Sprint para el requisito 17	2	20.12
Sprint para el requisito 18 y 19	10	100.60
Sprint para el requisito 20	6	60.36
Sprint para el requisito 21	6	60.36
Sprint para el requisito 22	3	30.18
Sprint para el requisito 23	7	70.42
Sprint para el requisito 24	4	40.24
Sprint para el requisito 25	24	241.44
Sprint para el requisito 26	4	40.24
Sprint para el requisito 27	4	40.24
Configuración y paso al entorno de producción	40	475.60
TOTAL	338	2367,76

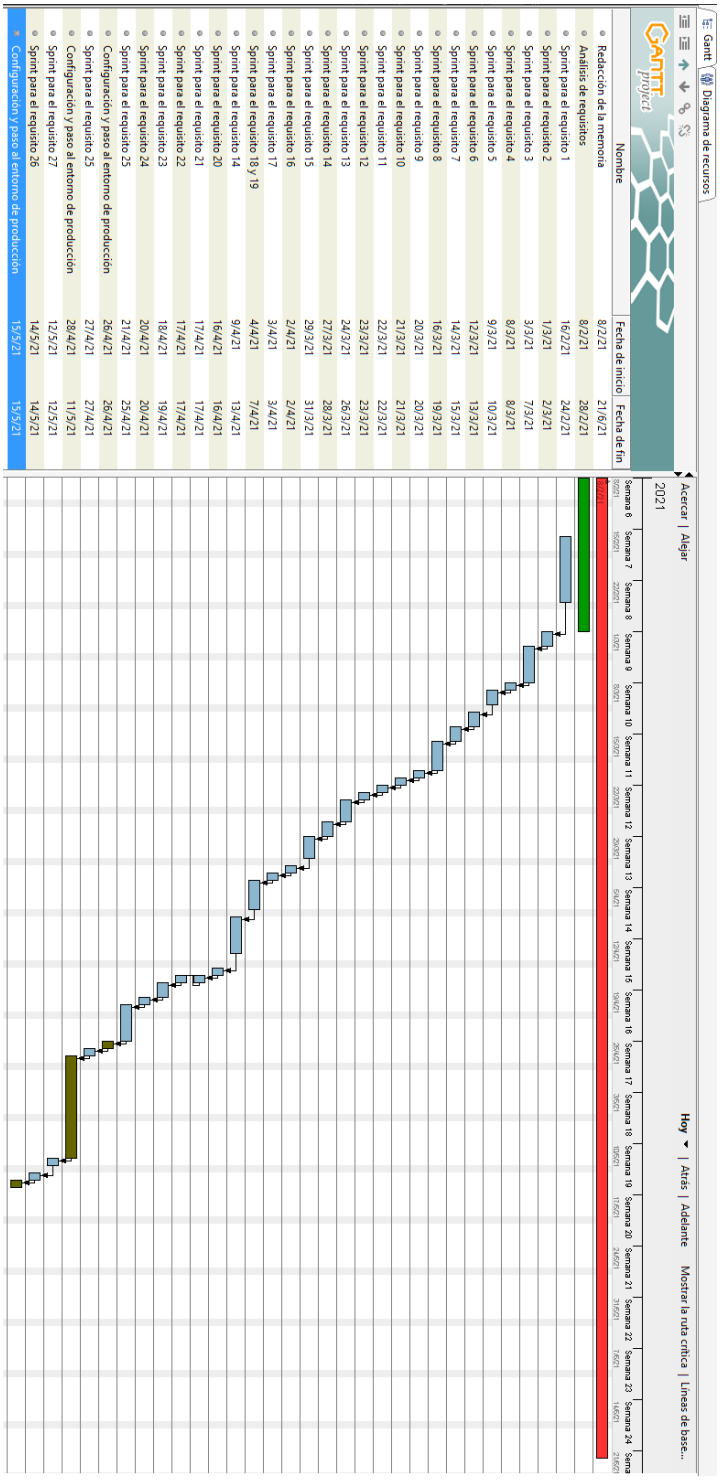


Figura 4.3: Diagrama de Gantt del proyecto

Capítulo 5

Diseño y desarrollo de la aplicación

En este capítulo, se indicará cómo se ha diseñado y desarrollado la aplicación web, indicando los sprints que se han hecho a lo largo del desarrollo en base a la metodología indicada con anterioridad.

5.1 Análisis de requisitos y formación (Sprint 0)

Antes de empezar a desarrollar la aplicación web, primero se deben indicar los requisitos que se deben cumplir, tanto los funcionales como los no funcionales, para poder saber las necesidades que debe cubrir y cómo. También se ha aprovechado el sprint para profundizar más en aprender a usar mejor las tecnologías y herramientas que se deben usar.

5.1.1 Requisitos funcionales

En este apartado, se mostrarán cada uno de los requisitos funcionales, los cuales representan las agrupaciones de funcionalidades que se esperan del sistema.

1. El sistema debe permitir poder loguearse en el sistema mediante usuario y contraseña, logueando al usuario como facultativo o admin. del sistema según el rol que corresponda.

2. Como administrador, el usuario puede ver la lista de usuarios registrados en el sistema, indicando nombre y apellidos, login, fecha de alta y rol dentro del sistema, pudiendo a partir de dicha lista actualizar o dar de alta/baja dichos usuarios.

3. Como administrador, el usuario puede dar de alta, actualizar información o dar de baja usuarios con el mismo rol o con el rol de facultativo, aportando los datos que se consideren necesarios, tales como nombre, apellidos, DNI/NIF, foto de perfil, teléfono de contacto, email, contraseña, etc.

4. En dicha lista, el administrador puede aplicar criterios de búsqueda en base a los distintos campos disponibles.

5. El administrador puede gestionar los horarios de los usuarios, indicando la jornada laboral que tienen durante la semana.

6. En el caso de usuarios facultativos, el administrador puede añadir una o varias especialidades disponibles para dichos facultativos registrados dentro del sistema, pudiendo actualizar el nombre de la especialidad e indicar si está activa o no.

7. En el caso de medicamentos, el administrador puede gestionar la lista de medicamentos disponibles para poder realizar las recetas para los pacientes del sistema, pudiendo crear, actualizar e indicar si dichos medicamentos están activos o no.

8. En el caso de pruebas diagnósticas, el administrador puede gestionar la lista de pruebas diagnósticas que se realizan dentro de la clínica, indicando para cada una el nombre y pudiendo crearlas, actualizarlas y dejarlas para disponibilidad o no.

9. En el caso de métodos anticonceptivos, el administrador puede gestionar la lista de pruebas diagnósticas que se realizan dentro de la clínica, indicando para cada una el nombre y pudiendo crearlas, actualizarlas y dejarlas para disponibilidad o no.

10. Como administrador, el usuario puede indicar el nombre de la empresa en la cual se esté usando la aplicación junto con un logotipo, siendo visibles en una o varias partes de la aplicación.

11. Como facultativo, el usuario puede ver un listado de las pacientes registradas dentro del sistema, mostrando nombre, apellidos, DNI y número de historia en el SERGAS, pudiendo buscar pacientes en base a campos disponibles en el listado.

12. Como facultativo, el usuario puede consultar antecedentes relacionados con la paciente, tales como hábitos, vacunas, alergias, antecedentes familiares entre otros. También se puede dejar constancia de la fecha de la última regla y el n.º de partos, cesáreas y abortos que haya tenido.

13. Como facultativo, el usuario puede consultar el diario clínico asociado a la paciente, pudiendo consultar la fecha de la entrada del diario, la actividad que se ha realizado y qué facultativo la ha realizado, junto con un conjunto de archivos adjuntos que sirvan de apoyo para la entrada del diario clínico (estos pueden ser imágenes como por ejemplo ecografías o documentos aparte con formato como por ejemplo PDF). A mayores también puede consultar los cuestionarios realizados a la paciente en caso de que haya acudido a la clínica con anterioridad.

14. Como facultativo, el usuario puede dar de alta a pacientes, modificar sus datos o darlas de baja dentro del sistema, indicando toda la información de la paciente que se considere necesario, como el nombre, apellidos, DNI/NIF, teléfono de contacto, email, etc., incluyendo los antecedentes que tenga.

15. Como facultativo, el usuario puede añadir una entrada al curso clínico de una paciente, pudiendo añadir preguntas a un cuestionario, pruebas complementarias o prescribir recetas en ese momento.

16. Como facultativo, dentro de la lista el usuario puede marcar las pacientes que le interese como pacientes de interés, mostrándose como tal en otro listado aparte donde se muestre en una lista con las mismas características mencionadas anteriormente en el listado de pacientes pero sólo mostrando las pacientes que al médico le interesan, pudiendo también desmarcarlas para que no aparezcan en la lista.

17. Como facultativo, el usuario puede ver en otra lista aparte la lista de las últimas pacientes que se han visto, mostrando su nombre y la fecha de la última cita que han tenido con el facultativo, ordenando por recencia en base a la fecha mencionada anteriormente.

18. Como facultativo, el usuario dispone de una agenda dentro de la cual puede consultar las citas que tiene para hoy, mañana o durante los próximos días, mostrándose en ese orden respectivamente.

19. Como administrador, el usuario puede añadir citas para las pacientes, indicando la paciente junto con el facultativo por la cual será atendida, cuándo será atendida, la actividad que se va a realizar y comentarios a mayores asociados a la paciente junto con el estado de la cita, pudiendo modificarlas o cancelarlas en caso de que sea necesario.

20. Como administrador, el usuario puede generar plantillas de informes de seguimiento para las pacientes para distintos fines, pudiendo consultarlos o descargarlos.

21. Como facultativo, el usuario puede intercambiar mensajes con otros facultativos que estén registrados dentro del sistema, mostrando la fecha y la hora en la que se ha realizado el mensaje.

22. Como facultativo, el usuario puede solicitar una interconsulta con otro facultativo que esté registrado dentro del sistema a partir de la entrada del diario clínico que se esté creando en ese momento, indicando el motivo por el cual se está realizando.

23. Como facultativo, el usuario también puede intercambiar mensajes entre facultativos que tengan tareas en común, funcionando de forma parecida a un foro donde se comunican entre ellos.

24. Cualquier usuario del sistema puede lanzar un aviso por la razón que sea, indicando el motivo del aviso y cuándo se ha realizado.

25. Como administrador, el usuario puede comprobar un registro de las acciones ocurridas durante la ejecución de la aplicación, pudiendo aplicar filtros dentro de él y generar un informe de acceso del registro.

26. Como administrador, el usuario puede realizar una copia de seguridad de la BBDD de forma manual, descargándola para que la guarde donde considere conveniente.

27. Como administrador, el usuario puede recuperar la BBDD en un momento determinado a partir de la copia de seguridad que se indique.

5.1.2 Requisitos no funcionales

En este apartado, se mostrarán cada uno de los requisitos no funcionales, los cuales representan aspectos que se deben cumplir como por ejemplo seguridad o estructuración del código.

1. La aplicación web debe seguir el patrón de diseño Modelo-Vista-Controlador (MVC).

2. Los datos deben estar guardados dentro de una BBDD relacional, como por ejemplo MySQL.

3. Se debe aplicar el uso de librerías de código abierto que se apoyan internamente en APIs Java de más bajo nivel, entre las cuales están por ejemplo Hibernate para la persistencia en BBDD, Spring MVC para aplicar el patrón de diseño Modelo-Vista-Controlador, Thymeleaf para el renderizado de plantillas en HTML, JQuery para agilizar el agregado de funcionalidades a dichas plantillas en aspectos como validación o menús desplegables o Bootstrap para el estilizado de éstas mediante el uso de clases CSS basadas en componentes.

4. La aplicación debe seguir las pautas indicadas por la Agencia Española de Protección de datos (AEPD), Ley Orgánica de Protección de Datos y Garantía de los Derechos Digitales 3/2018, Ley de Servicios de la Sociedad de la Información y del Comercio Electrónico 34/2002 y el Reglamento General de Protección de datos UE 2016/679.

5. La aplicación web se debe desarrollar sobre un ordenador con acceso a Internet y el cual disponga de lo necesario para el desarrollo de ésta.

6. El código fuente de la aplicación se debe guardar dentro de un repositorio de control de versiones en Git.

7. La aplicación junto con la BBDD se deben desplegar mediante el uso de contenedores Docker con el apoyo de docker-compose para la automatización del despliegue de éstos, todo ello sobre CentOS Minimal.

8. Se debe usar alguna librería de JavaScript para el estilizado de texto sobre ciertos aspectos de la aplicación.

5.1.3 Modelo de datos

En este apartado, se indica tanto el modelo entidad-relación (ver figura 5.1) como el modelo de datos (ver figura 5.2). El primer modelo se ha creado a partir de los requisitos funcionales que se han definido con anterioridad, indicando las entidades que van a definir las tablas de la BBDD en la cual se van a guardar los datos que maneja la aplicación junto con sus atributos y las relaciones entre ellas, mientras que en el modelo UML se indican las entidades persistentes, las cuales representan las tablas de la BBDD con las que interactúa la propia aplicación.

A partir de ambos modelos, podemos destacar las siguientes entidades, las cuales son:

- **Usuario** -> Entidad que representa la tabla donde se guarda los datos de los usuarios tanto personales como de usuario del sistema (nombre de usuario y contraseña) entre otros datos.

- **Rol** -> Entidad que representa los roles que se les puede asignar a los usuarios. Los únicos roles disponibles hasta el momento son los de facultativo y de administrativo, pero se puede añadir más en caso de que se quiera añadir y/o adaptar funcionalidades.

- **Jornada** -> Entidad débil que representa las jornadas laborales que tienen cada uno de los usuarios del sistema. Cada jornada/registro de la tabla contiene el día de la semana junto con una hora de inicio y de fin.

- **Especialidad** -> Entidad que representa las especialidades que se le puede asignar a los usuarios que tengan el rol de facultativo.

- **Mensaje** -> Entidad que representa los mensajes privados que se mandan los usuarios del sistema entre si. Cabe destacar que tiene una relación consigo misma para representar que puede haber mensajes que se escriban como respuesta a un mensaje anterior.

- **Tarea común** -> Entidad que representa las tareas comunes que tienen los usuarios del sistema entre si, guardando la descripción de la tarea común y el título.

- **Mensaje grupal** -> Entidad que representa los mensajes grupales que se mandan en las tareas comunes, relacionándose con los usuarios y las tareas comunes para saber a qué usuario y qué tarea común van asociados.
- **Aviso** -> Entidad que representa los avisos que se lanzan dentro de la aplicación y los cuales van dirigidos a todos los usuarios de la aplicación.
- **Paciente** -> Entidad que representa la tabla en la cual se guardan todos los datos de las pacientes. Se guarda tanto sus datos personales como sus antecedentes personales y ginecológicos. Se añade una relación extra entre usuario y paciente para indicar qué pacientes son de interés para qué usuarios del sistema.
- **Anticonceptivo** -> Entidad que representa los métodos anticonceptivos que se puede asignar a las pacientes del sistema.
- **Cita** -> Entidad que representa las citas que han tenido las pacientes al pasar por la clínica ginecológica y las cuales representan las entradas de su historial clínico.
- **Entrada del calendario** -> Entidad que representa las entradas del calendario de las agendas de los usuarios del sistema que tengan el rol de facultativo, indicando la fecha, el motivo de la entrada del calendario, el estado y para qué paciente es junto con el facultativo por la cual será atendida.
- **Receta** -> Entidad que representa las recetas que se han prescrito para las pacientes en las citas, pudiendo descargarse más tarde desde las entradas del diario clínico.
- **Medicamento** -> Entidad que representa los medicamentos que se pueden recetar en las recetas prescritas.
- **Archivo** -> Entidad que representa los archivos que sirven como pruebas diagnósticas de las pacientes, los cuales se van añadiendo a partir de las entradas del diario clínico.
- **Prueba diagnóstica** -> Entidad que representa las pruebas diagnósticas que se pueden añadir para las pacientes.
- **Pregunta** -> Entidad que representa las preguntas que se han hecho con anterioridad dentro de la clínica ginecológica para que se puedan volver a usar.
- **Respuesta** -> Entidad que representa las respuestas recibidas por parte de las pacientes en las correspondientes citas realizadas.

