

Trabajo fin de grado

Desarrollo de Aplicaciones Web

2020/2022

Instituto tecnológico

edix

Ignacio Viseras Riego - Rodrigo Sendino Sanz

Índice

1	Introducción Introducción [EN] 1-2
2	Objetivos del proyecto Situación actual Sistemas actuales 3-7
3	Módulos Formativos Aplicados en el TFG 8-9
4	Herramientas/Lenguajes Utilizados 10-15
5	Fases del Proyecto 16-36
6	Conclusiones Visuales 37-44
7	Agradecimientos 45
8	Anexos 45
9	Bibliografía 46

1 Introducción

En este Trabajo Fin de Grado para el Ciclo Formativo de Grado Superior en Desarrollo de Aplicaciones Web, exponemos una aplicación web basada en la gestión servicios sanitarios en el ámbito de uso de medico/cliente.

Se ha elegido esta idea debido a la incipiente cantidad de servicios sanitarios online a consecuencia de esta pasada pandemia, en la cual, nos hemos podido percatar de una atención médica de forma telemática es algo necesario y ha llegado para quedarse y puede abrir nuevos campos e integraciones en la rama del desarrollo web que podría mejorar sustancialmente el uso del cualquier sistema sanitario, generando una sensación mucho más positiva y cercana para el usuario.

Esta aplicación web está enfocada para uso médico, ya que cuenta con diferentes campos e información almacenada del usuario que podrá consultar y actualizar los datos siempre que quiera, de estos datos se almacenarán sólo los más vitales, por lo que hace que la aplicación sea más ligera y rápida.

Con esta página web se trata de crear un sistema de gestión médica ágil y sencillo que facilite las gestiones tanto de los pacientes como de los médicos, generando un historial clínico que se puede consultar en cualquier momento, la capacidad de pedir citas y visualizar las mismas en un calendario, ver las medicinas asignadas y los datos de diagnóstico y físicos de forma individual y con tu médico, también se ha incorporado un sistema de mensajes con el cual puedes realizar una consulta a tu médico o poder enviar mensajes a otros pacientes, en definitiva crear una red sanitaria en la que puedas acceder de una forma clara y accesible a la información que necesites en el momento que necesites.

En conclusión, esta aplicación expone la información de manera que los pacientes puedan acceder fácilmente a ella y puedan interactuar para realizar consultas con su médico y realizar las funciones de citación médica y puedan visualizar las mismas de forma clara en su navegador, y puedan acceder hasta una descarga del pdf de la misma citación, esta aplicación pone toda la información a servicio del consultante para que pueda tener toda la información necesaria sobre sus citas médicas, medicinas, diagnósticos, sin que tenga que realizar ningún procedimiento dificultoso para el usuario.

1 Introduction

In this Final Degree Project for the Higher Degree Training Cycle in Web Application Development, we expose a web application based on the management of health services in the field of doctor/client use.

This idea has been chosen due to the incipient number of online health services as a result of this past pandemic, in which, we have been able to realize that telematic medical care is something necessary and has come to stay and can open new fields and integrations in the branch of web development that could substantially improve the use of any health system, showing a much more positive and close feeling for the user.

This web application is focused on medical use, since it has different fields and saved information of the user that can consult and update the data whenever he wants, of these data only the most vital will be stored, which makes the application more light and fast.

With this web page, it is about creating an agile and simple medical management system that facilitates the management of both patients and doctors, discovering a clinical history that can be consulted at any time, the ability to request appointments and view them in a calendar, see the exposed medicines and the diagnostic and physical data individually and with your doctor, a message system has also been incorporated with which you can consult your doctor or be able to send messages to other patients, in definitively create a health network in which you can access in a clear and accessible way the information you need at the time you need.

In conclusion, this application exposes the information in a way that patients can easily access it and can interact to make consultations with their doctor and perform the functions of medical appointment and they can see the same clear forms in their browser, and they can access up to a download the pdf of the same citation, this application puts all the information at the service of the consultant so that he can have all the necessary information about his medical appointments, medications, diagnoses, without having to carry out any difficult procedure for the user.

2 Objetivos del proyecto

Situación actual

Para diversificarnos y destacarnos en el mercado nos hemos fijado 3 factores diferenciadores.

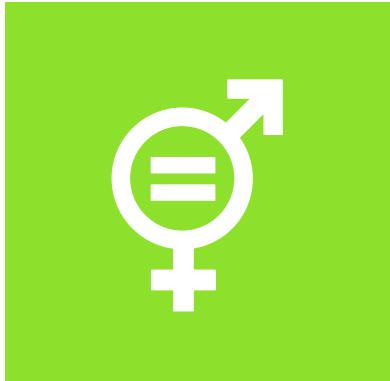
En esta sección muestro los objetivos prioritarios y como los hemos reflejado nuestra propia estrategia.



Cercanía

Conseguir un trato directo y ágil

Al tener la información accesible siempre que se quiera consultar creamos un trato sincero y directo



Accesibilidad

Criterios de diseño y opciones de usuario

Abordando temas de accesibilidad utilizamos un diseño sencillo y con opciones inclusivas para todos.