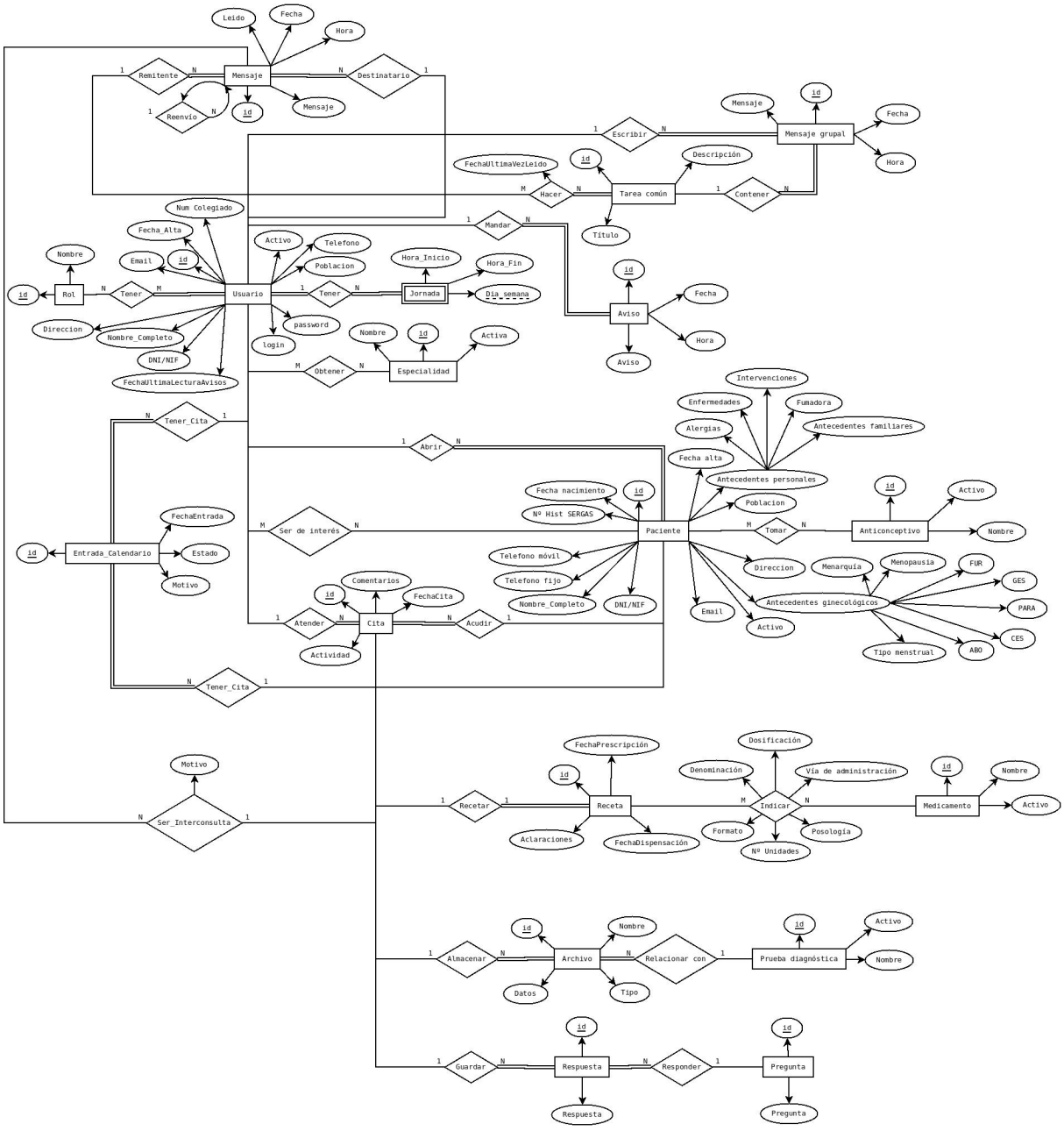


Figura 5.1: Modelo entidad-relación

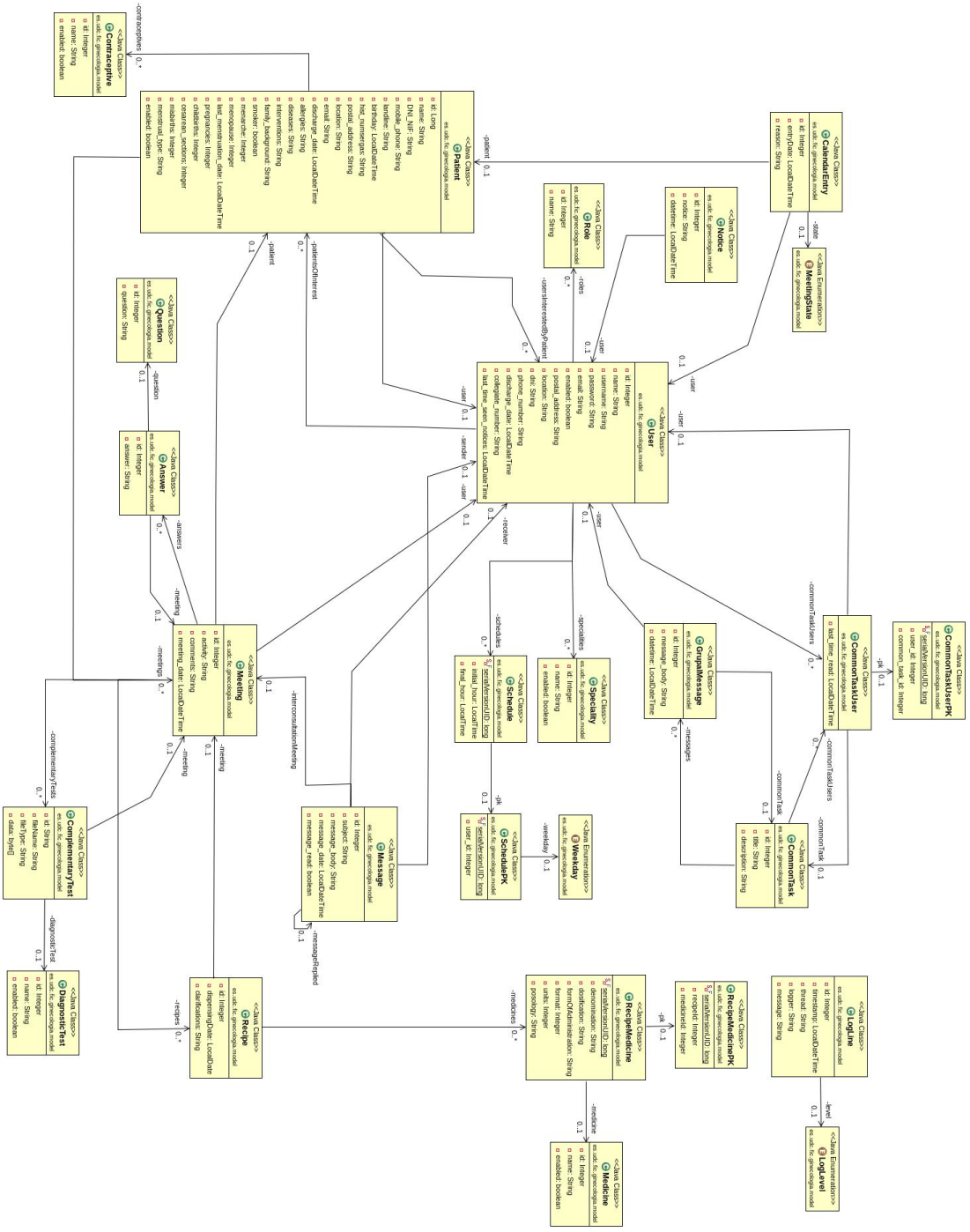


Figura 5.2: Modelo UML

5.2 Desarrollo de la aplicación web (Sprint 1 en adelante)

En esta sección, se indicará cada uno de los sprints realizados para cada uno de los requisitos, indicando para cada uno de ellos cómo se han dividido en pequeñas historias de usuario para poder dividirlos en partes más pequeñas y comprender mejor el requisito que se debe implementar. Los requisitos son los indicados dentro del Sprint 0.

5.2.1 Sprint 1

- **Descripción** -> Sprint para cubrir el requisito 1
- **Historias de usuario**:
  - HU-1 Login dentro del sistema -> El usuario puede entrar dentro del sistema.
  - HU-2 Loguearse fuera del sistema -> El usuario puede salir del sistema.

5.2.2 Sprint 2

- **Descripción** -> Sprint para cubrir el requisito 2
- **Historias de usuario**:
  - HU-3 Ver lista de usuarios -> El usuario puede ver la lista de usuarios registrados dentro del sistema.

5.2.3 Sprint 3

- **Descripción** -> Sprint para cubrir el requisito 3
- **Historias de usuario**:
  - HU-4 Dar de alta usuario -> El usuario puede registrar un perfil que pueda ser usado para un usuario dentro del sistema.
  - HU-5 Actualizar usuario -> El usuario puede actualizar un perfil que haya sido registrado anteriormente dentro del sistema.
  - HU-6 Dar de baja usuario -> El usuario puede dar de baja un perfil que haya sido registrado anteriormente dentro del sistema.

5.2.4 Sprint 4

- **Descripción** -> Sprint para cubrir el requisito 4
- **Historias de usuario**:

  - HU-7 Buscar usuarios -> El usuario puede buscar perfiles que ya estén registrados dentro del sistema.

5.2.5 Sprint 5

- **Descripción** -> Sprint para cubrir el requisito 5
- **Historias de usuario**:
  - HU-8 Ver jornada laboral -> El usuario puede ver el horario de un perfil de usuario en concreto del sistema.
  - HU-9 Añadir jornada laboral -> El usuario puede añadir una jornada laboral para el usuario del sistema que se esté modificando.
  - HU-10 Modificar jornada laboral -> El usuario puede modificar una jornada laboral para el usuario del sistema que se esté modificando.
  - HU-11 Quitar jornada laboral -> El usuario puede modificar una jornada laboral para el usuario del sistema que se esté modificando.

5.2.6 Sprint 6

- **Descripción** -> Sprint para cubrir el requisito 6
- **Historias de usuario**:
  - HU-12 Ver especialidades -> El usuario puede ver las especialidades disponibles para asignar a los facultativos.
  - HU-13 Añadir especialidad -> El usuario puede añadir una especialidad para asignar a los facultativos.
  - HU-14 Modificar especialidad -> El usuario puede modificar una especialidad para asignar a los facultativos.
  - HU-15 Dar de baja especialidad -> El usuario puede quitar una especialidad para asignar a los facultativos.
  - HU-16 Buscar especialidades -> El usuario puede buscar en la lista de especialidades en base a unos criterios de búsqueda determinados.
  - HU-17 Asignar especialidad a facultativo -> El usuario puede asignar una o varias especialidades a un facultativo.
  - HU-18 Quitar especialidad a facultativo -> El usuario puede quitar una o varias especialidades a un facultativo.

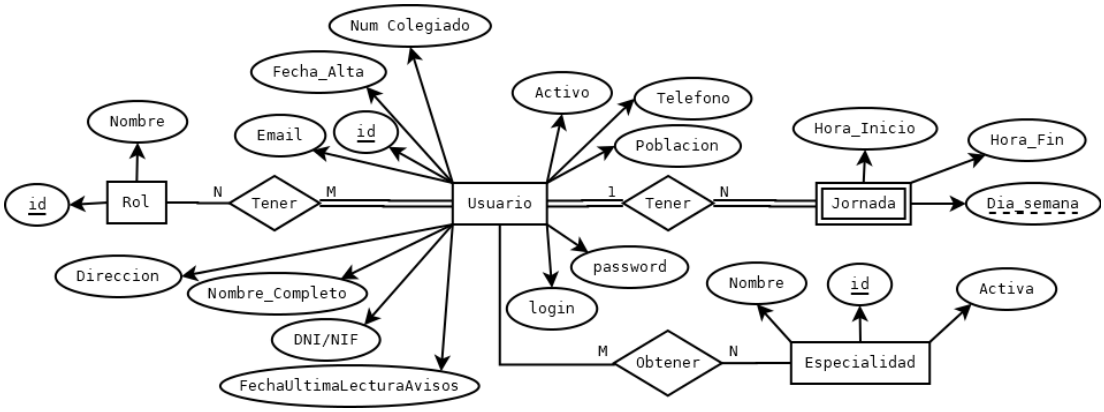
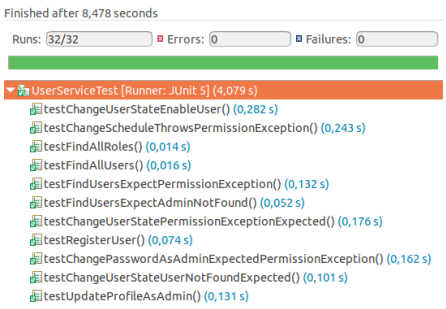
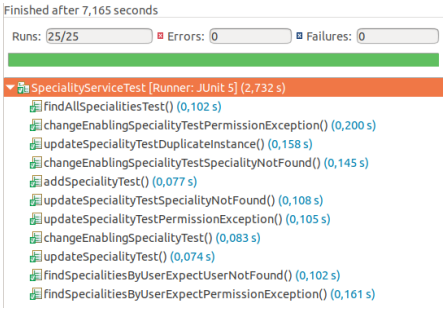


Figura 5.3: Estado del modelo de datos en el sprint del 1 al 6 (ver figura C.1)



(a) Pruebas unitarias del servicio de usuarios



(b) Pruebas unitarias del servicio de gestión de especialidades

Figura 5.4: Pruebas unitarias en los sprints del 1 al 6

5.2.7 Sprint 7

- **Descripción** -> Sprint para cubrir el requisito 7
- **Historias de usuario**:
  - HU-19 Ver medicamentos -> El usuario puede ver los medicamentos disponibles para recetar a las pacientes.
  - HU-20 Añadir medicamento -> El usuario puede añadir los medicamentos para recetar a las pacientes.
  - HU-21 Modificar medicamento -> El usuario puede modificar los datos de un medicamento para recetar a las pacientes.
  - HU-22 Quitar medicamento -> El usuario puede dar de baja un medicamento para recetar a las pacientes.
  - HU-23 Buscar medicamentos -> El usuario puede buscar medicamentos en base a una serie de criterios de búsqueda.

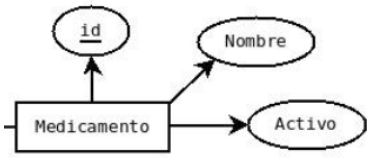
5.2.8 Sprint 8

- **Descripción** -> Sprint para cubrir el requisito 8
- **Historias de usuario**:
  - HU-24 Ver pruebas diagnósticas -> El usuario puede ver la lista de pruebas diagnósticas disponibles dentro de la clínica.
  - HU-25 Añadir prueba diagnóstica -> El usuario puede añadir una prueba diagnóstica para realizar dentro de la clínica.
  - HU-26 Modificar prueba diagnóstica -> El usuario puede modificar los datos de una prueba diagnóstica para realizar dentro de la clínica.
  - HU-27 Dar de baja prueba diagnóstica -> El usuario puede dar de baja una prueba diagnóstica para realizar dentro de la clínica.
  - HU-28 Buscar pruebas diagnósticas -> El usuario puede buscar pruebas diagnósticas en base a una serie de criterios de búsqueda.

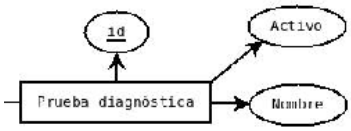
5.2.9 Sprint 9

- **Descripción** -> Sprint para cubrir el requisito 9
- **Historias de usuario**:

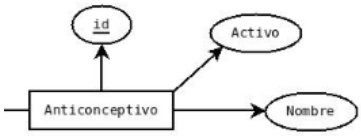
- HU-29 Ver métodos anticonceptivos -> El usuario puede ver la lista de métodos anticonceptivos registrados en el sistema.
- HU-30 Añadir método anticonceptivo -> El usuario puede registrar un método anticonceptivo dentro del sistema.
- HU-31 Modificar método anticonceptivo -> El usuario puede ver la lista de métodos anticonceptivos registrados en el sistema.
- HU-32 Dar de baja método anticonceptivo -> El usuario puede dar de baja un método anticonceptivo registrado en el sistema.
- HU-33 Buscar métodos anticonceptivos -> El usuario puede buscar métodos anticonceptivos registrados en el sistema.



(a) Estado del modelo de datos en el sprint 7

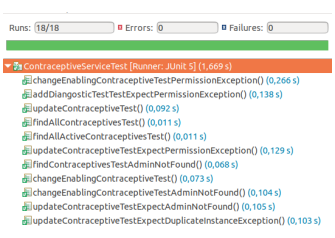


(b) Estado del modelo de datos en el sprint 8

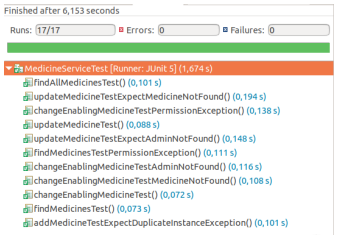


(c) Estado del modelo de datos en el sprint 9

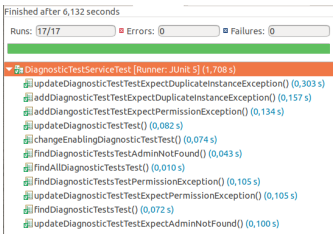
Figura 5.5: Estado del modelo de datos en los sprints del 7 al 9 (Ver figura C.2). Tanto en el caso de gestión de especialidades como de métodos anticonceptivos y pruebas diagnósticas, la pantalla es similar.



(a) Pruebas unitarias del servicio de gestión de métodos anticonceptivos



(b) Pruebas unitarias del servicio de gestión de medicamentos



(c) Pruebas unitarias del servicio de gestión de pruebas diagnósticas

Figura 5.6: Pruebas unitarias en los sprints del 7 al 9

5.2.10 Sprint 10

- **Descripción** -> Sprint para cubrir el requisito 10
- **Historias de usuario**:
  - HU-34 Cambiar nombre y logo -> El usuario puede cambiar el nombre y el logo de la aplicación web (Ver figura C.3).

5.2.11 Sprint 11

- **Descripción** -> Sprint para cubrir el requisito 11
- **Historias de usuario**:
  - HU-35 Ver pacientes -> El usuario puede ver los pacientes registrados en el sistema hasta el momento.
  - HU-36 Buscar pacientes -> El usuario puede buscar pacientes en base a una serie de criterios de búsqueda.
  - HU-37 Ver detalles de paciente -> El usuario puede ver los detalles de una paciente registrada en el sistema.

5.2.12 Sprint 12

- **Descripción** -> Sprint para cubrir el requisito 12
- **Historias de usuario**:
  - HU-38 Ver antecedentes -> El usuario puede ver los antecedentes relacionados con la paciente que se hayan anotado con anterioridad.

5.2.13 Sprint 13

- **Descripción** -> Sprint para cubrir el requisito 13
- **Historias de usuario**:
  - HU-39 Consultar curso clínico -> El usuario puede ver el historial de visitas de la paciente.
  - HU-40 Consultar entrada del curso clínico -> El usuario puede ver en detalle una de las entradas del curso clínico de la paciente.
  - HU-41 Ver cuestionario -> El usuario puede ver el cuestionario realizado en la entrada del curso clínico.

  - HU-42 Ver pruebas complementarias de la entrada del curso clínico -> El usuario puede ver las pruebas complementarias añadidas en la entrada del curso clínico.

5.2.14 Sprint 14

- **Descripción** -> Sprint para cubrir el requisito 14
- **Historias de usuario**:
  - HU-43 Añadir paciente -> El usuario puede ver las pacientes registradas en el sistema hasta el momento.
  - HU-44 Modificar paciente -> El usuario puede modificar los datos a los que se tengan acceso de una paciente en concreto.
  - HU-45 Dar de baja paciente -> El usuario puede dar de alta o de baja los datos de una paciente.

5.2.15 Sprint 15

- **Descripción** -> Sprint para cubrir el requisito 15
- **Historias de usuario**:
  - HU-46 Añadir entrada a curso clínico -> El usuario puede añadir una entrada al curso clínico de la paciente.
  - HU-47 Añadir pregunta en cuestionario -> El usuario puede añadir una pregunta que se haya contestado por parte de la paciente dentro de la cita asociada a la entrada del curso clínico.
  - HU-48 Añadir prueba complementaria a entrada del curso clínico -> El usuario puede añadir pruebas complementarias a la entrada del curso clínico.
  - HU-49 Prescribir receta -> El usuario puede prescribir una receta para la paciente asociándola a la cita que se esté dando en ese momento.

5.2.16 Sprint 16

- **Descripción** -> Sprint para cubrir el requisito 16
- **Historias de usuario**:
  - HU-50 Marcar paciente como de interés -> El usuario puede marcar pacientes como pacientes de interés.

- HU-51 Ver pacientes de interés -> El usuario puede ver las pacientes de interés para el facultativo.
- HU-52 Desmarcar paciente como de interés -> El usuario puede desmarcar pacientes como pacientes de interés.

5.2.17 Sprint 17

- **Descripción** -> Sprint para cubrir el requisito 17
- **Historias de usuario**:
  - HU-53 Ver últimos pacientes vistos -> El usuario puede ver las últimas pacientes vistas por el facultativo.

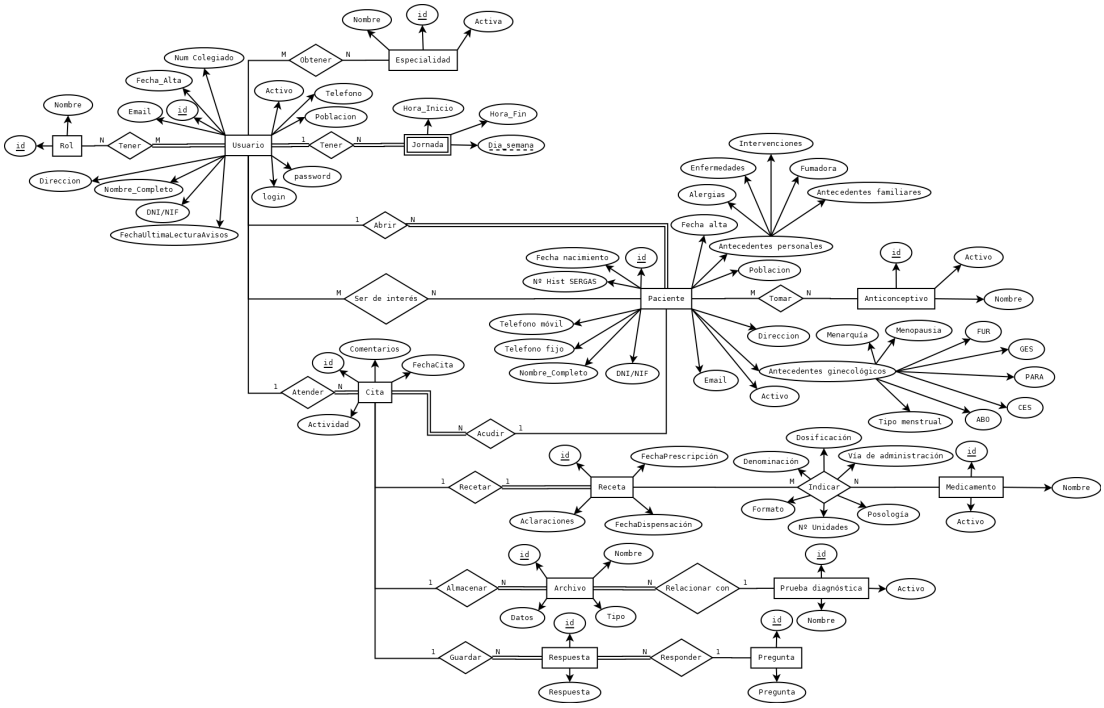


Figura 5.7: Estado del modelo de datos en los sprints del 10 al 17 (Ver figuras C.4 y C.5)

5.2.18 Sprint 18

- **Descripción** -> Sprint para cubrir los requisitos 18 y 19
- **Historias de usuario**:
  - HU-54 Consultar agenda -> El usuario puede consultar las citas que tiene el usuario.

  - HU-55 Ver citas -> El usuario puede ver las citas registradas hasta el momento.
  - HU-56 Añadir cita -> El usuario puede añadir una cita a la agenda.
  - HU-57 Modificar cita -> El usuario puede modificar una cita registrada en la agenda.
  - HU-58 Cancelar cita -> El usuario puede cancelar una cita registrada en la agenda.
  - HU-59 Atender cita -> El usuario puede atender una cita que esté registrada en la agenda.

5.2.19 Sprint 19

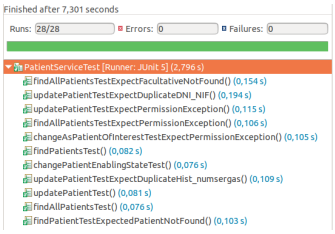
- **Descripción** -> Sprint para cubrir el requisito 20
- **Historias de usuario**:
  - HU-60 Generar informe de seguimiento -> El usuario puede generar un informe de seguimiento a partir de los datos de una paciente.

5.2.20 Sprint 20

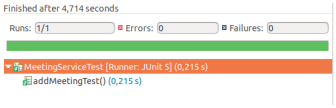
- **Descripción** -> Sprint para cubrir el requisito 21
- **Historias de usuario**:
  - HU-61 Ver mensajes privados -> El usuario puede ver los mensajes privados recibidos por otros usuarios.
  - HU-62 Leer mensaje privado -> El usuario puede ver los mensajes privados recibidos por otros usuarios.
  - HU-63 Enviar mensaje privado -> El usuario puede enviar un mensaje privado a otro usuario del sistema.

5.2.21 Sprint 21

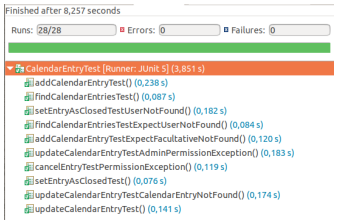
- **Descripción** -> Sprint para cubrir el requisito 22
- **Historias de usuario**:
  - HU-64 Solicitar interconsulta -> El usuario puede solicitar una interconsulta a un facultativo a partir de una entrada del curso clínico para pedir opinión.



(a) Pruebas unitarias del servicio de gestión de pacientes



(b) Pruebas unitarias del servicio de gestión de entradas del curso clínico



(c) Pruebas unitarias del servicio de gestión de agendas

Figura 5.8: Pruebas unitarias en los sprints del 10 al 18

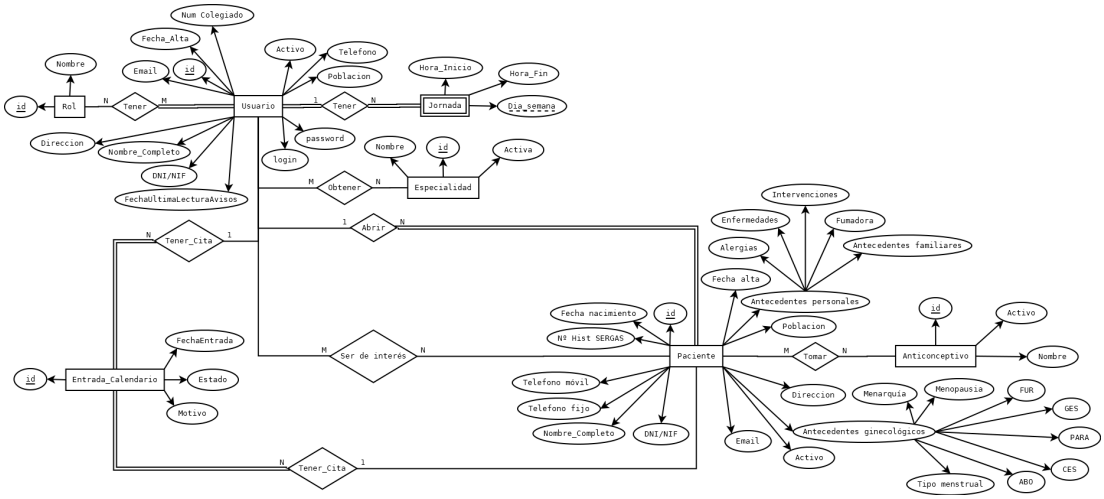


Figura 5.9: Estado del modelo de datos en el sprint 18 (Ver figuras C.6 C.7)

5.2.22 Sprint 22

- **Descripción** -> Sprint para cubrir el requisito 23
- **Historias de usuario**:
  - HU-65 Ver tareas comunes -> El usuario puede ver las tareas comunes asociadas al usuario.
  - HU-66 Ver mensajes de tareas comunes -> El usuario puede leer los mensajes asociados a una tarea común seleccionada.
  - HU-67 Abrir tarea común -> El usuario puede abrir una tarea común entre usuarios.
  - HU-68 Añadir mensaje para tarea común -> El usuario puede añadir un mensaje para la tarea común que se haya creado.

5.2.23 Sprint 23

- **Descripción** -> Sprint para cubrir el requisito 24
- **Historias de usuario**:
  - HU-69 Ver avisos -> El usuario puede ver los avisos recibidos.
  - HU-70 Añadir aviso -> El usuario puede añadir un aviso a cualquier usuario registrado en el sistema.

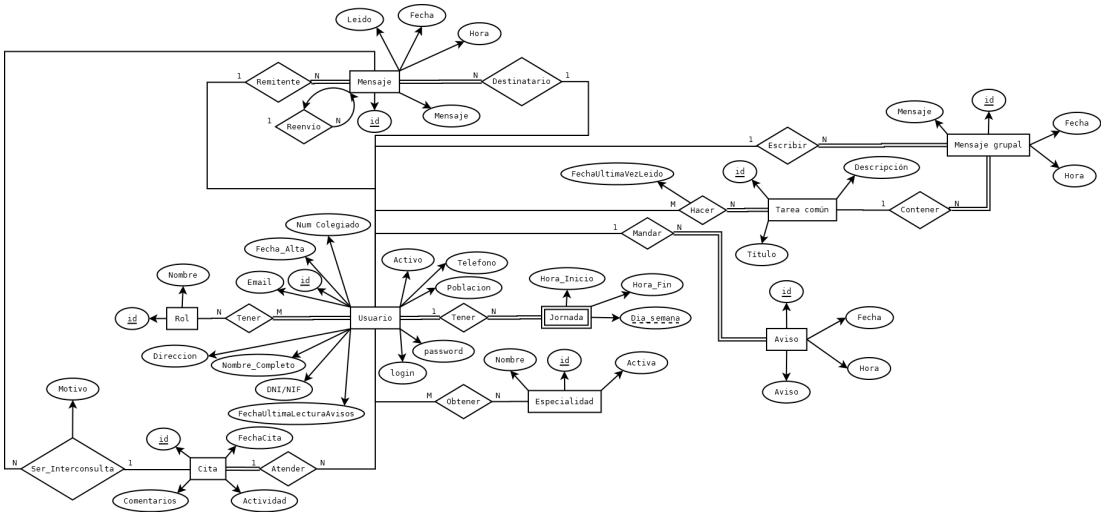
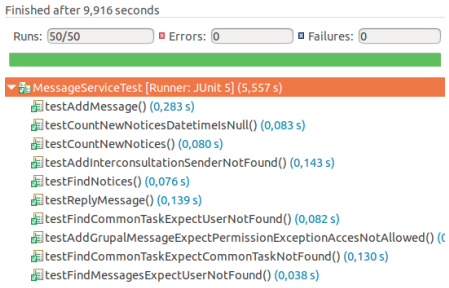


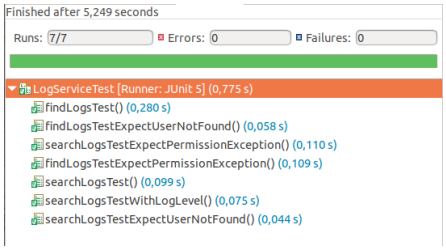
Figura 5.10: Estado del modelo de datos en los sprints del 20 al 23 (Ver figuras C.8, C.9 y C.10).

5.2.24 Sprint 24

- **Descripción** -> Sprint para cubrir el requisito 25 (Ver figura C.11).
- **Historias de usuario**:
  - HU-71 Comprobar log de acciones -> El usuario puede comprobar el historial de acciones realizadas dentro del sistema.
  - HU-72 Generar informe de acceso -> El usuario puede generar un informe de acceso en base al estado actual del log de acciones.
  - HU-73 Filtrar log de acciones -> El usuario puede filtrar las acciones registradas en el log de acciones.



(a) Pruebas unitarias del servicio de mensajería



(b) Pruebas unitarias del servicio de logs

Figura 5.11: Pruebas unitarias en los sprints del 20 al 24

5.2.25 Sprint 25

- **Descripción** -> Sprint para cubrir el requisito 26 (Ver figura C.12).
- **Historias de usuario**:
  - HU-74 Generar copia de seguridad -> El usuario puede generar una copia de seguridad del estado actual dela BBDD.

5.2.26 Sprint 26

- **Descripción** -> Sprint para cubrir el requisito 27
- **Historias de usuario**:
  - HU-75 Restaurar BBDD desde copia de seguridad -> El usuario puede restaurar la BBDD a partir de un archivo en formato .sql el cual permite restaurar el estado de la BBDD en ese momento.

Capítulo 6

Estructura del proyecto

En este apartado, se indica la estructura en la cual se ha organizado el proyecto para organizar el código implementado y los ficheros de configuración usados para el proyecto.

6.1 Código en Java

Las clases que se han programado mediante el uso de Java se clasifican dentro de paquetes Java dependiendo del propósito que complan dentro de la aplicación web para la lógica de negocio.

Los paquetes que se mencionan se usan en Java con el fin de clasificar el código escrito de forma modular, conteniendo desde interfaces hasta clases distribuídas como un archivo. Dichos archivos se pueden importar en otras clases o interfaces de Java mediante el uso de la sentencia ”import”, pudiendo reutilizar su código en distintas partes del proyecto según sea necesario.

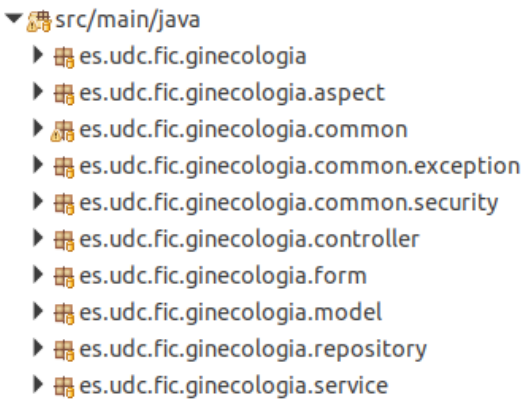


Figura 6.1: Estructura de paquetes Java dentro del proyecto

Los paquetes creados para el proyecto, por lo tanto, son los siguientes:

- **es.udc.fic.ginecologia** -> Es el paquete principal del proyecto, en el cual se guardan las clases necesarias para arrancar la aplicación web junto con el resto de módulos que usa.

  - **es.udc.fic.ginecologia.aspect** -> Es el paquete en el cual se contienen todas las clases necesarias para implementar programación orientada a aspectos, la cual se usa para implementar logging dentro de la capa de servicios, pudiendo registrar cuándo se ha ejecutado una funcionalidad de la capa de servicios y el resultado en caso de haber tenido éxito o saltado un error en el proceso. Un ejemplo de cómo se usaría la programación orientada a aspectos sería el siguiente:

```
@Before("execution(* es.udc.fic.ginecologia..service..*(..))")
  public void logBefore(JoinPoint joinPoint) {
      logger.info("Log before in: " +
  joinPoint.getSignature().getName());
  }

@AfterReturning(pointcut = "execution(*
  es.udc.fic.ginecologia..service..*(..))", returning = "result")
  public void logAfterReturning(JoinPoint joinPoint, Object
  result) {
      logger.info("logAfterReturning");
      logger.info("Log after in: " +
  joinPoint.getSignature().getName());

      String resultToString = result == null ? "null" :
  result.toString();

      logger.info("- And value returned is: " + resultToString);
  }

```

  - **es.udc.fic.ginecologia.common** -> Paquete que contiene clases con utilidades de uso más general para otras clases dentro del proyecto, como por ejemplo comparadores de fechas. Dentro también se incluye otros 2 paquetes, guardando clases que lanzan excepciones y para comprobar permisos de usuarios dento de la aplicación respectivamente.

    * **es.udc.fic.ginecologia.common.exception** -> Paquete que contiene clases usadas dentro de la capa de servicios para la ejecución de excepciones ante determinadas situaciones.

    - **es.udc.fic.ginecologia.common.security** -> Paquete que contiene clases usadas para el control de permisos dentro de las funcionalidades y contenidos ofrecidos por la aplicación web, incluyendo también ”handlers” para controlar y registrar peticiones que se realizan por parte de los usuarios.