Interactividad

Chat interno

Se incorporan un chat donde los usuarios pueden interactuar entre si

Sistemas actuales

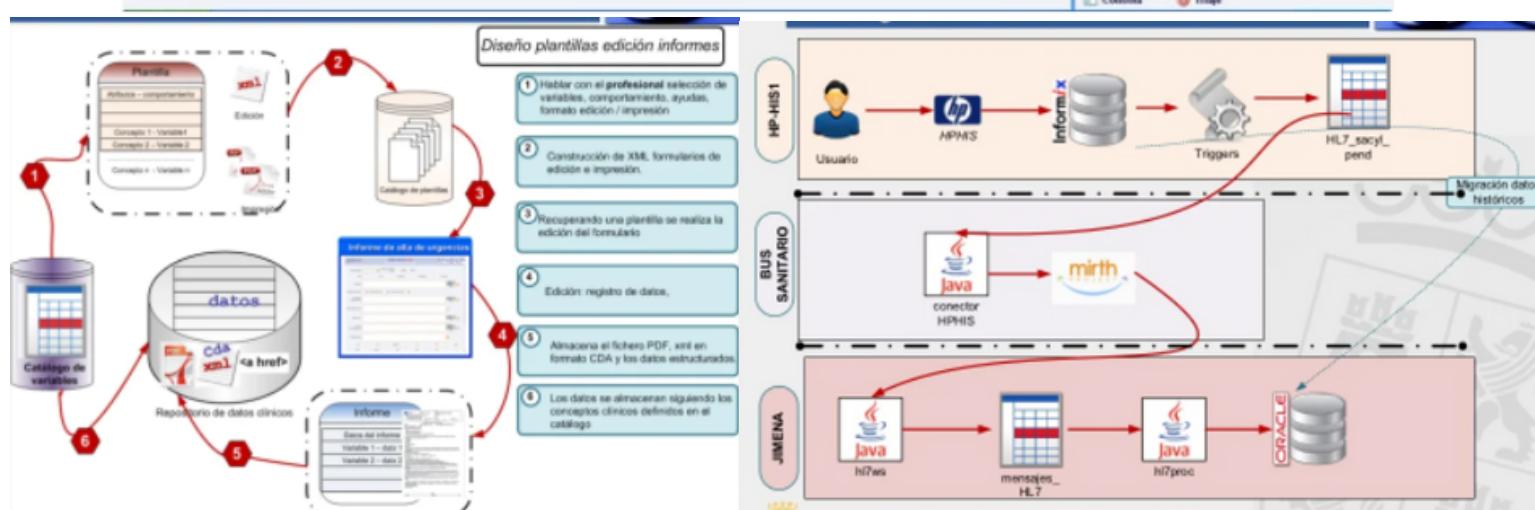
Con el fin de realizar un estudio de la competencia y comprobar los diferentes proyectos en curso de los diferentes sistemas de gestión sanitaria hemos realizado esta tabla:

Nombre	Descripción	Datos interesantes
--------	-------------	--------------------

Jimena
Servicio de Informático de Hospital en Castilla y Leon

- Buena aceptación de los médicos
- Telemedicina

The screenshot shows the Jimena software interface. At the top, there's a navigation bar with links like 'Favoritos', 'INTER (0)', 'TELE (0)', 'CEX', 'HOSP', 'URG', 'QUI', 'CRM', 'HRA', 'PLAB', 'PRUEBAS', 'CEL', 'PRX', 'Asistentes', 'Redigitalización', 'Diálogos', and 'Problemas'. Below the navigation is a search bar with fields for 'Centro: HMDC', 'Zona: Todos', 'Fecha: 22/02/2012', and a search button. The main area displays a list of patients with columns for 'Número' and 'Observaciones'. A message at the bottom says 'Mostrando 1-19 de 19'. To the right, there's a detailed view of a report history with sections for 'Historia', 'Documentos', 'Enlaces', and 'Anexos'. A legend at the bottom right defines icons for 'Detalle', 'Informes', 'Petición', 'Laboratorio', 'Inf. Entrada', 'Interconsultas', 'Primaria', 'R. clínica', 'Consola', and 'Trabajo'.



Sistemas actuales

Con el fin de realizar un estudio de la competencia y comprobar los diferentes proyectos en curso de los diferentes sistemas de gestión sanitaria hemos realizado esta tabla:

Nombre	Descripción	Datos interesantes
--------	-------------	--------------------

SACYL conecta

Aplicación de gestión de citas en Castilla y Leon

- Posibilidad de revisar y descargar documentos
- Búsqueda de centros cercanos

The screenshot shows the Sacyl Conecta app interface. At the top, a blue header bar displays the app's name. Below it, a large section is titled "Bienvenid@ a la nueva APP de Sacyl". To the right, there is an image of a hand holding a smartphone displaying the app. The main content area is divided into several sections: "Selección usuario" (with icons for "Cita previa primaria" and "Test COVID-19"), "Documento clínico" (with icons for "Buscar centros" and "Documentación Clínica"), and "Gestión usuarios" (with icons for "Pediatría CyL" and "Gestión usuarios"). The bottom of the screen features the Junta de Castilla y León logo and the Sacyl logo.

This screenshot shows the "DOCUMENTACIÓN CLÍNICA" section of the Sacyl Conecta app. It includes a sub-section for "DOCUMENTOS DISPONIBLES" which lists "Hoja de medicación", "Certificado digital COVID UE", and "Pruebas diagnósticas COVID-19". A red box highlights the "Certificado digital COVID UE" section, which shows "3 certificados disponibles" and a "Ver certificados" button. Below this, there is a note about viewing information for patients in the SINTROM document. The overall layout is clean and organized, with clear icons and text.

Sistemas actuales

Con el fin de realizar un estudio de la competencia y comprobar los diferentes proyectos en curso de los diferentes sistemas de gestión sanitaria hemos realizado esta tabla:

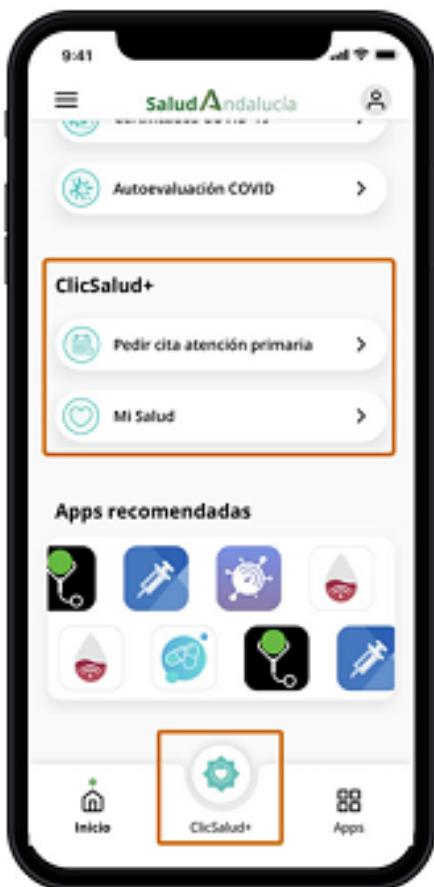
Nombre	Descripción	Datos interesantes
Mi Carpeta de Salud	Consulta su información clínica, citas sanitarias y otros servicios en Madrid	<ul style="list-style-type: none">• Acceso mediante cl@ve, dni electrónico...



Sistemas actuales

Con el fin de realizar un estudio de la competencia y comprobar los diferentes proyectos en curso de los diferentes sistemas de gestión sanitaria hemos realizado esta tabla:

Nombre	Descripción	Datos interesantes
Clic Salud+	Consulta su información clínica, citas sanitarias y otros servicios en Andalucía	<ul style="list-style-type: none">• Consulta de datos personales



Módulos Formativos Aplicados en el TFG

Durante el desarrollo del proyecto, hemos utilizado diferentes los módulos del grado los cuales son:

Programación y Desarrollo Web Entorno Servidor

En el **Back-End** de la aplicación se ha utilizado el lenguaje de programación **Java** utilizando el paradigma de programación orientada a objetos, esta parte se encarga de darle todo el dinamismo de la pagina y controla su lógica.

Lenguajes de Marcas y Desarrollo Web Entorno Cliente

Hemos utilizado esta asignatura para la parte **Front-End** aplicando el **HTML** (estructura), **CSS** (estilos) y **JavaScript** (Peticiones a apis, alertas).

Javascript también se ha utilizado en el formulario de registro para validar la información introducida.

Bases de Datos

En nuestra aplicación hemos creado un esquema de base de datos que consta de 10 tablas, con diferentes relaciones, las tablas son:

- usuario
- usuario_perfil
- perfil
- diagnostico
- información
- cita
- historial clinico
- medicina
- tipo
- comentarios

Más adelante exploraremos mas afondo estas tablas.

Entornos de Desarrollo

Con el fin de realizar un versionado de la aplicación se ha utilizado git el cual es un software de control de versiones diseñado por Linus Torvalds y también hemos realizado la plataforma GitHub creando el repositorio de este mismo proyecto.

Despliegue de Aplicaciones Web

Hemos utilizado este modulo para realizar un resumen/tutorial de como realizar el despliegue de esta aplicación.

Diseño de Interfaces Web

En este modulo se basa todo el sistema de estilo y esquema visual de la pagina, creando la guía visual y el wireframe de la aplicación junto a su modelo totalmente responsive.

También hemos trabajado las asignaturas de **Inglés** traduciendo la introducción de este trabajo e **Empresa e Iniciativa Emprendedora** en los puntos anteriores de Situación actual y Sistemas actuales

4 Herramientas Utilizadas

A continuación vamos a mostrar las herramientas y lenguajes que hemos utilizado en esta aplicación.

Eclipse Java EE



Eclipse es un entorno de desarrollo software multi-lenguaje construido alrededor de un workspace al que pueden incluirse un gran número de plug-ins que proporcionan funcionalidades concretas relacionadas con lenguajes específicos o con la interacción con otras herramientas implicadas en el desarrollo de una aplicación.

A screenshot of the Eclipse Java IDE interface. The central area shows a Java code editor with the file 'AbstractBitStreamIndexWriter.java' open. The code defines an abstract class 'AbstractBitStreamIndexWriter' that implements the 'IndexWriter' interface. It contains several protected fields and methods related to indexing documents and terms. To the left is the 'Package Explorer' view showing the project structure with various source and build files. At the bottom is the 'Problems' view, which displays a list of errors and warnings from the build process. One error is visible: 'NLS unused message: osgi.nls.warnings in: org.eclipse.wst.jsdt.debug.internal.ui.message'. The 'Plug-in' column lists 'org.eclipse.osgi' and the 'Date' column shows '2/22/12 1:16 PM'.

Fuente:

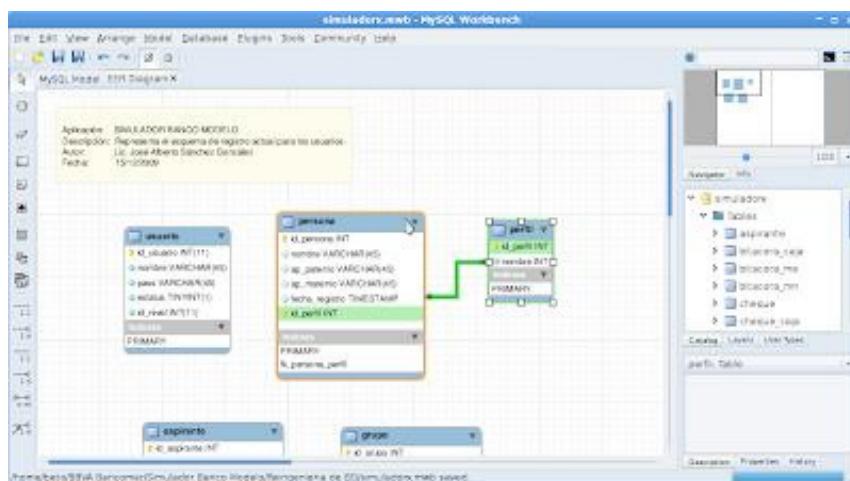
<https://es.wikipedia.org/wiki/Archivo:Eclipse-Luna-Logo.svg>

<https://stackoverflow.com/questions/5053834/eclipse-ide-for-java-full-dark-theme>

MySQL WorkBench



MySQL Workbench es una herramienta visual de diseño de bases de datos que integra desarrollo de software, administración de bases de datos, diseño de bases de datos, gestión y mantenimiento para el sistema de base de datos MySQL.



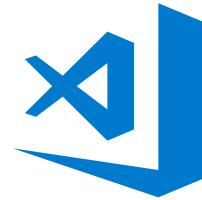
Fuente:

<https://icon-icons.com/es/icono/mysql-oficial-logo/169938>

<https://jaehoo.wordpress.com/2009/12/15/mysql-workbench-en-ubuntu-9-10/>

<https://www.arsys.es/blog/interfaces-graficas-mysql>

11



Visual Studio Code

Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y Web. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.