- **es.udc.fic.ginecologia.controller** -> Paquete que contiene clases encargadas de recibir peticiones por parte de los usuarios y devolverles la respuesta necesaria según requiera la situación, renderizando por ejemplo una plantilla web o posibilitando descargar un archivo. Un ejemplo de controlador sería el siguiente:

```
@Controller
  public class UserController {
    @Autowired
    UserService userService;

    @Autowired
    SpecialityService specialityService;

    @Autowired
    PermissionChecker permissionChecker;

    // Login form
    @GetMapping("/login")
    public String loginPage() {
      return "login";
    }

    // Login error
    @GetMapping("/login-error")
    public String loginErrorPage(Model model) {
      model.addAttribute("loginError", true);
      return "login";
    }
  }

```

- **es.udc.fic.ginecologia.form** -> Paquete donde se guardan clases para generar los campos de cada uno de los formularios usados en la parte cliente, con el fin de que cuando se lance una petición desde un formulario por parte de un usuario, se lean correctamente los datos recibidos para operar con ellos, como por ejemplo actualizar usuarios, registrar pacientes, guardar recetas, etc. Un ejemplo de cómo sería una clase de este tipo es el siguiente:

```
1         public class LoginForm {
2           @NotBlank
3             @Size(min=3, max = 60)
4             private String username;
5
6             @NotBlank
7             @Size(min = 6, max = 40)
8             private String password;
9
10            public String getUsername() {
11                return username;
12            }
13
14            public void setUsername(String username) {
15                this.username = username;
16            }
17
18            public String getPassword() {
19                return password;
20            }
21
22            public void setPassword(String password) {
23                this.password = password;
24            }
25        }
26
```

- **es.udc.fic.ginecologia.model** -> Paquete donde se guardan las clases persistentes de la aplicación, las cuales mapean las tablas de la BBDD usadas por ella para manejar adecuadamente el tipo de los datos entre Java y MySQL.
- **es.udc.fic.ginecologia.repository** -> Paquete que guarda las interfaces y clases DAO, las cuales son usadas para agilizar las consultas a la BBDD a través de las clases persistentes mencionadas anteriormente. Dichas interfaces proporcionan solicitudes CRUD (Create, Read, Update, Delete) ya montadas para interactuar con la BBDD, pudiendo añadir consultas más específicas a través de convenciones de nombrado, mediante anotaciones con JPQL o mediante clases específicas en las cuales se implementa un método indicado en la interfaz, posibilitando por ejemplo hacer consultas dinámicas mediante el uso de palabras clave. Un ejemplo de implementación de una interfaz DAO sería el siguiente:

```
1       public interface LogLineDao extends CrudRepository<LogLine,
    Integer> {
2           public Iterable<LogLine> findAllByOrderByTimestampDesc();
3
4           @Query("SELECT l FROM LogLine l WHERE l.level=?1 AND
    CONVERT(l.timestamp, DATETIME) BETWEEN ?2 AND ?3 ORDER BY
    l.timestamp DESC")
5           public Iterable<LogLine>
    findByLevelAndTimestampOrderByTimestampDesc(LogLevel level,
    LocalDateTime date1,
6               LocalDateTime date2);
7
8           @Query("SELECT l FROM LogLine l WHERE
    CONVERT(l.timestamp, DATETIME) BETWEEN ?1 AND ?2 ORDER BY
    l.timestamp DESC")
9           public Iterable<LogLine>
    findByTimestampRangeOrderByTimestampDesc(LocalDateTime date1,
    LocalDateTime date2);
10      }
11
```

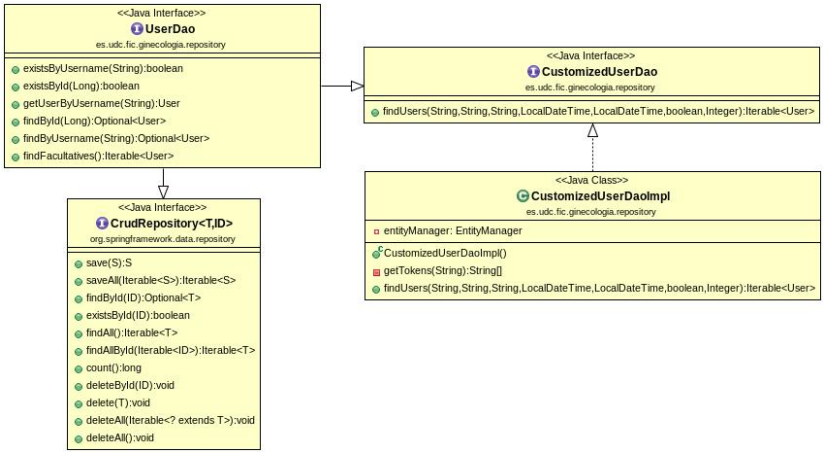


Figura 6.2: Diagrama UML del DAO para la gestión de usuarios

– **es.udc.fic.ginecologia.service** -> Paquete encargado de guardar las clases que encapsulan los servicios y funcionalidades ofrecidas por la aplicación web a los usuarios, ayudándose para ello de los DAOs del paquete anteriormente mencionado. Dentro de dicho paquete también se guardan clases para realizar pruebas tanto unitarias como de integración sobre las coases encargadas de ofrecer las funcionalidades, ya que son unitarias porque se está probando por separado cada funcionalidad ante distintos casos y de integración porque las pruebas se están

realizando sobre una memoria común, en este caso la BBDD.

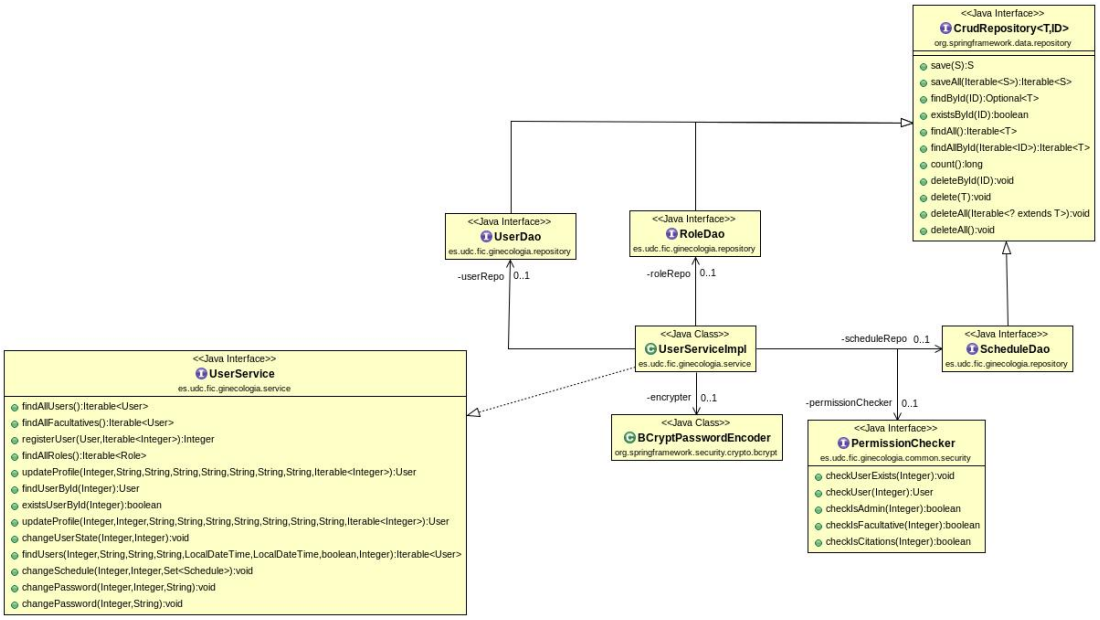


Figura 6.3: Diagrama UML del servicio para la gestión de usuarios

Cabe destacar que los paquetes listados anteriormente se han mostrado de forma jerárquica, ubicándose dentro del proyecto Maven en el directorio ”src/main/java”, mientras que las clases para la realización de las pruebas de la capa de servicios se ubican en el directorio ”src/test/java”.

6.2 Ficheros de recursos

Dentro del proyecto, se han tenido que añadir múltiples ficheros de recursos para distintos fines, tales como por ejemplo configuración, funcionalidades del lado cliente, logging, etc., ubicándolos dentro de una carpeta del proyecto Maven denominada ”src/main/resources” o en el caso de pruebas unitarias en ”src/test/resources”.

Los ficheros de recursos que se han añadido para el proyecto son los siguientes:

- **application.properties** -> Fichero en formato .properties en el cual se indican aspectos principales que se configuran en la aplicación. Los aspectos que se configuran en este fichero es principalmente el nivel general de logging de la aplicación, la fuente de la BBDD para el entorno de desarrollo, configuración para el uso de Hibernate y subida/bajada de ficheros en diferentes formatos que se guardan en la BBDD sean imágenes o ficheros en PDF. En el caso de ”src/test/resources”, sólo se ha añadido un archivo

”application.resources” para que el entorno de pruebas use una BBDD aparte de la del entorno de desarrollo o la de producción.

- **log4j2.xml** -> Fichero en formato .xml en el cual se configura Log4j2, que es la herramienta que se usa para manejar los registros de acciones dentro de la aplicación, configurándolo de forma que opere de una forma u otra según el contexto en el cual se esté ejecutando la aplicación. Los registros se muestran tanto en una tabla almacenada dentro de la BBDD como en un fichero e impresos en terminal dependiendo del caso, limitando el nivel de los registros que se muestran ya que en el caso de la BBDD a pesar de que es más cómodo acceder a los registros es posible correr el riesgo de que se acabe el espacio en disco para la BBDD.

- **messages.properties** -> Fichero en formato .properties en el cual se guarda la traducción de todos los textos para las plantillas HTML usadas dentro del proyecto para la parte cliente, traduciéndolos del inglés al español.

En cuanto a las carpetas de recursos, se han añadido las siguientes:

- **sql** -> Carpeta donde se almacenan scripts en formato SQL para inicializar las tablas de la BBDD para tener datos ya preparados para ser usados tanto dentro del entorno de desarrollo como en producción.

- **static** -> Carpeta donde se almacenan recursos utilizados por las plantillas del lado cliente. En ella se guardan otros ficheros agrupados en carpetas de la siguiente forma:

  - **assets** -> Recursos utilizados por las plantillas HTML y ficheros JavaScript propios, agrupándose de la siguiente manera:

    * **demo** -> Demos de librerías en JavaScript para mostrar en un ejemplo de uso de determinadas librerías o extensiones de Javascript o JQuery.
    * **img** -> Imágenes estáticas usadas en algunas plantillas en concreto para determinados fines, como un logo predeterminado para la aplicación web o mensajes de error.
    * **plugins** -> Extensiones de JavaScript o de JQuery que ha sido necesario descargar para su uso.

  - **css** -> Ficheros en formato .css encargados de estilizar las plantillas HTML que se usan.

  - **js** -> Ficheros en formato .js que se encargan de añadir interacción y funcionalidad a las plantillas en HTML usando JQuery o JavaScript nativo según sea necesario.

- **templates** -> Carpeta donde se guardan las plantillas en formato HTML que se usan dentro del proyecto, agrupándolas según la funcionalidad que cumplan dentro de la aplicación web.

6.3 Otros ficheros

Otros ficheros que se han creado para la configuración del proyecto son los siguientes:

- **Dockerfiles** -> Son ficheros Dockerfile para poder configurar los contenedores que se usan dentro del entorno de producción de la aplicación, en concreto para el contenedor de BBDD y el de la propia aplicación.
- **.gitattributes** -> Fichero de configuración para el repositorio de Git en el cual se indica que ficheros se deben tratar como ficheros binarios y que ficheros se deben tratar como ficheros de texto.
- **.gitignore** -> Fichero de configuración para el repositorio de Git en el cual se indica que ficheros y carpetas se deben ignorar a la hora de añadir nuevos cambios al repositorio. De esta forma, se evita añadir contenidos innecesarios que otras personas que participen en el proyecto no vayan a usar.
- **README.md** -> Fichero que indica información y un resumen sobre el proyecto.
- **docker-compose.yml** -> Fichero en formato YAML que se encarga de crear los contenedores de Docker necesarios para tener operativo el proyecto dentro del entorno en el cual se ha desplegado la aplicación.
- **docker-compose-test.yml** -> Fichero en formato YAML que se encarga de crear los contenedores de Docker necesarios para tener operativo el proyecto dentro de la máquina virtual creada para probar el entorno de Docker.

Capítulo 7

Conclusiones

Los sistemas de gestión de pacientes son una parte fundamental en la práctica asistencial diaria, ya que permiten a través de un sistema complejo poder manejar la información de los pacientes y compartirla entre los especialistas que lo usan, pudiendo manejar adecuadamente el control de la información que se maneja. Con el desarrollo de esta aplicación web, es posible no sólo controlar dicha información sino poder compartir opiniones entre especialistas usuarios de la aplicación, especializándola concretamente en el área de la ginecología y siendo desarrollada a través del uso de herramientas de código abierto.

El proyecto ha sido una oportunidad por parte del alumno para aprender a manejar herramientas de código abierto y aplicarlas a un proyecto del mundo real, usando además una metodología de desarrollo para la organización del trabajo aplicado a éste a través de Scrum, una de las metodologías más usadas actualmente en el mundo del desarrollo software, aplicándose también a otros ámbitos.

7.1 Trabajo realizado

El trabajo que se ha realizado ha permitido crear una aplicación web con varias funcionalidades útiles para la gestión de la información de las pacientes, las cuales son:

- Aplicación adaptada a la Ley de protección de datos, aplicando los puntos de la normativa que influyen en el caso de el seguimiento de pacientes en una consulta médica de ginecología (ver Anexo **??**).

- Generación de copias de seguridad y restauración de la BBDD desde la propia aplicación, pudiendo modificar la copia de seguridad en caso de ser necesario a la hora de realizar la restauración.

- Control del acceso a las diferentes partes de la aplicación web, pudiendo filtrar por fecha y hora, estado de la línea de registro y palabras clave, pudiendo generar un informe de acceso a partir de ellos.

- Gestión de pacientes registradas dentro del sistema, pudiendo ver una lista de ellas y filtrarla en base a varios campos de la tabla, posibilitando el registro y/o modificación de los datos de éstas indicando desde datos personales hasta antecedentes tanto personales como ginecológicos. Se da también la posibilidad de marcar o desmarcar pacientes que sean de potencial interés para los facultativos o ver un historial de las últimas pacientes que hayan sido atendidas con anterioridad, pudiendo incluso bloquear/desbloquear el acceso de éstas en caso de ser necesario por parte de los administradores del sistema.

- Gestión del curso clínico de las pacientes que acuden a la clínica ginecológica, indicando la actividad realizada junto con la posibilidad de añadir comentarios mediante un editor de texto, realizar un cuestionario a través de preguntas guardadas o preguntas propias para que se guarden más tarde una vez que se guarde la entrada del curso clínico y puedan ser reutilizadas, añadir pruebas complementarias las cuales pueden ser archivos de cualquier tipo sean PDF, imágenes o cualquier otro formato de fichero para ser descargados y visualizados posteriormente y prescribir una o varias recetas para ser descargadas e impresas.

- Generación de informes para cada paciente en base a los datos que se tenga almacenados de cada una de ellas, incluyendo el historial que tengan dentro del curso clínico.

- Gestión de usuarios registrados dentro del propio sistema, pudiendo visualizarlos en una lista en la cual se puede aplicar una serie de filtros de búsqueda sobre ella y crear/-modificar usuarios, añadiendo datos personales, uno o varios roles tanto de administrador como de facultativo y/o de citaciones o incluso horarios y especialidades médicas en el caso de usuarios con el rol de facultativo. También es posible bloquear/desbloquear usuarios del sistema en caso de ser necesario.

- Gestión de especialidades registradas dentro del sistema para poder ser asignadas/quitadas a los usuarios que tengan el rol de facultativo, pudiendo desde registrarlas, aplicar filtros de búsqueda, modificarlas o incluso bloquearlas o desbloquearlas en caso de ser necesario.

- Gestión de medicamentos registrados dentro del sistema para poder ser prescritos en las recetas para las pacientes por parte de los usuarios que tengan el rol de facultativo, pudiendo desde registrarlos, aplicar filtros de búsqueda, modificarlos o incluso bloquearlos o desbloquearlos en caso de ser necesario.

- Gestión de pruebas diagnósticas registradas dentro del sistema para poder ser añadidas en las entradas del curso clínico, pudiendo desde registrarlas, aplicar filtros de búsqueda, modificarlas o incluso bloquearlas o desbloquearlas en caso de ser necesario.

- Gestión de métodos anticonceptivos registrados dentro del sistema para poder ser añadidos o quitados dentro de los antecedentes ginecológicos de las pacientes por parte de los usuarios que tengan el rol de facultativo, pudiendo desde registrarlos, aplicar filtros de búsqueda, modificarlos o incluso bloquearlos o desbloquearlos en caso de ser necesario.

- Gestión de agendas para los usuarios con el rol de facultativo, pudiendo desde añadir, modificar o cancelar citas en caso de llegar a ser necesario por parte de los usuarios del sistema que tengan roles de administrador y/o de citaciones. Dichas citas aparecen como entradas de calendario en cada una de las agendas de los facultativos, pudiendo a partir de ellas atender las citas de las pacientes para posteriormente añadir la entrada del curso clínico correspondiente.

- Buzón de mensajería en el cual se puede leer los mensajes privados que hayan sido mandados y/o recibidos por parte del propio usuario del sistema, pudiendo escribir mensajes de respuesta o generar interconsultas hacia otros usuarios del sistema en caso de ser facultativos para poder consultar dudas a partir de entradas del curso clínico de las pacientes.

- Foro de tareas comunes donde se puede consultar las tareas que se tengan en común con otros usuarios, pudiendo escribir mensajes grupales dentro de ellas para que sean leídos por los usuarios de dicha tarea común o crear otras nuevas en caso de ser necesario, indicando un nombre y una descripción de éstas.

- Buzón de avisos común para todos los usuarios del sistema en el cual cada uno de ellos puede escribir cualquier aviso importante pra que sea leído por el resto, guardando un historial de los avisos lanzados hasta el momento.

7.2 Trabajo futuro

Como trabajo futuro, se abre la posibilidad de usar una aplicación web de código abierto dentro de clínicas ginecológicas, pudiendo incluso adaptarla para cualquier otra especialidad médica existente, posibilitando también en consecuencia ampliar las funcionalidades de ésta o ser adaptadas a necesidades particulares de otras clínicas, ya que al ser de código abierto se puede modificar o ser usada como base para el desarrollo de otras aplicaciones de la misma

índole.

Otras posibilidades de trabajo futuro relacionadas con la aplicación sería añadir como nuevas funcionalidades la inclusión de poder ver directamente archivos, vídeos o imágenes de cualquier índole, ya que sólo es posible descargarlos para verlos en local o también añadir otras funcionalidades que vayan en otras aplicaciones del mercado que puedan ser interesantes.

Otra posible funcionalidad de interés sería la posibilidad de interactuar con el uso de ecógrafos, comunicándose con la propia aplicación mediante el uso del protocolo DICOM[64][65]. El protocolo DICOM es un estándar de transmisión de imágenes médicas y datos entre hardware que se usa en el ámbito de la medicina, usándose en casos de visualización, almacenamiento, impresión y transmisión de imágenes, pudiendo integrarse tanto en escáneres, servidores, estaciones de trabajo, etc., las cuales deben tener una declaración de conformidad DICOM que establece claramente las clases DICOM que soportan.

Apéndices

Apéndice A

Historias de usuario detalladas

En este apartado, se mostrarán cada una de las historias de usuario las cuales representan cada una de las funcionalidades disponibles dentro de la aplicación web, indicando aspectos como los actores que pueden realizar la acción descrita, los eventos a realizar y las condiciones que se deben cumplir.