También es personalizable, por lo que los usuarios pueden cambiar el tema del editor, los atajos de teclado y las preferencias. Es gratuito y de código abierto,¹² aunque la descarga oficial está bajo software privativo e incluye características personalizadas por Microsoft.³

The screenshot shows the Visual Studio Code interface with three tabs open: 'main.js', 'style.css', and 'home.html'. The 'EXPLORER' sidebar on the left shows the project structure, including files like '.github', '.vscode', 'assets', 'static', 'images', 'javascript', 'style.css', 'favicon.ico', 'templates', 'home.html', 'app.py', 'README.md', 'requirements.txt', and 'vscode.github-issues'. The 'main.js' tab contains JavaScript code for a paw toggle feature. The 'style.css' tab contains CSS for a jumbotron and example rows. The 'home.html' tab contains the HTML template for the application.

```
main.js
theCatSaidNo > static > javascript > main.js > ...
1 pawToggled = false;
2 var myTimeout;
3
4 function callbackToggle() {
5     return function () {
6         if (pawToggled) {
7             document.getElementById('paw').style.backgroundImage = 'url("https://raw.githubusercontent.com/...
8         }
9     }
10 }
11
12 function togglePaw() {
13     if (!pawToggled) {
14         // Runs when we toggle the button
15         document.getElementsByClassName('jumbotron')[0].style.backgroundColor = '#6b7381';
16         myTimeout = setTimeout(callbackToggle, 250);
17     } else {
18         document.getElementsByClassName('jumbotron')[0].style.backgroundColor = '#fff';
19         clearTimeout(myTimeout);
20     }
21     pawToggled = !pawToggled;
22 }
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37

style.css
theCatSaidNo > static > stylesheets > style.css > ...
1 body {
2     font-family: 'Montserrat', 'Lato', 'Open Sans', sans-serif;
3     color: #0b7381;
4     background: #fff;
5     -webkit-touch-callout: none;
6     -webkit-user-select: none;
7     -khtml-user-select: none;
8     -moz-user-select: none;
9     -ms-user-select: none;
10    user-select: none;
11    transition: background-color 0.25s;
12 }
13
14 .jumbotron {
15     background: #6b7381;
16     color: #bdc1c8;
17 }
18 .jumbotron h1 {
19     color: #fff;
20 }
21 .example {
22     margin: 4rem auto;
23 }
24 .example > .row {
25     margin-top: 2rem;
26     height: 5rem;
27     vertical-align: middle;
28     text-align: center;
29     border: 1px solid #rgba(189, 193, 204);
30 }
31 .example > .row:first-of-type {
32     border: none;
33     height: auto;
34     text-align: left;
35 }
36 .example h3 {
37     font-weight: 400;
}

home.html
theCatSaidNo > templates > home.html > ...
1 <!DOCTYPE html>
2 <html>
3
4     <head>
5         <title>The Cat said No!</title>
6         <link href="https://fonts.googleapis.com/css?family=Montserrat|Lato|Open+Sans" rel="stylesheet">
7         <link rel="stylesheet" href="https://raw.githubusercontent.com/theCatSaidNo/...
8             integrity="sha384-BVY1lS1FeK1v...
9             <link rel="stylesheet" href="https://raw.githubusercontent.com/theCatSaidNo/...
10            <link rel="stylesheet" href="..</style>
11     </head>
12
13     <body class="preload">
14         <div class="centered">
15             <button type="button" class="handle" onclick="togglePaw()" id="paw">
16                 <div class="handle"></div>
17             </button>
18             <div class="catpaw-container">
19                 The Cat said No!</h1>
23                 <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
24                     $(window).load(function () {
25                         $("body").removeClass("preload");
26                     });
27                 </script>
28                 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js" integrity="sha384-Bvk...crossorigin="anonymous"></script>
29             </div>
30         </div>
31     </body>
32
33
34
35
36
37
```

Fuente:

<https://www.whatstnew.com/2021/10/23/visual-studio-code-el-editor-de-codigo-de-microsoft-ya-tiene-una-version-web/>

Git / GitHub



GitHub es una forja (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de ordenador.

El software que opera GitHub fue escrito en Ruby on Rails. Desde enero de 2010, GitHub opera bajo el nombre de GitHub, Inc. Anteriormente era conocida como Logical Awesome LLC. El código de los proyectos alojados en GitHub se almacena generalmente de forma pública.

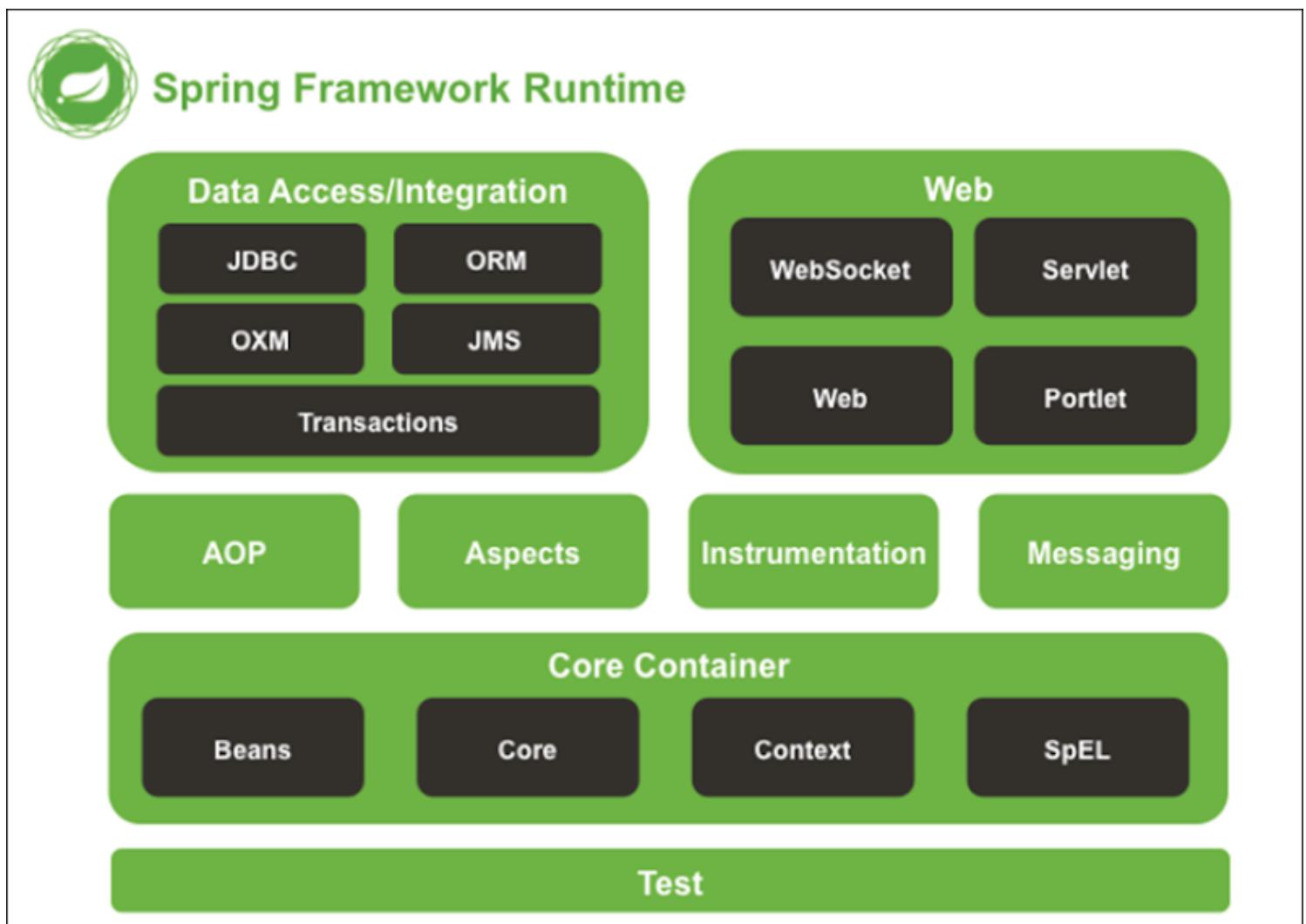
The screenshot shows the GitHub desktop application interface. At the top, there are three tabs: 'Current Repository' (set to 'desktop'), 'Current Branch' ('esc-pr'), and 'Fetch origin' (last fetched 3 minutes ago). Below these are two tabs: 'Changes' and 'History', with 'History' being the active tab. The main area displays a list of commits from a pull request. The first commit is titled 'Add event handler to dropdown component'. It shows a commit message from 'iAmWillShepherd and Markus Olsson committed a day ago' with a link to '#3972'. The commit details show it was co-authored by 'Markus Olsson <niik@users.noreply.github.com>'. The commit message is 'Add event handler to dropdown component'. The code diff shows changes made to 'app/src/ui/t.../dropdown.tsx'. The diff highlights several additions (green) and deletions (red). The code snippet includes imports like 'React.Component<', 'OcticonSymbol', and 'DropdownState'. The commit also includes a note about remaking a triangle octicon in a 12px version. Other commits listed include 'Merge branch 'master' into esc-pr', 'Merge pull request #4044 from des...', 'Merge pull request #4070 from desk...', 'bump to beta3', 'Merge pull request #4057 from desk...', 'Merge pull request #4067 from desk...', and 'Release to 1.1.0-beta2'.

Fuente:

<https://www.aluracursos.com/blog/git-y-github-que-son-y-primeros-pasos>

Spring es un framework del lenguaje de programación java, y un framework en programación es el resultado de la evolución de la ingeniería del software, estos son creados por programadores para programadores, con la finalidad de estandarizar el trabajo, resolver, agilizar y manejar los problemas y complejidades que van apareciendo en el mundo de la programación, a medida las exigencias van creciendo.

Creando así, en la comunidad de desarrolladores, un abanico de posibilidades para una creación cada vez más evolucionada de aplicaciones.



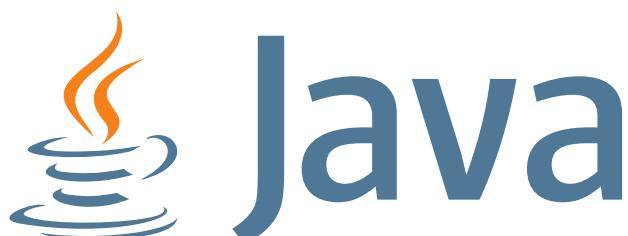
4 Lenguajes Utilizados

Para la realización de esta aplicación web se han utilizado fundamentalmente 2 lenguajes:

Java

Java es un lenguaje de programación orientado a objetos que se incorporó al ámbito de la informática en los años noventa. La idea de Java es que pueda realizarse programas con la posibilidad de ejecutarse en cualquier contexto, en cualquier ambiente, siendo así su portabilidad uno de sus principales logros.

Fue desarrollado por Sun Microsystems, posteriormente adquirido por Oracle. En la actualidad puede utilizarse de modo gratuito, pudiéndose conseguir sin problemas un paquete para desarrolladores que oriente la actividad de programar en este lenguaje. Puede ser modificado por cualquiera, circunstancia que lo convierte en lo que comúnmente se denomina “código abierto”.



JavaScript

JavaScript es el lenguaje de programación encargado de dotar de mayor interactividad y dinamismo a las páginas web. Cuando JavaScript se ejecuta en el navegador, no necesita de un compilador. El navegador lee directamente el código, sin necesidad de terceros. Por tanto, se le reconoce como uno de los tres lenguajes nativos de la web junto a HTML (contenido y su estructura) y a CSS (diseño del contenido y su estructura).



5 Fases del Proyecto

Esta sección expondremos los diferentes ciclos que ha tenido el proyecto:

1 Idea

El equipo, después de pensar varias ideas para realizar su trabajo fin de grado (TFG). Nos surgen algunas ideas, las cuales eran montar un blog donde usuario intercambien información sobre diversos temas o una web de un restaurante de comida. Después de meditarlo optamos por realizar un proyecto que auné estos conceptos y sea algo actual, por lo que se opto por realizar una aplicación web de gestión médica.

2 Casos de Uso

El diagrama de casos de uso es una forma de diagrama de comportamiento en lenguaje de modelado unificado (UML, del inglés Unified Modelling Language), con la que se representan procesos empresariales, así como sistemas y procesos de programación orientada a objetos.