1. Login dentro del sistema
    - Descripción: El usuario puede entrar dentro del sistema.
    - Actores: facultativo, administrador, citaciones.
    - Flujo de eventos:
        1) El usuario introduce el usuario y la contraseña.
        2) Se comprueban las credenciales.
    - Precondiciones: el usuario debe tener un perfil registrado dentro del sistema.
    - Poscondiciones: el usuario tiene acceso a las funcionalidades que tenga disponibles en base a su rol.
2. Loguearse fuera del sistema
    - Descripción: El usuario puede salir del sistema.
    - Actores: administrador, facultativo, citaciones.
    - Flujo de eventos:
        1) El usuario selecciona la opción de salir del sistema.
        2) El usuario vuelve a la pantalla de login.
    - Precondiciones: El usuario se ha logueado dentro del sistema.
    - Poscondiciones: El usuario sale del sistema y es redirigido a la página de login.

3. Ver lista de usuarios

    - Descripción: El usuario puede ver la lista de usuarios registrados dentro del sistema.
    - Actores: administrador.
    - Flujo de eventos:
        1) El usuario abre el menú de opciones de la aplicación.
        2) El usuario pulsa sobre la opción de ver usuarios registrados.
    - Precondiciones: el usuario debe estar registrado como administrador.
    - Poscondiciones: se muestra la ista de perfiles registrados dentro del sistema.

4. Dar de alta usuario

    - Descripción: El usuario puede registrar un perfil que pueda ser usado para un usuario dentro del sistema.
    - Actores: administrador.
    - Flujo de eventos:
        1) El administrador entra en la página de registro de usuarios.
        2) El usuario introduce todos los datos del usuario que se consideren necesarios.
    - Precondiciones: el usuario debe tener rol de administrador del sistema.
    - Poscondiciones: queda registrado un perfil con los datos que se han asociado en el registro.

5. Actualizar usuario

    - Descripción: El usuario puede actualizar un perfil que haya sido registrado anteriormente dentro del sistema.
    - Actores: administrador.
    - Flujo de eventos:
        1) El usuario busca el usuario en la lista de usuarios disponibles.
        2) El usuario pulsa sobre la opción de actualizar el usuario.
        3) El usuario modifica los datos del usuario que se consideren pertinentes.
        4) El usuario pulsa sobre el botón de actualizar.
    - Precondiciones:
        (a) El usuario debe tener rol de administrador del sistema.
        (b) El perfil debe estar registrado en el sistema.

- Poscondiciones: el perfil debe estar actualizado con las modificaciones que se han realizado.

6. Dar de baja usuario

   - Descripción: El usuario puede dar de baja un perfil que haya sido registrado anteriormente dentro del sistema.
   - Actores: administrador.
   - Flujo de eventos:
     1) El usuario busca el usuario en la lista de usuarios disponibles.
     2) El usuario pulsa sobre la opción de marcar usuario como inactivo.
     3) Se muestra el perfil marcado como inactivo en el sistema.
   - Precondiciones:
     (a) El usuario debe tener rol de administrador del sistema.
     (b) El perfil debe estar registrado en el sistema.
   - Poscondiciones: el perfil debe estar marcado como inactivo.

7. Buscar usuarios

   - Descripción: El usuario puede buscar perfiles que ya estén registrados dentro del sistema.
   - Actores: administrador.
   - Flujo de eventos:
     1) El usuario entra en la ventana de lista de usuarios.
     2) El usuario utiliza los filtros disponibles para buscar en base a datos como el login, el rol o la fecha de alta.
     3) Se muestran los datos que se están buscando.
   - Precondiciones: El usuario debe tener rol de administrador del sistema.
   - Poscondiciones: La lista de perfiles que se muestra debe coincidir con los criterios de búsqueda aplicados.

8. Ver jornada laboral

   - Descripción: El usuario puede ver el horario de un perfil de usuario en concreto del sistema.
   - Actores: administrador.
   - Flujo de eventos:

1) El usuario escoge un perfil en concreto que esté registrado en el sistema.
2) Al entrar en los datos del perfil, el usuario ve una sección dentro de la página donde se observa las jornadas laborales del empleado.

- Precondiciones:
  (a) El usuario debe tener rol de administrador del sistema.
  (b) El perfil debe estar registrado en el sistema.
- Poscondiciones: el usuario ve los dias laborables que tiene el usuario a lo largo de la semana.

9. Añadir jornada laboral

- Descripción: El usuario puede añadir una jornada laboral para el usuario del sistema que se esté modificando.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario escoge un perfil en concreto que esté registrado en el sistema.
  2) Al entrar en los datos del perfil, el usuario ve una sección dentro de la página donde se observa las jornadas laborales del empleado.
  3) Dentro de la sección de horarios, el usuario marca el día de la semana que quiere habilitar para la jornada laboral.
  4) El usuario indica la hora de inicio y la hora de fin de la jornada laboral.
  5) El usuario pulsa sobre el botón para guardar la jornada laboral.
  6) El usuario ve los cambios guardados en el horario.
- Precondiciones:
  (a) El usuario debe tener rol de administrador del sistema.
  (b) El perfil debe estar registrado en el sistema.
- Poscondiciones: se ve la jornada añadida al usuario.

10. Modificar jornada laboral

- Descripción: El usuario puede modificar una jornada laboral para el usuario del sistema que se esté modificando.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario escoge un perfil en concreto que esté registrado en el sistema.

2) Al entrar en los datos del perfil, el usuario ve una sección dentro de la página donde se observa las jornadas laborales del empleado.
3) El usuario modifica uno de los campos de un día de la semana en concreto.
4) El usuario pulsa sobre el botón para guardar cambios en el horario.
5) Se actualiza la jornada laboral del perfil de usuario.

- Precondiciones:
  (a) El usuario debe tener rol de administrador del sistema.
  (b) El perfil debe estar registrado en el sistema.
  (c) El usuario debe tener al menos una jornada laboral registrada.
- Poscondiciones: se ven los cambios en la jornada laboral del usuario.

11. Quitar jornada laboral

- Descripción: El usuario puede modificar una jornada laboral para el usuario del sistema que se esté modificando.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario escoge un perfil en concreto que esté registrado en el sistema. 12
  2) Al entrar en los datos del perfil, el usuario ve una sección dentro de la página donde se observa las jornadas laborales del empleado.
  3) El usuario deshabilita uno de los días de la semana en concreto.
  4) El usuario pulsa sobre el botón para guardar cambios en el horario.
  5) Se actualiza la jornada laboral del perfil de usuario.
- Precondiciones:
  (a) El usuario debe tener rol de administrador del sistema.
  (b) El perfil debe estar registrado en el sistema.
  (c) El usuario debe tener al menos una jornada laboral registrada.
- Poscondiciones: se ven los cambios en la jornada laboral del usuario.

12. Ver especialidades

- Descripción: El usuario puede ver las especialidades disponibles para asignar a los facultativos.
- Actores: administrador.
- Flujo de eventos:

1) El usuario escoge la opción en el menú desplegable “Gestionar especialidades”.
2) Al pulsar sobre la opción, el usuario ve la lista de especialidades registradas hasta el momento.

- Precondiciones:
  (a) El usuario debe tener rol de administrador del sistema.
- Poscondiciones: se muestra una lista con las especialidades disponibles.

13. Añadir especialidad

- Descripción: El usuario puede añadir una especialidad para asignar a los facultativos.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario escoge la opción en el menú desplegable “Gestionar especialidades”.
  2) El usuario pulsa sobre el botón “Añadir especialidad”.
  3) Se despliega un formulario donde el usuario cubre los datos necesarios para registrar la especialidad.
  4) El usuario pulsa sobre el botón de “Guardar especialidad”.
- Precondiciones:
  (a) El usuario debe tener rol de administrador del sistema.
- Poscondiciones: En la lista de especialidades se muestra la especialidad guardada.

14. Modificar especialidad

- Descripción: El usuario puede modificar una especialidad para asignar a los facultativos.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario escoge la opción en el menú desplegable “Gestionar especialidades”.
  2) El usuario pulsa sobre la opción de modificar de alguna de las especialidades.
  3) El usuario cambia los datos de la especialidad que considere necesarios.
  4) El usuario guarda los cambios realizados.
  5) Se muestra la especialidad con los datos modificados.

- Precondiciones:
  (a) El usuario debe tener rol de administrador del sistema.
  (b) Debe haber por lo menos una especialidad registrada dentro del sistema.
- Poscondiciones: En la lista de especialidades se muestra la especialidad modificada.

15. Dar de baja especialidad

- Descripción: El usuario puede quitar una especialidad para asignar a los facultativos.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario escoge la opción en el menú desplegable "Gestionar especialidades".
  2) El usuario pulsa sobre el botón de dar de baja especialidad.
  3) Se muestra la especialidad en la lista como inactiva.
- Precondiciones:
  (a) El usuario debe tener rol de administrador del sistema.
  (b) Debe haber por lo menos una especialidad registrada dentro del sistema.
- Poscondiciones: En la lista de especialidades se muestra la especialidad dada de baja.

16. Buscar especialidades

- Descripción: El usuario puede buscar en la lista de especialidades en base a unos criterios de búsqueda determinados.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario escoge la opción en el menú desplegable "Gestionar especialidades".
  2) El usuario pulsa sobre el botón para desplegar los criterios de búsqueda.
  3) El usuario cubre los campos sobre los que quiere buscar.
  4) El usuario pulsa sobre el botón de "Buscar".
  5) Se muestran las especialidades que cumplen con los criterios de búsqueda.
- Precondiciones:
  (a) El usuario debe tener rol de administrador del sistema.

(b) Debe haber por lo menos una especialidad registrada dentro del sistema.

- Poscondiciones: Se muestran las especialidades que cumplan con los criterios de búsqueda.

17. Asignar especialidad a facultativo

- Descripción: El usuario puede asignar una o varias especialidades a un facultativo.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de modificar usuario para uno de los usuarios registrados dentro del sistema.
  2) Si el usuario tiene el rol de facultativo, se mostrará una sección donde están todas las especialidades de las que dispone el usuario con dicho rol.
  3) En el lado izquierdo se muestran las especialidades que no tiene asignadas el usuario, mientras que en el lado derecho se muestran las especialidades ya asignadas.
  4) Se seleccionan las especialidades que se quieren asignar.
  5) Se pulsa sobre el botón de transferir especialidades.
  6) Se pulsa sobre el botón de guardar cambios para las especialidades.
  7) Se muestra al usuario con las especialidades asignadas.
- Precondiciones:
  (a) El usuario debe tener rol de administrador del sistema.
  (b) Debe haber por lo menos una especialidad registrada dentro del sistema.
  (c) El usuario a modificar debe tener rol de facultativo.
- Poscondiciones: se deben ver las especialidades asignadas al usuario.

18. Quitar especialidad a facultativo

- Descripción: El usuario puede quitar una o varias especialidades a un facultativo.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de modificar usuario para uno de los usuarios registrados dentro del sistema.
  2) Si el usuario tiene el rol de facultativo, se mostrará una sección donde están todas las especialidades de las que dispone el usuario con dicho rol.

3) En el lado izquierdo se muestran las especialidades que no tiene asignadas el usuario, mientras que en el lado derecho se muestran las especialidades ya asignadas.
4) Se seleccionan las especialidades que se quieren quitar.
5) Se pulsa sobre el botón de quitar especialidades.
6) Se pulsa sobre el botón de guardar cambios para las especialidades.
7) Se muestra al usuario con las especialidades quitadas.

- Precondiciones:
  (a) El usuario debe tener rol de administrador del sistema.
  (b) Debe haber por lo menos una especialidad registrada dentro del sistema.
  (c) El usuario a modificar debe tener rol de facultativo.
  (d) El usuario debe tener especialidades asignadas.
- Poscondiciones: se deben ver las especialidades quitadas al usuario.

19. Ver medicamentos

- Descripción: El usuario puede ver los medicamentos disponibles para recetar a las pacientes.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario elige la opción de gestionar medicamentos dentro del menú desplegable.
  2) El usuario ve los medicamentos que están disponibles para recetar hasta el momento.
- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
- Poscondiciones: se deben ver los medicamentos disponibles hasta el momento.

20. Añadir medicamento

- Descripción: El usuario puede añadir los medicamentos para recetar a las pacientes.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario elige la opción de gestionar medicamentos dentro del menú desplegable.

2) El usuario pulsa sobre el botón “Añadir medicamento”.
3) El usuario cubre los campos necesarios para añadir el medicamento.
4) El usuario pulsa sobre el botón “Añadir”.
5) Se muestra el medicamento en la lista.

- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
- Poscondiciones: Se debe mostrar el nuevo medicamento en la lista.

21. Modificar medicamento

- Descripción: El usuario puede modificar los datos de un medicamento para recetar a las pacientes.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario elige la opción de gestionar medicamentos dentro del menú desplegable.
  2) El usuario pulsa sobre la opción de modificar medicamento en alguno de ellos.
  3) El usuario modifica los datos que considere necesarios.
  4) El usuario pulsa sobre el botón “Modificar”.
  5) Se muestra el medicamento en la lista con los datos modificados.
- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
  (b) Debe haber por lo menos un medicamento registrado en el sistema.
- Poscondiciones: los datos del medicamento se han modificado correctamente.

22. Quitar medicamento

- Descripción: El usuario puede dar de baja un medicamento para recetar a las pacientes.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario elige la opción de gestionar medicamentos dentro del menú desplegable.
  2) El usuario pulsa sobre la opción de quitar medicamento en alguno de ellos.
  3) El medicamento queda marcado como quitado en la lista.

- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
  (b) Debe haber por lo menos un medicamento registrado en el sistema.
- Poscondiciones: el medicamento se ha marcado como quitado correctamente.

23. Buscar medicamentos

- Descripción: El usuario puede buscar medicamentos en base a una serie de criterios de búsqueda.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario elige la opción de gestionar medicamentos dentro del menú desplegable.
  2) El usuario aplica los criterios de búsqueda que considere necesarios.
  3) El usuario pulsa sobre el botón “Buscar”.
  4) Se muestra en la lista los medicamentos que cumplan con los criterios de búsqueda.
- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
- Poscondiciones: se deben mostrar los medicamentos que coincidan con los criterios de búsqueda.

24. Ver pruebas diagnósticas

- Descripción: El usuario puede ver la lista de pruebas diagnósticas disponibles dentro de la clínica.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario elige la opción de gestionar medicamentos dentro del menú desplegable.
  2) El usuario ve la lista de pruebas diagnósticas que están disponibles dentro del sistema.
- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
- Poscondiciones: Se muestra la lista de pruebas diagnósticas disponibles hasta el momento dentro del sistema.

25. Añadir prueba diagnóstica

    - Descripción: El usuario puede añadir una prueba diagnóstica para realizar dentro de la clínica.
    - Actores: administrador.
    - Flujo de eventos:
        1) El usuario elige la opción de gestionar medicamentos dentro del menú desplegable.
        2) El usuario pulsa sobre el botón "Añadir prueba diagnóstica".
        3) El usuario cubre todos los campos que considere necesarios.
        4) El usuario pulsa sobre el botón "Añadir".
        5) El usuario ve la prueba diagnóstica añadida en la lista.
    - Precondiciones:
        (a) El usuario debe tener el rol de administrador.
    - Poscondiciones: Se muestra la prueba diagnóstica añadida dentro de la lista.

26. Modificar prueba diagnóstica

    - Descripción: El usuario puede modificar los datos de una prueba diagnóstica para realizar dentro de la clínica.
    - Actores: administrador.
    - Flujo de eventos:
        1) El usuario elige la opción de gestionar medicamentos dentro del menú desplegable.
        2) El usuario pulsa sobre el botón para modificar sobre una de las pruebas diagnósticas en concreto.
        3) El usuario modifica los datos de la prueba diagnóstica que considere necesarios.
        4) El usuario pulsa sobre el botón de "Modificar".
        5) Se muestran los datos actualizados en la prueba diagnóstica.
    - Precondiciones:
        (a) El usuario debe tener el rol de administrador.
        (b) Debe haber por lo menos una prueba diagnóstica registrada.
    - Poscondiciones: Los datos de la prueba diagnóstica deben de haberse modificado correctamente.

27. Dar de baja prueba diagnóstica

    - Descripción: El usuario puede dar de baja una prueba diagnóstica para realizar dentro de la clínica.
    - Actores: administrador.
    - Flujo de eventos:
        1) El usuario elige la opción de gestionar medicamentos dentro del menú desplegable.
        2) El usuario pulsa sobre el botón para dar de baja la prueba diagnóstica en concreto.
        3) La prueba diagnóstica se muestra como inactiva dentro de la lista.
    - Precondiciones:
        (a) El usuario debe tener el rol de administrador.
        (b) Debe haber por lo menos una prueba diagnóstica registrada.
    - Poscondiciones: la prueba diagnóstica debe de estar marcada como inactiva.

28. Buscar pruebas diagnósticas

    - Descripción: El usuario puede buscar pruebas diagnósticas en base a una serie de criterios de búsqueda.
    - Actores: administrador.
    - Flujo de eventos:
        1) El usuario elige la opción de gestionar medicamentos dentro del menú desplegable.
        2) El usuario aplica los criterios de búsqueda que considere necesarios.
        3) El usuario pulsa sobre el botón “Buscar”.
        4) Se muestra en la lista las pruebas diagnósticas que cumplan con los criterios de búsqueda.
    - Precondiciones:
        (a) El usuario debe tener el rol de administrador.
    - Poscondiciones: se debe mostrar las pruebas diagnósticas que cumplan con los criterios de búsqueda.

29. Ver métodos anticonceptivos

    - Descripción: El usuario puede ver la lista de métodos anticonceptivos registrados en el sistema.

- Actores: administrador.
- Flujo de eventos:
  1) El usuario elige la opción de gestionar métodos anticonceptivos dentro del menú desplegable.
  2) El usuario ve la lista de métodos anticonceptivos que están disponibles dentro del sistema.
- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
- Poscondiciones: se debe mostrar todos los métodos anticonceptivos que se han registrado dentro del sistema.

30. Añadir método anticonceptivo

- Descripción: El usuario puede registrar un método anticonceptivo dentro del sistema.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario elige la opción de gestionar métodos anticonceptivos dentro del menú desplegable.
  2) El usuario pulsa sobre el botón "Añadir prueba diagnóstica".
  3) El usuario cubre todos los campos que considere necesarios.
  4) El usuario pulsa sobre el botón "Añadir".
  5) El usuario ve la prueba diagnóstica añadida en la lista.
- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
- Poscondiciones: el método anticonceptivo debe quedar registrado en el sistema.