En nuestro proyecto los actores involucrados en los casos de uso son: Paciente y Médico

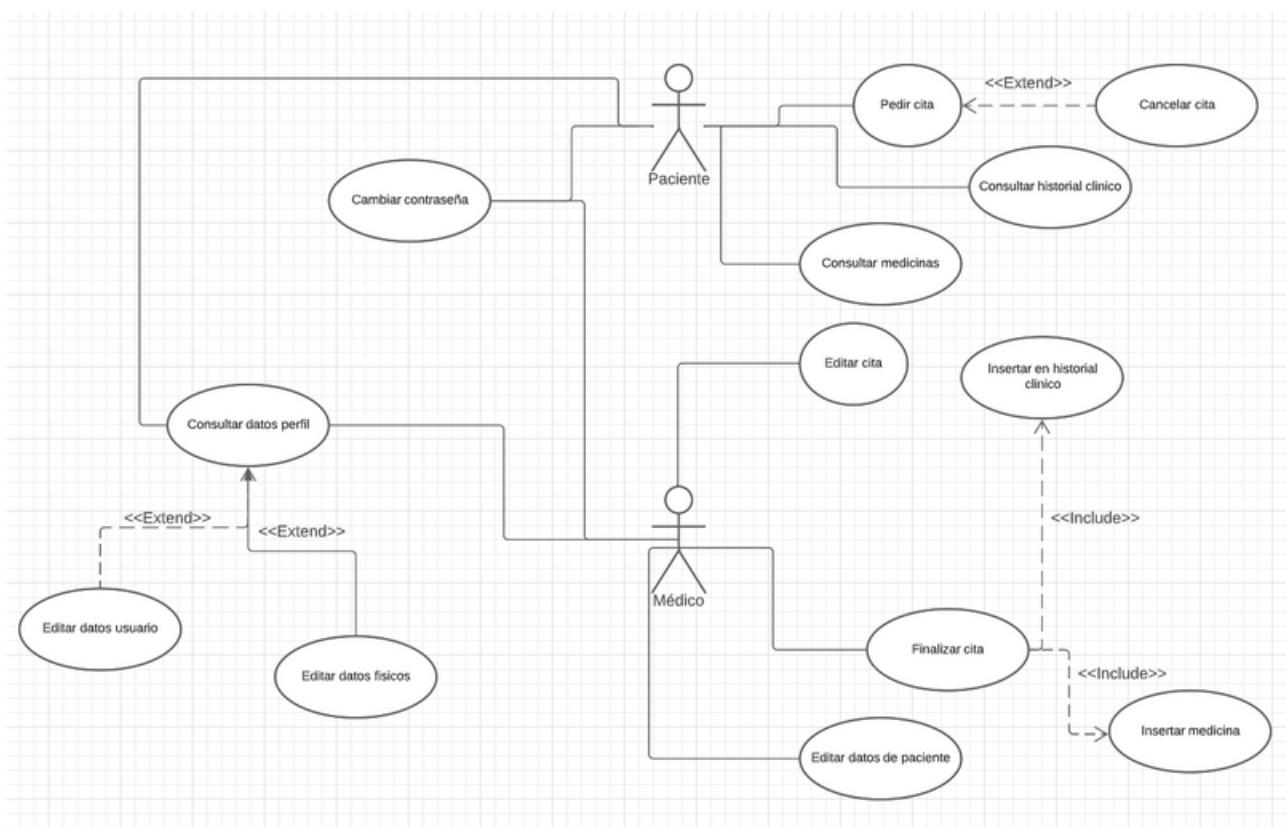
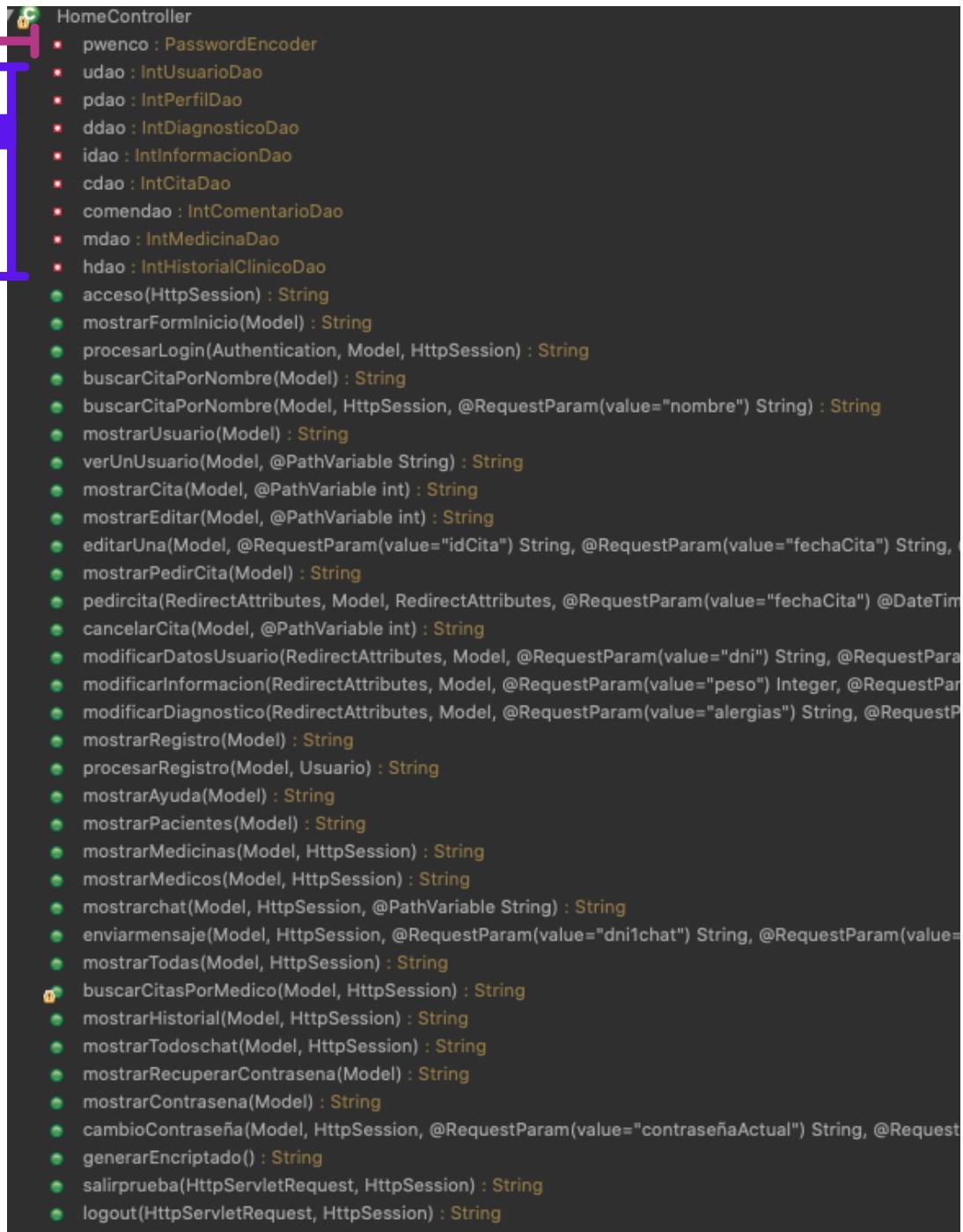
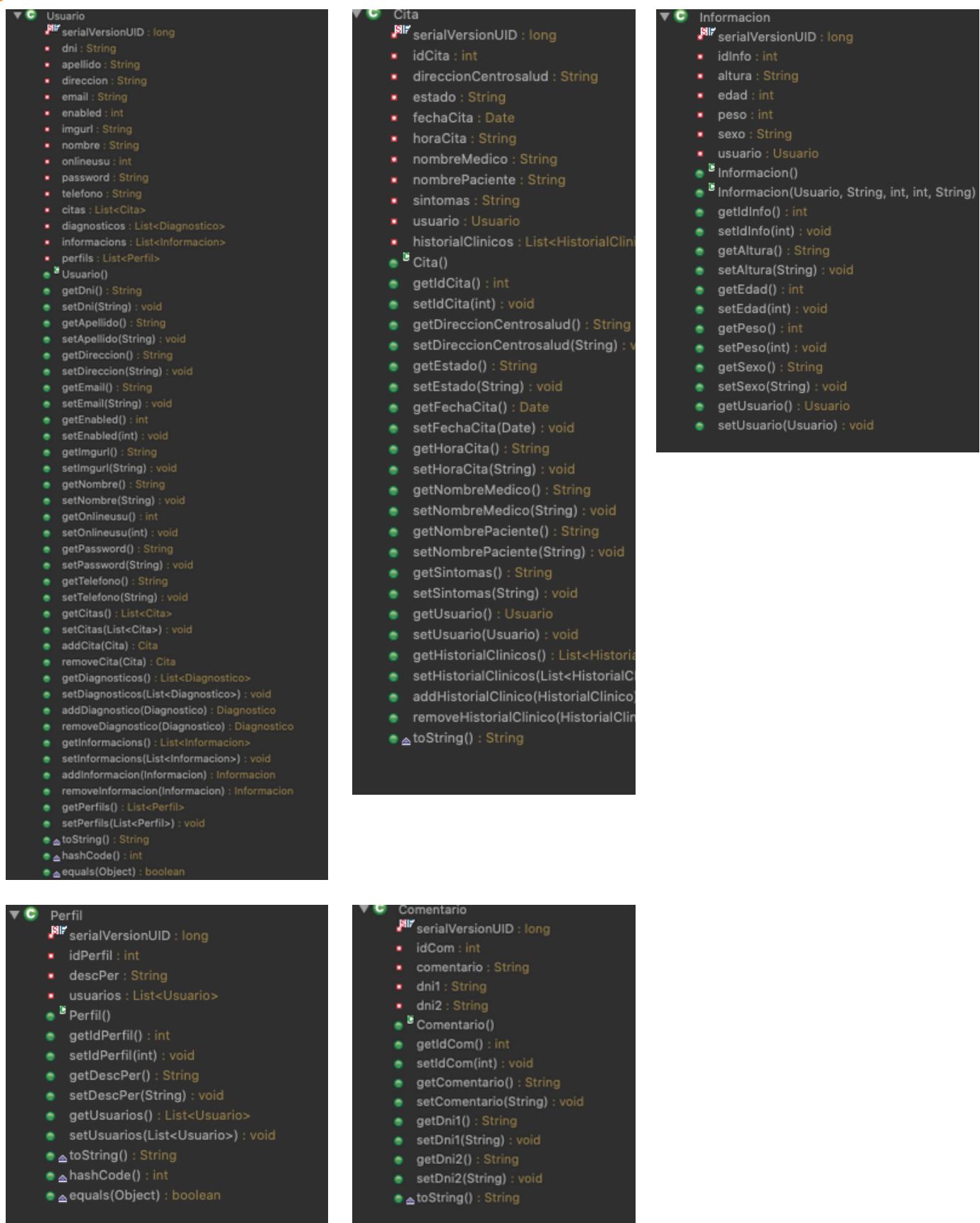