31. Modificar método anticonceptivo

- Descripción: El usuario puede ver la lista de métodos anticonceptivos registrados en el sistema.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario elige la opción de gestionar métodos anticonceptivos dentro del menú desplegable.
  2) El usuario pulsa sobre la opción para modificar la prueba diagnóstica.

3) El usuario cubre todos los campos que considere necesarios.
4) El usuario pulsa sobre el botón “Modificar”.

- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
- Poscondiciones: el método anticonceptivo queda modificado correctamente dentro del sistema.

32. Dar de baja método anticonceptivo

- Descripción: El usuario puede dar de baja un método anticonceptivo registrado en el sistema.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario elige la opción de gestionar métodos anticonceptivos dentro del menú desplegable.
  2) El usuario pulsa sobre la opción para dar de baja la prueba diagnóstica.
- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
- Poscondiciones: el método anticonceptivo queda marcado como dado de baja en el sistema.

33. Buscar métodos anticonceptivos

- Descripción: El usuario puede buscar métodos anticonceptivos registrados en el sistema.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario aplica los criterios de búsqueda que considere necesarios.
  2) El usuario pulsa sobre el botón “Buscar”.
- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
- Poscondiciones: Se muestra en la lista los métodos anticonceptivos que cumplan con los criterios de búsqueda.

34. Cambiar nombre y logo

- Descripción: El usuario puede cambiar el nombre y el logo de la aplicación web.

- Actores: administrador.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción para cambiar el nombre y el logo.
  3) El usuario ajusta el nombre y el logo de la empresa.
  4) El usuario pulsa sobre el botón para guardar los cambios.
- Precondiciones:
  (a) El usuario debe estar logueado en el sistema con el rol de administrador.
- Poscondiciones: el nombre y el logo de la empresa son cambiados correctamente.

35. Ver pacientes

- Descripción: El usuario puede ver los pacientes registrados en el sistema hasta el momento.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
- Precondiciones:
  (a) El usuario debe tener el rol de facultativo, de administrador o rol de citaciones.
- Poscondiciones: se deben ver todas las pacientes registradas dentro del sistema.

36. Buscar pacientes

- Descripción: El usuario puede buscar pacientes en base a una serie de criterios de búsqueda.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  3) El usuario cubre los campos de búsqueda dependiendo de los criterios de búsqueda que quiera aplicar.
  4) El usuario pulsa sobre el botón de buscar.
- Precondiciones:
  (a) El usuario debe tener el rol de facultativo, de administrador o rol de citaciones.

- Poscondiciones: se muestran las pacientes que cumplen con los criterios de búsqueda.

37. Ver detalles de paciente

    - Descripción: El usuario puede ver los detalles de una paciente registrada en el sistema.
    - Actores: facultativo, administrador, citaciones.
    - Flujo de eventos:
        1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
        2) El usuario pulsa sobre el número de historia de la paciente, el cual es un enlace a los detalles de la paciente.
    - Precondiciones:
        (a) El usuario debe tener el rol de facultativo, de administrador o rol de citaciones.
        (b) Debe haber por lo menos una paciente registrada en el sistema.
    - Poscondiciones: se muestra una pantalla donde se ve en detalle toda la información necesaria relacionada con la paciente.

38. Ver antecedentes

    - Descripción: El usuario puede ver los antecedentes relacionados con la paciente que se hayan anotado con anterioridad.
    - Actores: facultativo, administrador, citaciones.
    - Flujo de eventos:
        1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
        2) El usuario pulsa sobre el número de historia de la paciente, el cual es un enlace a los detalles de la paciente.
        3) El usuario pulsa sobre el botón desplegable “Ver antecedentes”, donde se abre un desplegable con dicha información.
    - Precondiciones:
        (a) El usuario debe tener el rol de facultativo, de administrador o rol de citaciones.
        (b) La paciente debe estar registrada en el sistema.
    - Poscondiciones: se muestran todos los antecedentes añadidos hasta el momento para la paciente con toda la información necesaria.

39. Consultar curso clínico

- Descripción: El usuario puede ver el historial de visitas de la paciente.
- Actores: facultativo.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  2) El usuario pulsa sobre el número de historia de la paciente, el cual es un enlace a los detalles de la paciente.
  3) El usuario pulsa sobre el desplegable para ver el curso clínico.
- Precondiciones:
  (a) El usuario debe tener el rol de facultativo.
  (b) La paciente debe estar registrada en el sistema.
- Poscondiciones: se muestran todas las visitas del curso clínico ordenadas de forma descendente por recencia.

40. Consultar entrada del curso clínico

- Descripción: El usuario puede ver en detalle una de las entradas del curso clínico de la paciente.
- Actores: facultativo.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  2) El usuario pulsa sobre el número de historia de la paciente, el cual es un enlace a los detalles de la paciente.
  3) El usuario pulsa sobre el desplegable para ver el curso clínico.
  4) El usuario pulsa sobre una de las entradas del diario clínico.
- Precondiciones:
  (a) El usuario debe tener el rol de facultativo.
  (b) La paciente debe estar registrada en el sistema.
- Poscondiciones: se muestran todos los detalles de la entrada del curso clínico.

41. Ver cuestionario

- Descripción: El usuario puede ver el cuestionario realizado en la entrada del curso clínico.
- Actores: facultativo.
- Flujo de eventos:

1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
2) El usuario pulsa sobre el número de historia de la paciente, el cual es un enlace a los detalles de la paciente.
3) El usuario pulsa sobre el desplegable para ver el curso clínico.
4) El usuario pulsa sobre una de las entradas del diario clínico.
5) El usuario pulsa sobre el desplegable para ver el cuestionario.

- Precondiciones:
  (a) El usuario debe tener el rol de facultativo.
  (b) La paciente debe estar registrada en el sistema.
  (c) La paciente debe tener al menos una entrada dentro del curso clínico.
- Poscondiciones: se muestra todas las preguntas realizadas dentro de la entrada del curso clínico.

42. Ver pruebas complementarias de la entrada del curso clínico

- Descripción: El usuario puede ver las pruebas complementarias añadidas en la entrada del curso clínico.
- Actores: facultativo.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  2) El usuario pulsa sobre el número de historia de la paciente, el cual es un enlace a los detalles de la paciente.
  3) El usuario pulsa sobre el desplegable para ver el curso clínico.
  4) El usuario pulsa sobre una de las entradas del diario clínico.
  5) El usuario pulsa sobre el desplegable para ver las pruebas complementarias.
- Precondiciones:
  (a) El usuario debe tener el rol de facultativo.
  (b) La paciente debe estar registrada en el sistema.
  (c) La paciente debe tener al menos una entrada dentro del curso clínico.
- Poscondiciones: se muestran todas las pruebas complementarias añadidas a la entrada del curso clínico.

43. Añadir paciente

- Descripción: El usuario puede ver las pacientes registradas en el sistema hasta el momento.

- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  2) El usuario pulsa sobre el botón "Añadir paciente".
  3) El usuario cubre los datos del paciente que se consideren necesarios, tales como filiación o antecedentes.
  4) Se muestra la paciente con los datos cubiertos dentro de la lista.
- Precondiciones:
  (a) El usuario debe tener el rol de facultativo, de administrador o tener el rol de citaciones.
- Poscondiciones: al registrarse a la paciente, se pasa al caso de uso "Añadir entrada al diario clínico".

44. Modificar paciente

- Descripción: El usuario puede modificar los datos a los que se tengan acceso de una paciente en concreto.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  2) El usuario pulsa sobre el botón para modificar paciente de la paciente que se quiera modificar.
  3) El usuario modifica los datos del paciente a los cuales pueda acceder y considere necesario modificar.
  4) El usuario pulsa sobre el botón de modificar.
  5) Se muestran los datos de la paciente con los cambios realizados.
- Precondiciones:
  (a) El usuario debe tener el rol de facultativo, de administrador o tener el rol de citaciones.
  (b) Debe haber por lo menos una paciente registrada en el sistema.
- Poscondiciones: los datos de la paciente se han modificado con éxito.

45. Dar de baja paciente

- Descripción: El usuario puede dar de alta o de baja los datos de una paciente.
- Actores: administrador.

- Flujo de eventos:
  1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  2) El usuario pulsa sobre el botón para dar de alta o de baja los datos de una paciente en concreto.
- Precondiciones:
  (a) El usuario debe tener el rol de administrador.
  (b) Debe haber por lo menos una paciente registrada en el sistema.
- Poscondiciones: la paciente se ha dado de alta o de baja con éxito.

46. Añadir entrada a curso clínico

- Descripción: El usuario puede añadir una entrada al curso clínico de la paciente.
- Actores: facultativo.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  2) El usuario pulsa sobre el botón para modificar paciente de la paciente que se quiera modificar.
  3) El usuario pulsa sobre el desplegable del diario clínico, mostrando un editor de texto para añadir información sobre la entrada del diario adjuntando debajo las entradas del diario ordenadas descendentemente por recencia, desplegando la última visita.
  4) El usuario cubre la información de la visita que vaya obteniendo de la paciente.
  5) El usuario pulsa sobre el botón para añadir entrada.
- Precondiciones:
  (a) El usuario debe tener el rol de facultativo.
  (b) Debe haber por lo menos una paciente registrada en el sistema.
- Poscondiciones: la entrada se debe haber añadido correctamente al curso clínico.

47. Añadir pregunta en cuestionario

- Descripción: El usuario puede añadir una pregunta que se haya contestado por parte de la paciente dentro de la cita asociada a la entrada del curso clínico.
- Actores: facultativo.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.

2) El usuario pulsa sobre el botón para modificar paciente de la paciente que se quiera modificar.
3) El usuario pulsa sobre el desplegable del diario clínico, mostrando un editor de texto para añadir información sobre la entrada del diario adjuntando debajo las entradas del diario ordenadas descendentemente por recencia, desplegando la última visita.
4) El usuario pulsa sobre la pestaña desplegable dedicada al cuestionario.
5) El usuario pulsa sobre el botón "Añadir pregunta".
6) El usuario escoge una de las preguntas disponibles y cubre la respuesta.
7) El usuario pulsa sobre el botón de añadir.
8) Una vez que el usuario ha terminado la visita, pulsa sobre el botón de guardar.

- Precondiciones:
  (a) El usuario debe tener el rol de facultativo.
  (b) Debe haber por lo menos una paciente registrada en el sistema.
- Poscondiciones: junto a la entrada del curso clínico se debe haber añadido el cuestionario realizado.

48. Añadir prueba complementaria a entrada del curso clínico

- Descripción: El usuario puede añadir pruebas complementarias a la entrada del curso clínico.
- Actores: facultativo.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  2) El usuario pulsa sobre el botón para modificar paciente de la paciente que se quiera modificar.
  3) El usuario pulsa sobre el desplegable del diario clínico, mostrando un editor de texto para añadir información sobre la entrada del diario adjuntando debajo las entradas del diario ordenadas descendentemente por recencia, desplegando la última visita.
  4) El usuario pulsa sobre la pestaña desplegable dedicada a las pruebas complementarias.
  5) El usuario pulsa sobre el botón "Añadir prueba complementaria".
  6) El usuario indica la prueba complementaria a añadir en base a las pruebas clínicas disponibles en el sistema.
  7) El usuario adjunta el archivo asociado a la prueba complementaria.

8) Una vez que el usuario ha terminado la visita, pulsa sobre el botón de guardar.

- Precondiciones:
  (a) El usuario debe tener el rol de facultativo.
  (b) Debe haber por lo menos una paciente registrada en el sistema.
  (c) Debe haber por lo menos una prueba complementaria registrada dentro del sistema.
- Poscondiciones: la prueba complementaria debe aparecer registrada en la entrada del curso clínico, pudiendo descargarse.

49. Prescribir receta

- Descripción: El usuario puede prescribir una receta para la paciente asociándola a la cita que se esté dando en ese momento.
- Actores: facultativo.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  2) El usuario pulsa sobre el botón para modificar paciente de la paciente que se quiera modificar.
  3) El usuario pulsa sobre el desplegable del diario clínico, mostrando un editor de texto para añadir información sobre la entrada del diario adjuntando debajo las entradas del diario ordenadas descendentemente por recencia, desplegando la última visita.
  4) El usuario pulsa sobre la pestaña para prescribir receta en la cita.
  5) El usuario selecciona los medicamentos a recetar para la paciente junto con las aclaraciones que se consideren necesarias.
  6) El usuario pulsa sobre el botón para prescribir la receta.
- Precondiciones:
  (a) El usuario debe tener el rol de facultativo.
  (b) Debe haber por lo menos una paciente registrada en el sistema.
  (c) Debe haber por lo menos un medicamento registrado en el sistema.
- Poscondiciones: una vez prescrita la receta, se puede generar un PDF que contenga el logo y el nombre de la clínica junto con toda la información relacionada con la receta, es decir, el n.º de colegiado del facultativo, el n.º de historia de la paciente en el SERGAS y el listado de medicamentos recetados junto con las aclaraciones añadidas a la receta.

50. Marcar paciente como de interés

    - Descripción: El usuario puede marcar pacientes como pacientes de interés.
    - Actores: facultativo.
    - Flujo de eventos:
        1) El usuario se loguea dentro del sistema.
        2) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
        3) El usuario pulsa sobre la opción para marcar paciente como de interés para la paciente que le interese.
    - Precondiciones:
        (a) El usuario debe estar logueado como facultativo.
        (b) Debe haber por lo menos una paciente registrada dentro del sistema.
    - Poscondiciones: La paciente queda marcada como paciente de interés para el usuario.

51. Ver pacientes de interés

    - Descripción: El usuario puede ver las pacientes de interés para el facultativo.
    - Actores: facultativo.
    - Flujo de eventos:
        1) El usuario se loguea dentro del sistema.
        2) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
        3) El usuario pulsa sobre la pestaña para ver pacientes de interés.
    - Precondiciones:
        (a) El usuario debe estar logueado como facultativo.
        (b) Debe haber por lo menos una paciente registrada dentro del sistema.
    - Poscondiciones: se muestran todas las pacientes de interés del facultativo.

52. Desmarcar paciente como de interés

    - Descripción: El usuario puede desmarcar pacientes como pacientes de interés.
    - Actores: facultativo.
    - Flujo de eventos:
        1) El usuario se loguea dentro del sistema.
        2) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.

3) El usuario pulsa sobre la pestaña para ver pacientes de interés.
4) El usuario pulsa sobre el botón para desmarcar paciente de interés sobre la paciente que ya no le interese tener marcada.

- Precondiciones:
  (a) El usuario debe estar logueado como facultativo.
  (b) Debe haber por lo menos una paciente registrada dentro del sistema.
  (c) Debe haber por lo menos una paciente de interés por parte del usuario.
- Poscondiciones: La paciente queda desmarcada como paciente de interés.

53. Ver últimos pacientes vistos

- Descripción: El usuario puede ver las últimas pacientes vistas por el facultativo.
- Actores: facultativo.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  3) El usuario pulsa sobre la pestaña para ver últimos pacientes vistos.
- Precondiciones:
  (a) El usuario debe estar logueado como facultativo.
  (b) Debe haber por lo menos una paciente registrada dentro del sistema.
- Poscondiciones: Se muestran las últimas pacientes atendidas por el facultativo mediante cita.

54. Consultar agenda

- Descripción: El usuario puede consultar las citas que tiene el usuario.
- Actores: facultativo.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la tarjeta para consultar la agenda.
- Precondiciones:
  (a) El usuario debe estar logueado como facultativo.
- Poscondiciones: el usuario ve todas las citas que tiene en la agenda desde el día actual en adelante.

55. Ver citas

- Descripción: El usuario puede ver las citas registradas hasta el momento.
- Actores: administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción para ver el calendario en el menú desplegable.
- Precondiciones:
  (a) El usuario debe estar logueado como administrador o tener el rol de citaciones.
- Poscondiciones: se muestran las citas de los facultativos registradas hasta el momento.

56. Añadir cita

- Descripción: El usuario puede añadir una cita a la agenda.
- Actores: administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción para ver el calendario en el menú desplegable.
  3) El usuario pulsa sobre el botón "Añadir cita".
  4) El usuario cubre los datos necesarios para la cita.
- Precondiciones:
  (a) El usuario debe estar logueado como administrador o tener el rol de citaciones.
- Poscondiciones: se ha registrado la cita correctamente dentro del sistema.

57. Modificar cita

- Descripción: El usuario puede modificar una cita registrada en la agenda.
- Actores: administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción para ver el calendario en el menú desplegable.
  3) El usuario pulsa sobre el botón para modificar cita de la cita que se quiera modificar.
  4) El usuario modifica los datos de la cita que se consideren necesarios.

5) El usuario pulsa sobre el botón para aceptar los cambios.

- Precondiciones:
  (a) El usuario debe estar logueado como administrador o tener el rol de citaciones.
  (b) Debe haber por lo menos una cita registrada en el sistema.
- Poscondiciones: los datos de la cita deben estar modificados correctamente.

58. Cancelar cita

- Descripción: El usuario puede cancelar una cita registrada en la agenda.
- Actores: administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción para ver el calendario en el menú desplegable.
  3) El usuario pulsa sobre el botón para cancelar cita de la cita que se quiera modificar.
- Precondiciones:
  (a) El usuario debe estar logueado como administrador o tener el rol de citaciones.
  (b) Debe haber por lo menos una cita registrada en el sistema.
- Poscondiciones: la cita ya no aparece registrada en el sistema.

59. Atender cita

- Descripción: El usuario puede atender una cita que esté registrada en la agenda.
- Actores: facultativo.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la tarjeta para consultar la agenda.
  3) El usuario pulsa sobre la cita a atender.
  4) El usuario pulsa sobre el botón para marcar la cita como atendida.
- Precondiciones:
  (a) El usuario debe estar logueado como facultativo.
  (b) Debe haber por lo menos una cita registrada en el sistema.

(c) La cita debe estar relacionada con el facultativo.

(d) La cita debe ser del día actual.

- Poscondiciones: se marca la cita como atendida.

60. Generar informe de seguimiento

- Descripción: El usuario puede generar un informe de seguimiento a partir de los datos de una paciente.
- Actores: facultativo.
- Flujo de eventos:
  1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
  2) El usuario pulsa sobre el número de historia de la paciente, el cual es un enlace a los detalles de la paciente.
  3) El usuario pulsa sobre el botón para generar informe.
- Precondiciones:
  (a) El usuario debe estar logueado como facultativo.
  (b) Debe haber por lo menos una paciente registrada en el sistema.
- Poscondiciones: se genera un informe con todos los datos de la paciente que se consideren necesarios, indicando filiación, antecedentes y curso clínico.

61. Ver mensajes privados

- Descripción: El usuario puede ver los mensajes privados recibidos por otros usuarios.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la tarjeta para ver los mensajes privados.
- Precondiciones:
  (a) El usuario debe estar logueado en el sistema.
- Poscondiciones: el usuario ve todos los mensajes privados que ha recibido hasta el momento, en formato de previsualización indicando el asunto del mensaje, fecha y hora del envío y quién lo ha enviado.

62. Leer mensaje privado

- Descripción: El usuario puede ver los mensajes privados recibidos por otros usuarios.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la tarjeta para ver los mensajes privados.
  3) El usuario pulsa sobre el mensaje que se quiere leer.
- Precondiciones:
  (a) El usuario debe estar logueado en el sistema.
  (b) Debe haber recibido por lo menos un mensaje.
- Poscondiciones: el usuario ve en detalle el mensaje que ha recibido.

63. Enviar mensaje privado

- Descripción: El usuario puede enviar un mensaje privado a otro usuario del sistema.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la tarjeta para ver los mensajes privados.
  3) El usuario pulsa sobre el botón para enviar un nuevo mensaje.
  4) El usuario indica un asunto, el cuerpo del mensaje y el destinatario de este.
  5) El usuario pulsa sobre el botón de enviar.
- Precondiciones:
  (a) El usuario debe estar logueado en el sistema.
  (b) Debe haber más de un usuario registrado dentro del sistema.
- Poscondiciones: el mensaje se ha enviado con éxito.

64. Solicitar interconsulta

- Descripción: El usuario puede solicitar una interconsulta a un facultativo a partir de una entrada del curso clínico para pedir opinión.
- Actores: facultativo.
- Flujo de eventos:

1) El usuario pulsa sobre la opción de ver pacientes en el menú desplegable.
2) El usuario pulsa sobre el número de historia de la paciente, el cual es un enlace a los detalles de la paciente.
3) El usuario pulsa sobre el desplegable para ver el curso clínico.
4) El usuario pulsa sobre una de las entradas del diario clínico.
5) El usuario pulsa sobre la opción "Solicitar interconsulta".
6) El usuario busca y selecciona con el facultativo con el cual se quiere comunicar, escribiendo el motivo por el cual se realiza la interconsulta.
7) El usuario pulsa sobre el botón para enviar interconsulta.

- Precondiciones:
  (a) El usuario debe estar logueado como facultativo.
  (b) Debe haber por lo menos una paciente registrada en el sistema.
  (c) Debe haber por lo menos una entrada en el curso clínico.
  (d) Debe haber por lo menos más de un usuario con el rol de facultativo.
- Poscondiciones: la interconsulta se realiza con éxito.

65. Ver tareas comunes

- Descripción: El usuario puede ver las tareas comunes asociadas al usuario.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la tarjeta para ver las tareas comunes.
- Precondiciones:
  (a) El usuario debe estar logueado en el sistema.
- Poscondiciones: el usuario ve todas las tareas comunes que tiene asignado indicando el título y quién más está en la tarea común.

66. Ver mensajes de tareas comunes

- Descripción: El usuario puede leer los mensajes asociados a una tarea común seleccionada.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.

2) El usuario pulsa sobre la tarjeta para ver las tareas comunes.
3) El usuario pulsa sobre una de las tareas comunes de la lista.

- Precondiciones:
  (a) El usuario debe estar logueado en el sistema.
  (b) Debe haber por lo menos una tarea común registrada dentro del sistema.
- Poscondiciones: se ven todos los mensajes de la tarea común ordenados descendentemente por recencia indicando quién lo ha mandado.

67. Abrir tarea común

- Descripción: El usuario puede abrir una tarea común entre usuarios.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la tarjeta para ver las tareas comunes.
  3) El usuario pulsa sobre el botón para abrir una nueva tarea común.
  4) El usuario indica los usuarios implicados, el título de la tarea común y la descripción de esta.
  5) El usuario pulsa sobre el botón para crear la tarea común.
- Precondiciones:
  (a) El usuario debe estar logueado en el sistema.
  (b) Debe haber por lo menos más de un usuario registrado dentro del sistema.
- Poscondiciones: la tarea común debe estar registrada en el sistema.

68. Añadir mensaje para tarea común

- Descripción: El usuario puede añadir un mensaje para la tarea común que se haya creado.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la tarjeta para ver las tareas comunes.
  3) El usuario pulsa sobre una de las tareas comunes de la lista.
  4) El usuario escribe un mensaje a enviar para la tarea común.
  5) El usuario pulsa sobre el botón de enviar.

- Precondiciones:
  (a) El usuario debe estar logueado en el sistema.
  (b) Debe haber por lo menos una tarea común registrada dnetro del sistema.
- Poscondiciones: el mensaje queda registrado para la tarea común.

69. Ver avisos

- Descripción: El usuario puede ver los avisos recibidos.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la tarjeta para ver los avisos.
- Precondiciones:
  (a) El usuario debe estar logueado en el sistema.
- Poscondiciones: se muestran todos los avisos recibidos, indicando fecha y hora, causa del aviso, descripción y remitente.

70. Añadir aviso

- Descripción: El usuario puede añadir un aviso a cualquier usuario registrado en el sistema.
- Actores: facultativo, administrador, citaciones.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la tarjeta para ver los avisos.
  3) El usuario pulsa sobre el botón para añadir aviso.
  4) El usuario indica el destinatario del aviso, la causa del aviso y la descripción.
- Precondiciones:
  (a) El usuario debe estar logueado en el sistema.
  (b) Debe haber más de un usuario registrado dentro del sistema.
- Poscondiciones: el aviso se añade con éxito al sistema.

71. Comprobar log de acciones

- Descripción: El usuario puede comprobar el historial de acciones realizadas dentro del sistema.

- Actores: administrador.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción para ver el log de acciones del menú desplegable.
- Precondiciones:
  (a) El usuario debe estar logueado en el sistema con el rol de administrador.
- Poscondiciones: se muestra el log con todas las acciones realizadas dentro del sistema, indicando el nivel de gravedad del mensaje, fecha y hora en la que se ha realizado la acción, quién la ha realizado, y una descripción breve de la acción.

72. Generar informe de acceso

- Descripción: El usuario puede generar un informe de acceso en base al estado actual del log de acciones.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción para ver el log de acciones del menú desplegable.
  3) El usuario pulsa sobre el botón para generar el informe de acciones.
- Precondiciones:
  (a) El usuario debe estar logueado en el sistema con el rol de administrador.
- Poscondiciones: se genera un informe de acciones donde se muestre la misma información que en el caso de uso anterior.

73. Filtrar log de acciones

- Descripción: El usuario puede filtrar las acciones registradas en el log de acciones.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción para ver el log de acciones del menú desplegable.
  3) El usuario pulsa sobre el desplegable para filtrar acciones.

4) El usuario filtra el log de acciones en base al nivel de alerta que se busca y el rango de marcas de tiempo.
5) El usuario pulsa sobre el botón de buscar.

- Precondiciones:
  (a) El usuario debe estar logueado en el sistema con el rol de administrador.
- Poscondiciones: se muestra en la tabla todas las acciones del log que cumplen con los criterios de búsqueda del filtrado.

74. Generar copia de seguridad

- Descripción: El usuario puede generar una copia de seguridad del estado actual dela BBDD.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción para gestionar las copias de seguridad.
  3) El usuario pulsa sobre el botón para generar una copia de seguridad.
- Precondiciones:
  (a) El usuario debe estar logueado en el sistema con el rol de administrador.
- Poscondiciones: se genera un archivo en formato .zip, dentro del cual se almacena un archivo en formato .sql el cual es un script de restauración de BBDD para recuperar el estado de ésta en ese momento, indicando en el nombre la fecha y la hora en la cual se ha realizado.

75. Restaurar BBDD desde copia de seguridad

- Descripción: El usuario puede restaurar la BBDD a partir de un archivo en formato .sql el cual permite restaurar el estado de la BBDD en ese momento.
- Actores: administrador.
- Flujo de eventos:
  1) El usuario se loguea dentro del sistema.
  2) El usuario pulsa sobre la opción para gestionar las copias de seguridad.
  3) El usuario indica el archivo en formato .sql generado anteriormente para restaurar la BBDD.
  4) El usuario pulsa sobre el botón para restaurar la BBDD.
- Precondiciones:

(a) El usuario debe estar logueado en el sistema con el rol de administrador.

- Poscondiciones: se restaura la BBDD de forma que se recupera en el estado en el cual estaba.

Apéndice B

Pasos seguidos para el despliegue de la aplicación

NOTA: Se da por hecho que se tiene lo necesario para poder desplegar la aplicación.

A continuación, se indican los pasos que se han seguido para poder realizar el despliegue de la aplicación web:

1. Se accede a la carpeta del proyecto.
    - cd /ubicación/final/sistema-gestion-pacientes-ginecologia-<tag del proyecto>/
2. Se generan las imágenes de Docker necesarias para arrancar el proyecto.
    - docker build -t ginecology-service:<tag del proyecto> -f dockerfiles/ginecology-service/Dockerfile .
    - docker build -t ginecology-database:<tag del proyecto> -f dockerfiles/ginecology-database/Dockerfile .
3. Se realiza el despliegue de la aplicación en producción mediante el uso de Docker Swarm.
    - docker stack deploy -c docker-compose.yml ginecologia

- Enlace a la página: https://app.rnasa-imedir.udc.es/
- Enlace al proyecto (última versión hasta el momento): https://github.com/SergioCortizo/sistema-gestion-pacientes-ginecologia/releases/tag/1.0.7

- Usuarios disponibles en el entorno de producción (todos tienen como contraseña el nombre de usuario):
  - admin -> Permisos completos (administrador, facultativo y citaciones).
  - luz.cuesta -> Permisos de administrador.
  - jfeo -> Permisos de facultativo.
  - drodriguez -> Permisos de citaciones.