Diagrama de clases

Controller



Beans



Beans

```
▼ C HistorialClinico
  ✓ serialVersionUID : long
  □ idHistorial : int
  □ desCitahis : String
  □ instrucciones : String
  □ cita : Cita
  □ medicina : Medicina
  □ HistorialClinico()
  □ HistorialClinico(String, String, Cita, Medicina)
  □ getIdHistorial() : int
  □ setIdHistorial(int) : void
  □ getDesCitahis() : String
  □ setDesCitahis(String) : void
  □ getInstrucciones() : String
  □ setInstrucciones(String) : void
  □ getCita() : Cita
  □ setCita(Cita) : void
  □ getMedicina() : Medicina
  □ setMedicina(Medicina) : void
```

```
▼ C Diagnostico
  ✓ serialVersionUID : long
  □ idDiag : int
  □ alergias : String
  □ enfermedades : String
  □ operaciones : String
  □ tratamiento : String
  □ usuario : Usuario
  □ Diagnostico()
  □ Diagnostico(String, String, String, String, Usuario)
  □ getIdDiag() : int
  □ setIdDiag(int) : void
  □ getAlergias() : String
  □ setAlergias(String) : void
  □ getEnfermedades() : String
  □ setEnfermedades(String) : void
  □ getOperaciones() : String
  □ setOperaciones(String) : void
  □ getTratamiento() : String
  □ setTratamiento(String) : void
  □ getUsuario() : Usuario
  □ setUsuario(Usuario) : void
```

```
▼ C Medicina
  ✓ serialVersionUID : long
  □ codMed : int
  □ activo : String
  □ cantidad : int
  □ fechaFin : Date
  □ fechalinicio : Date
  □ nombreMed : String
  □ historialClinicos : List<HistorialClinico>
  □ tipo : Tipo
  □ Medicina()
  □ getCodMed() : int
  □ setCodMed(int) : void
  □ getActivo() : String
  □ setActivo(String) : void
  □ getCantidad() : int
  □ setCantidad(int) : void
  □ getFechaFin() : Date
  □ setFechaFin(Date) : void
  □ getFechalinicio() : Date
  □ setFechalinicio(Date) : void
  □ getNombreMed() : String
  □ setNombreMed(String) : void
  □ getHistorialClinicos() : List<HistorialClinico>
  □ setHistorialClinicos(List<HistorialClinico>) : void
  □ addHistorialClinico(HistorialClinico) : HistorialClinico
  □ removeHistorialClinico(HistorialClinico) : HistorialClinico
  □ getTipo() : Tipo
  □ setTipo(Tipo) : void
```

```
▼ C Tipo
  ✓ serialVersionUID : long
  □ idTipo : int
  □ descripcion : String
  □ medicinas : List<Medicina>
  □ Tipo()
  □ getIdTipo() : int
  □ setIdTipo(int) : void
  □ getDescripcion() : String
  □ setDescripcion(String) : void
  □ getMedicinas() : List<Medicina>
  □ setMedicinas(List<Medicina>) : void
  □ addMedicina(Medicina) : Medicina
  □ removeMedicina(Medicina) : Medicina
```