Apéndice C

Capturas de pantalla de la aplicación

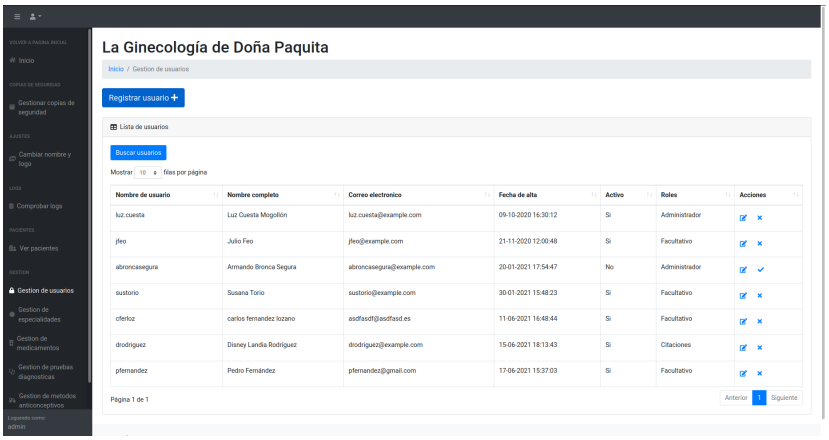


Figura C.1: Pantalla de listado de usuarios

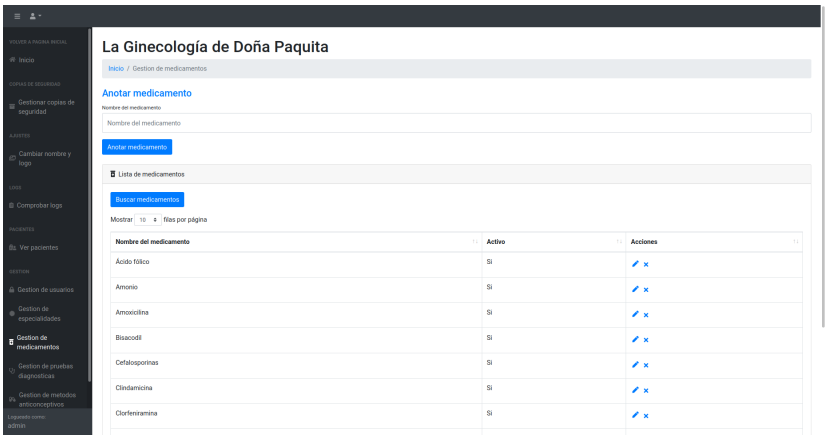


Figura C.2: Pantalla de gestión de medicamentos

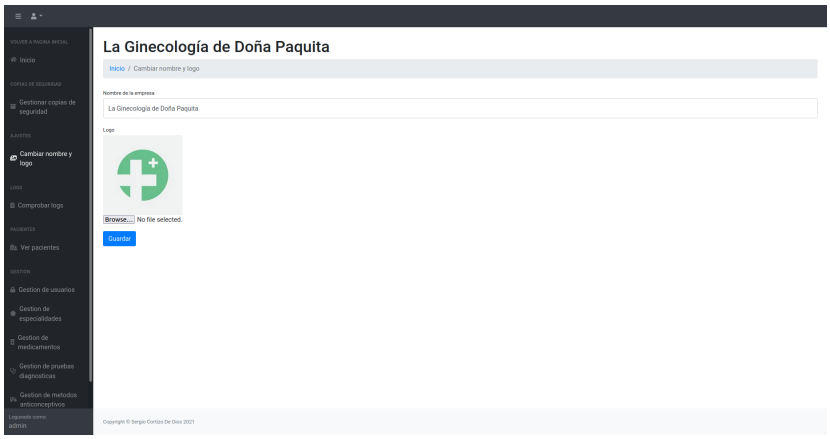


Figura C.3: Pantalla de cambio de nombre y logo

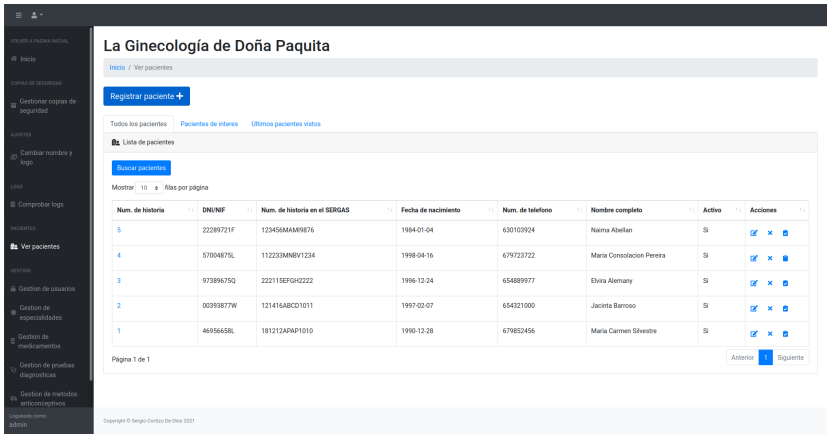


Figura C.4: Captura de pantalla de listado de pacientes

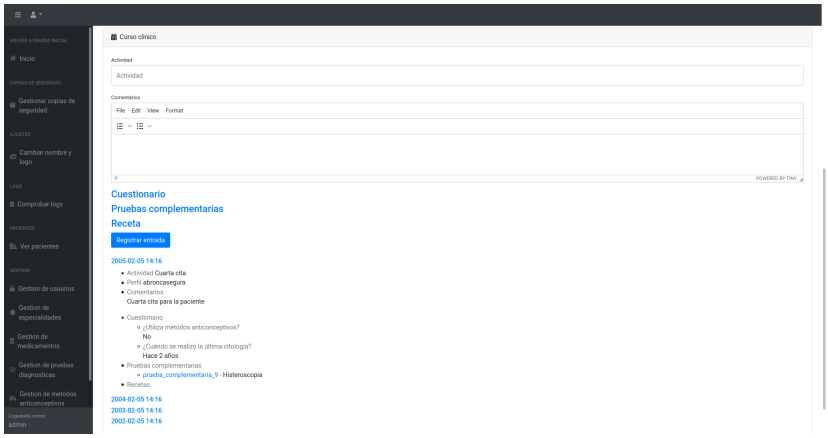


Figura C.5: Captura de pantalla de curso clínico

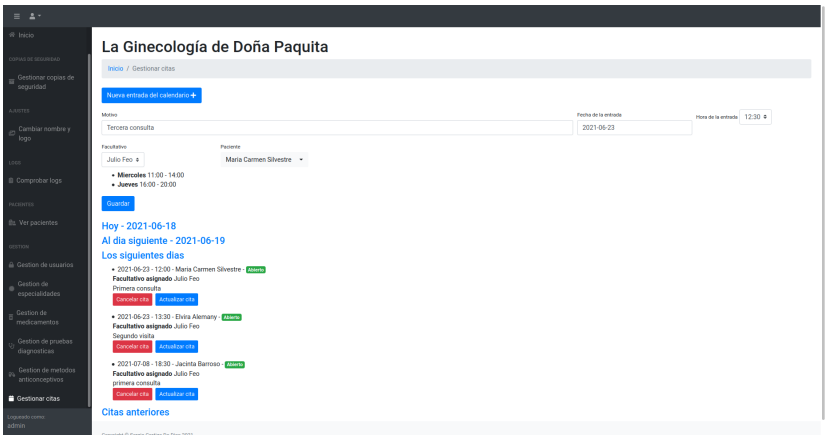


Figura C.6: Captura de pantalla de gestión de agendas

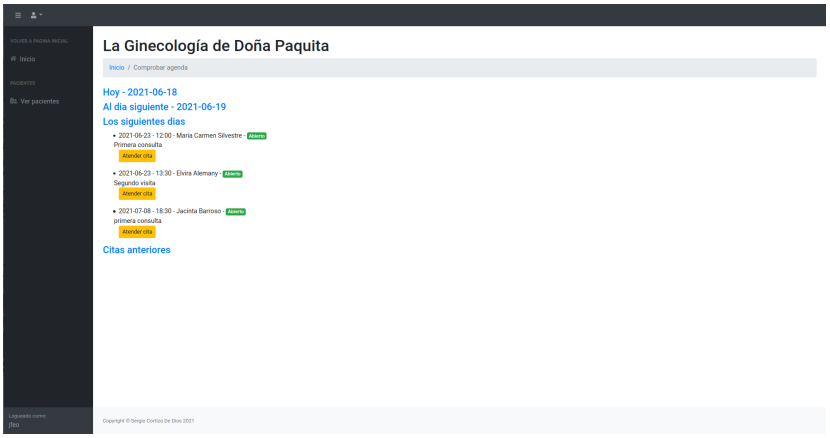


Figura C.7: Captura de pantalla de consulta de agenda

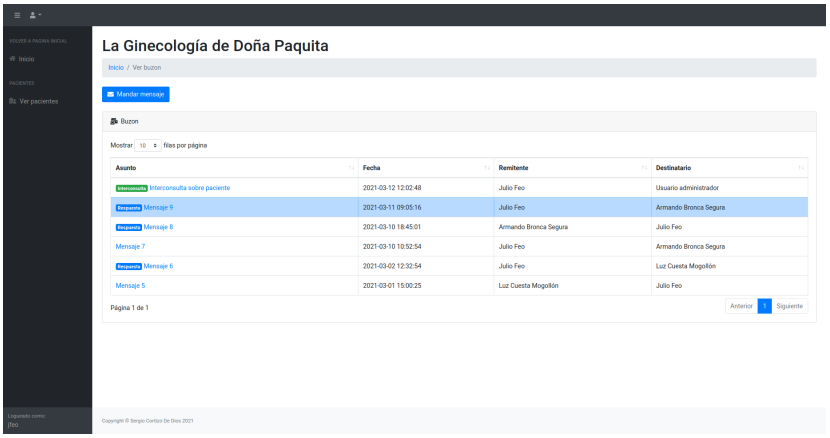


Figura C.8: Captura de pantalla de mensajería privada

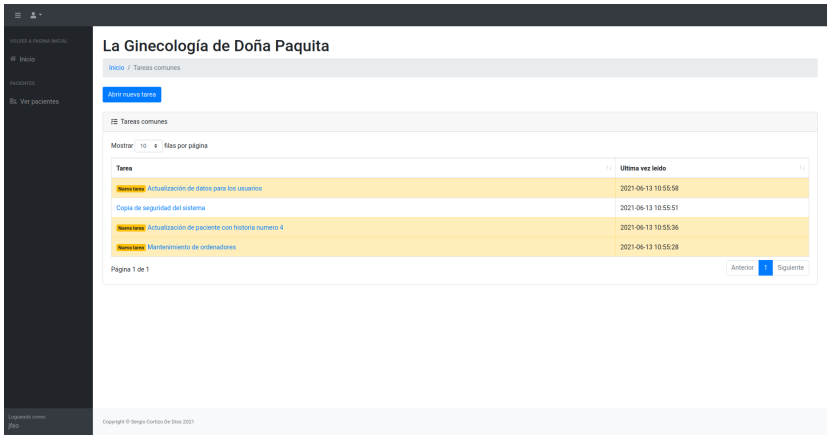


Figura C.9: Captura de pantalla de tareas comunes

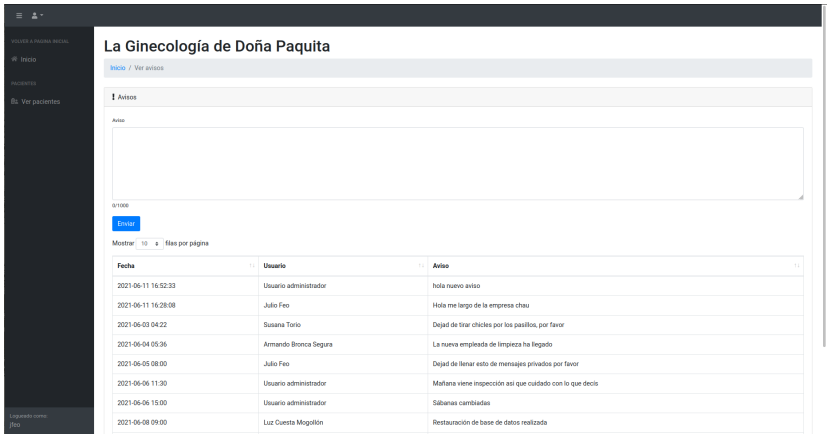


Figura C.10: Captura de pantalla de avisos

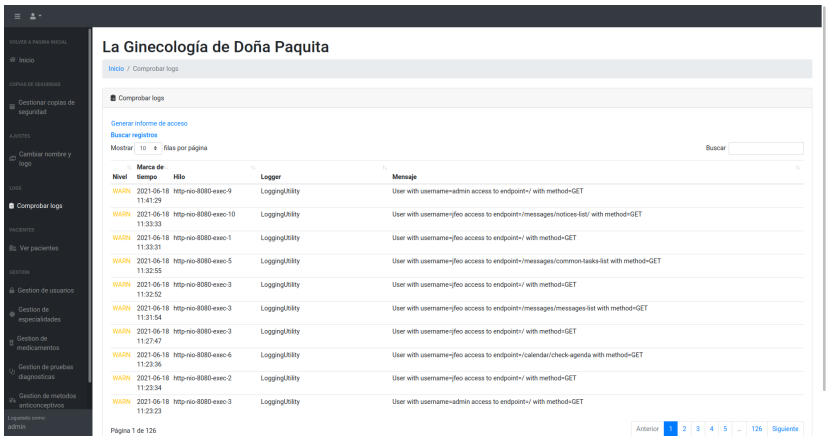


Figura C.11: Captura de pantalla de logs

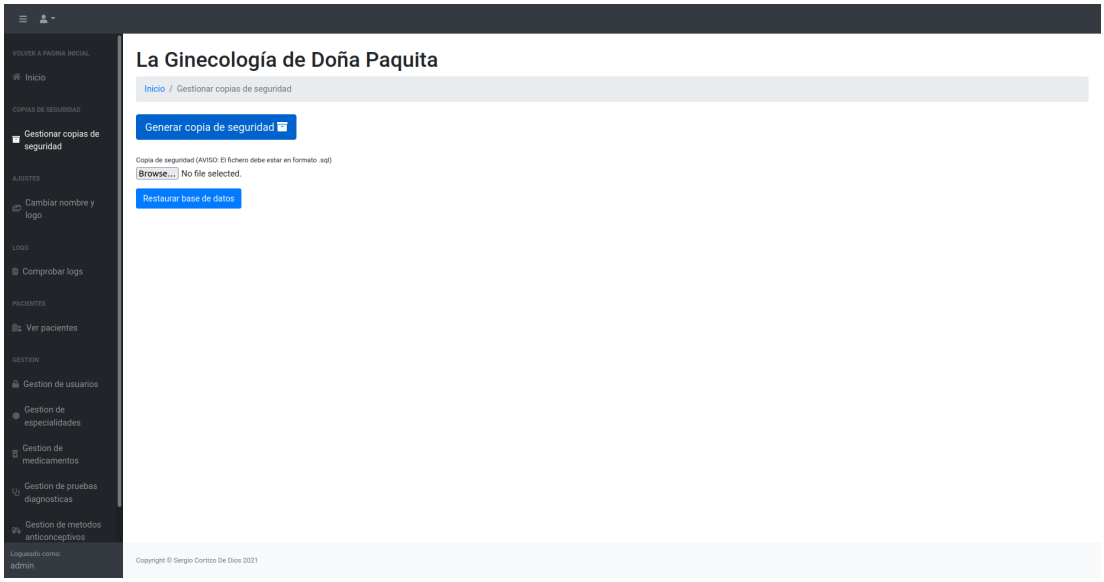


Figura C.12: Captura de pantalla de gestión de copias de seguridad

Bibliografía

[1] SIVSA, "Fenix, software de gestión clínica en la nube," consultado el 9 de Febrero de 2021. [En línea]. Disponible en: https://www.sivsa.com/site/fenix/

[2] AEPD, "Agencia española de protección de datos (aepd)," 2019, consultado el 11 de Febrero de 2021. [En línea]. Disponible en: https://www.aepd.es/sites/default/files/2019-11/guia-privacidad-desde-diseno.pdf

[3] B. O. del Estado, "Ley orgánica 3/2018, de 5 de diciembre, de protección de datos personales y garantía de los derechos digitales." 2018, consultado el 12 de Febrero de 2021. [En línea]. Disponible en: https://www.boe.es/boe/dias/2018/12/06/pdfs/BOE-A-2018-16673.pdf

[4] BOE, "Ley 34/2002, de 11 de julio, de servicios de la sociedad de la información y de comercio electrónico," 2002, consultado el 13 de Febrero de 2021. [En línea]. Disponible en: https://www.boe.es/buscar/pdf/2002/BOE-A-2002-13758-consolidado.pdf

[5] P. Europeo, "Reglamento general de protección de datos ue 2016/679," 2016, consultado el 13 de Febrero de 2021. [En línea]. Disponible en: https://www.boe.es/doue/2016/119/L00001-00088.pdf

[6] QSOFT, "Ginesalus, software para ginecología, obstetricia y fertilidad," consultado el 9 de Febrero de 2021. [En línea]. Disponible en: http://www.ginesalus.com/

[7] "Ubuntu 18.04.5 lts," consultado el 9 de Febrero de 2021. [En línea]. Disponible en: https://releases.ubuntu.com/18.04/

[8] "Eclipse," consultado el 14 de Febrero de 2021. [En línea]. Disponible en: https://www.eclipse.org/

[9] "Git," consultado el 16 de Febrero de 2021. [En línea]. Disponible en: https://git-scm.com/about/branching-and-merging

[10] "Java," consultado el 17 de Febrero de 2021. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)

[11] Wikipedia, "Hibernate," consultado el 18 de Febrero de 2021. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Hibernate

[12] Everis, "¿qué es java hibernate? ¿por qué usarlo?" consultado el 18 de Febrero de 2021. [En línea]. Disponible en: https://ifgeekthen.everis.com/es/que-es-java-hibernate-por-que-usarlo

[13] Wikipedia, "Spring framework," consultado el 18 de Febrero de 2021. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Spring_Framework

[14] ——, "Maven," consultado el 19 de Febrero de 2021. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Maven

[15] Maven, "Introduction to the pom," consultado el 19 de Febrero de 2021. [En línea]. Disponible en: http://maven.apache.org/guides/introduction/introduction-to-the-pom.html

[16] S. Boot, "Spring boot starter test," consultado el 20 de Febrero de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-test

[17] ——, "Spring boot starter data jpa," consultado el 20 de Febrero de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-data-jpa

[18] ——, "Spring boot starter web," consultado el 20 de Febrero de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-web

[19] ——, "Spring boot starter thymeleaf," consultado el 20 de Febrero de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-thymeleaf

[20] ——, "Spring boot devtools," consultado el 20 de Febrero de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-devtools

[21] Thymeleaf, "Thymeleaf extras spring security," consultado el 20 de Febrero de 2021. [En línea]. Disponible en: https://github.com/thymeleaf/thymeleaf-extras-springsecurity

[22] MySQL, "Mysql jdbc connector," consultado el 21 de Febrero de 2021. [En línea]. Disponible en: https://dev.mysql.com/downloads/connector/j/

[23] S. Framework, "Spring security crypto," consultado el 20 de Febrero de 2021. [En línea]. Disponible en: https://docs.spring.io/spring-security/site/docs/5.0.x/reference/html/crypto.html

[24] ——, "Spring security config," consultado el 20 de Febrero de 2021. [En línea]. Disponible en: https://docs.spring.io/spring-security/site/docs/5.0.x/reference/html/crypto.html

[25] ——, "Spring security web," consultado el 21 de Febrero de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/org.springframework.security/spring-security-web

[26] ——, "Spring boot configuration processor," consultado el 21 de Febrero de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-configuration-processor

[27] Hibernate, "Validation api," consultado el 21 de Febrero de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/javax.validation/validation-api

[28] Apache, "Apache tika core," consultado el 21 de Marzo de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/org.apache.tika/tika-core

[29] ——, "Apache commons codec," consultado el 21 de Marzo de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/commons-codec/commons-codec

[30] F. S. Project, "Flying saucer pdf," consultado el 14 de Abril de 2021. [En línea]. Disponible en: https://github.com/flyingsaucerproject/flyingsaucer

[31] P. Lombok, "Lombok," consultado el 21 de Abril de 2021. [En línea]. Disponible en: https://projectlombok.org/

[32] S. Framework, "Spring boot starter log4j2," consultado el 22 de Abril de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-log4j2

[33] ——, "Spring boot starter aop," consultado el 22 de Abril de 2021. [En línea]. Disponible en: https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-aop

[34] SeunMatt, "Mysql backup4j," consultado el 12 de Mayo de 2021. [En línea]. Disponible en: https://github.com/SeunMatt/mysql-backup4j

[35] S. Framework, "Spring boot maven plugin," consultado el 20 de Febrero de 2021. [En línea]. Disponible en: https://docs.spring.io/spring-boot/docs/current/maven-plugin/reference/htmlsingle/

[36] MojoHaus, "Sql maven plugin," consultado el 21 de Febrero de 2021. [En línea]. Disponible en: https://www.mojohaus.org/sql-maven-plugin/

[37] MySQL, "Mysql," consultado el 21 de Febrero de 2021. [En línea]. Disponible en: https://dev.mysql.com/doc/refman/8.0/en/

[38] Bootstrap, "Bootstrap," consultado el 23 de Febrero de 2021. [En línea]. Disponible en: https://getbootstrap.com/docs/5.0/getting-started/introduction/

[39] JQuery, "Jquery," consultado el 24 de Febrero de 2021. [En línea]. Disponible en: https://desarrolloweb.com/manuales/manual-jquery.html

[40] D. Stutz, "Bootstrap multiselect," consultado el 20 de Marzo de 2021. [En línea]. Disponible en: http://davidstutz.github.io/bootstrap-multiselect/

[41] V. Silva, "jquery ensure max length," consultado el 21 de Marzo de 2021. [En línea]. Disponible en: https://github.com/vsilva472/jquery-ensure-max-length

[42] J. V. Team, "jquery validation plugin," consultado el 21 de Marzo de 2021. [En línea]. Disponible en: https://jqueryvalidation.org/

[43] S. Ltd, "Datatables," consultado el 9 de Febrero de 2021. [En línea]. Disponible en: https://datatables.net/

[44] U. Solutions, "Bootstrap datepicker," consultado el 10 de Marzo de 2021. [En línea]. Disponible en: https://github.com/uxsolutions/bootstrap-datepicker

[45] lkfnn, "jquery-huntertimepicker," consultado el 15 de Junio de 2021. [En línea]. Disponible en: https://github.com/lkfnn/jquery-hunterTimePicker

[46] T. T. Inc., "Tinymce," consultado el 3 de Abril de 2021. [En línea]. Disponible en: https://www.tiny.cloud/

[47] D. Inc., "Docker," consultado el 1 de Mayo de 2021. [En línea]. Disponible en: https://www.redhat.com/es/topics/containers/what-is-docker

[48] C. Sanchez, "Docker maven image," consultado el 1 de Mayo de 2021. [En línea]. Disponible en: https://hub.docker.com/_/maven

[49] C. de Docker y el equipo de desarrollo de MySQL, "Docker mysql image," consultado el 1 de Mayo de 2021. [En línea]. Disponible en: https://hub.docker.com/_/mysql

[50] fradelg, "mysql-cron-backup," consultado el 4 de Mayo de 2021. [En línea]. Disponible en: https://hub.docker.com/r/fradelg/mysql-cron-backup

[51] D. Inc., "Docker compose," consultado el 1 de Mayo de 2021. [En línea]. Disponible en: https://docs.docker.com/compose/

[52] O. Corporation, "Oracle vm virtualbox," consultado el 3 de Mayo de 2021. [En línea]. Disponible en: https://www.virtualbox.org/manual/

[53] Scrum.org, "What is scrum?" consultado el 15 de Febrero de 2021. [En línea]. Disponible en: https://www.scrum.org/resources/what-is-scrum

[54] proyectosagiles.org, "Qué es scrum," consultado el 15 de Febrero de 2021. [En línea]. Disponible en: https://proyectosagiles.org/que-es-scrum/

[55] B. A. M. Friend, "Scrum," consultado el 15 de Febrero de 2021. [En línea]. Disponible en: https://beagilemyfriend.com/scrum/

[56] Scrum.org, "What is a scrum master?" consultado el 15 de Febrero de 2021. [En línea]. Disponible en: https://www.scrum.org/resources/what-is-a-scrum-master

[57] ——, "What is a product owner?" consultado el 15 de Febrero de 2021. [En línea]. Disponible en: https://www.scrum.org/resources/what-is-a-product-owner

[58] ——, "What is a developer in scrum?" consultado el 15 de Febrero de 2021. [En línea]. Disponible en: https://www.scrum.org/resources/what-is-a-scrum-developer

[59] C. Laborales, "Diferencias entre el salario bruto y el salario neto," consultado el 17 de Junio 2021. [En línea]. Disponible en: https://www.cuestioneslaborales.es/diferencias-entre-el-salario-bruto-y-el-salario-neto/

[60] ——, "La tributación mínima en la nómina de un trabajador," consultado el 17 de Junio 2021. [En línea]. Disponible en: https://www.cuestioneslaborales.es/la-retencion-minima-en-la-nomina-de-un-trabajador/

[61] Jobted, "Sueldo del analista de sistemas en españa," consultado el 17 de Junio 2021. [En línea]. Disponible en: https://www.jobted.es/salario/analista-sistemas

[62] ——, "Sueldo del programador en españa," consultado el 17 de Junio 2021. [En línea]. Disponible en: https://www.jobted.es/salario/programador

[63] tecnoempleo, "Informe empleo informática - junio 2021," consultado el 17 de Junio 2021. [En línea]. Disponible en: https://www.tecnoempleo.com/informe-empleo-informatica.php

[64] Wikipedia, "Dicom," consultado el 16 de Junio 2021. [En línea]. Disponible en: https://es.wikipedia.org/wiki/DICOM

[65] NEMA, "Dicom," consultado el 16 de Junio 2021. [En línea]. Disponible en: https://www.dicomstandard.org/current