Modelo DAO

```

▼ I IntCitaDao
  ▲ buscarCitas(String) : List<Cita>
  ▲ buscarTodos() : List<Cita>
  ▲ insertUna(Cita) : int
  ▲ buscarUnaCita(int) : Cita
  ▲ buscarCitasPorMedico(String) : List<Cita>
  ▲ buscarCitaPorNombre(String) : List<Cita>
  ▲ editarCita(Cita) : int

```

```

▼ C CitaDaoImplMy8
  □ citarepo : CitaRepo
  ▲ buscarCitas(String) : List<Cita>
  ▲ buscarTodos() : List<Cita>
  ▲ buscarCitaPorNombre(String) : List<Cita>
  ▲ insertUna(Cita) : int
  ▲ buscarUnaCita(int) : Cita
  ▲ buscarCitasPorMedico(String) : List<Cita>
  ▲ editarCita(Cita) : int

```

```

▼ I IntComentarioDao
  ▲ mostrarChat(String, String) : List<Comentario>
  ▲ enviarComentario(Comentario) : int

```

```

▼ C ComentarioDaoImplMy8
  □ comentariorepo : ComentarioRepo
  ▲ mostrarChat(String, String) : List<Comentario>
  ▲ enviarComentario(Comentario) : int

```

```

▼ I IntDiagnosticoDao
  ▲ buscarDiagnostico(String) : Diagnostico
  ▲ buscarTodos() : List<Diagnostico>
  ▲ insertUno(Diagnostico) : int
  ▲ editarDiagnostico(Diagnostico) : int

```

```

▼ C DiagnosticoDaoImplMy8
  □ diarepo : DiagnosticoRepo
  ▲ buscarDiagnostico(String) : Diagnostico
  ▲ buscarTodos() : List<Diagnostico>
  ▲ insertUno(Diagnostico) : int
  ▲ editarDiagnostico(Diagnostico) : int

```

```

▼ I IntHistorialClinicoDao
  ▲ insertUna(HistorialClinico) : int
  ▲ buscarHistorialClinicoPordni(String) : List<HistorialClinico>

```

```

▼ C HistorialClinicoDaoImplMy8
  □ historialrepo : HistorialClinicoRepo
  ▲ insertUna(HistorialClinico) : int
  ▲ buscarHistorialClinicoPordni(String) : List<HistorialClinico>

```

```

▼ I IntInformacionDao
  ▲ buscarInformacion(String) : Informacion
  ▲ buscarTodos() : List<Informacion>
  ▲ insertUno(Informacion) : int
  ▲ editarInformacion(Informacion) : int

```

```

▼ C InformacionDaoImplMy8
  □ infrepo : InformacionRepo
  ▲ buscarInformacion(String) : Informacion
  ▲ buscarTodos() : List<Informacion>
  ▲ insertUno(Informacion) : int
  ▲ editarInformacion(Informacion) : int

```

```

▼ I IntMedicinaDao
  ▲ buscarTodasMedicinas() : List<Medicina>
  ▲ buscarUnaMedicina(String) : Medicina
  ▲ buscarMedicinasUsuario(String) : List<Medicina>

```

```

▼ C MedicinaDaoImplMy8
  □ medirepo : MedicinaRepo
  ▲ buscarTodasMedicinas() : List<Medicina>
  ▲ buscarUnaMedicina(String) : Medicina
  ▲ buscarMedicinasUsuario(String) : List<Medicina>

```

```

▼ I IntPerfilDao
  ▲ buscarPerfil(int) : Perfil
  ▲ buscarTodos() : List<Perfil>
  ▲ insertUno(Perfil) : int

```

```

▼ C PerfilDaoImplMy8
  □ pfrepo : PerfilRepo
  ▲ buscarPerfil(int) : Perfil
  ▲ buscarTodos() : List<Perfil>
  ▲ insertUno(Perfil) : int

```

```

▼ I IntUsuarioDao
  ▲ buscarUsuario(String) : Usuario
  ▲ buscarTodos() : List<Usuario>
  ▲ buscarMedicos(String) : List<Usuario>
  ▲ buscarConectados(String) : List<Usuario>
  ▲ buscarPacientes() : List<Usuario>
  ▲ listaMedicos() : List<Usuario>
  ▲ contarMedicos() : Integer
  ▲ insertUno(Usuario) : int
  ▲ cambioContraseña(String, String) : void
  ▲ buscarUsuarioPorNombre(String) : Usuario
  ▲ editarUsuario(Usuario) : int

```

```

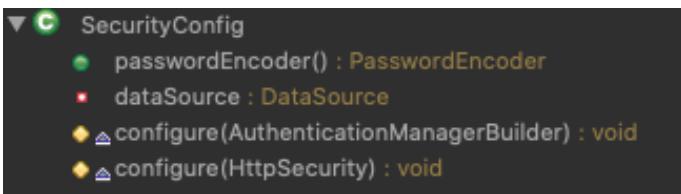
▼ C UsuarioDaoImplMy8
  □ urepo : UsuarioRepo
  ▲ buscarUsuario(String) : Usuario
  ▲ buscarTodos() : List<Usuario>
  ▲ cambioContraseña(String, String) : void
  ▲ insertUno(Usuario) : int
  ▲ buscarUsuarioPorNombre(String) : Usuario
  ▲ editarUsuario(Usuario) : int
  ▲ contarMedicos() : Integer
  ▲ buscarConectados(String) : List<Usuario>
  ▲ buscarPacientes() : List<Usuario>
  ▲ buscarMedicos(String) : List<Usuario>
  ▲ listaMedicos() : List<Usuario>

```

Modelo repository

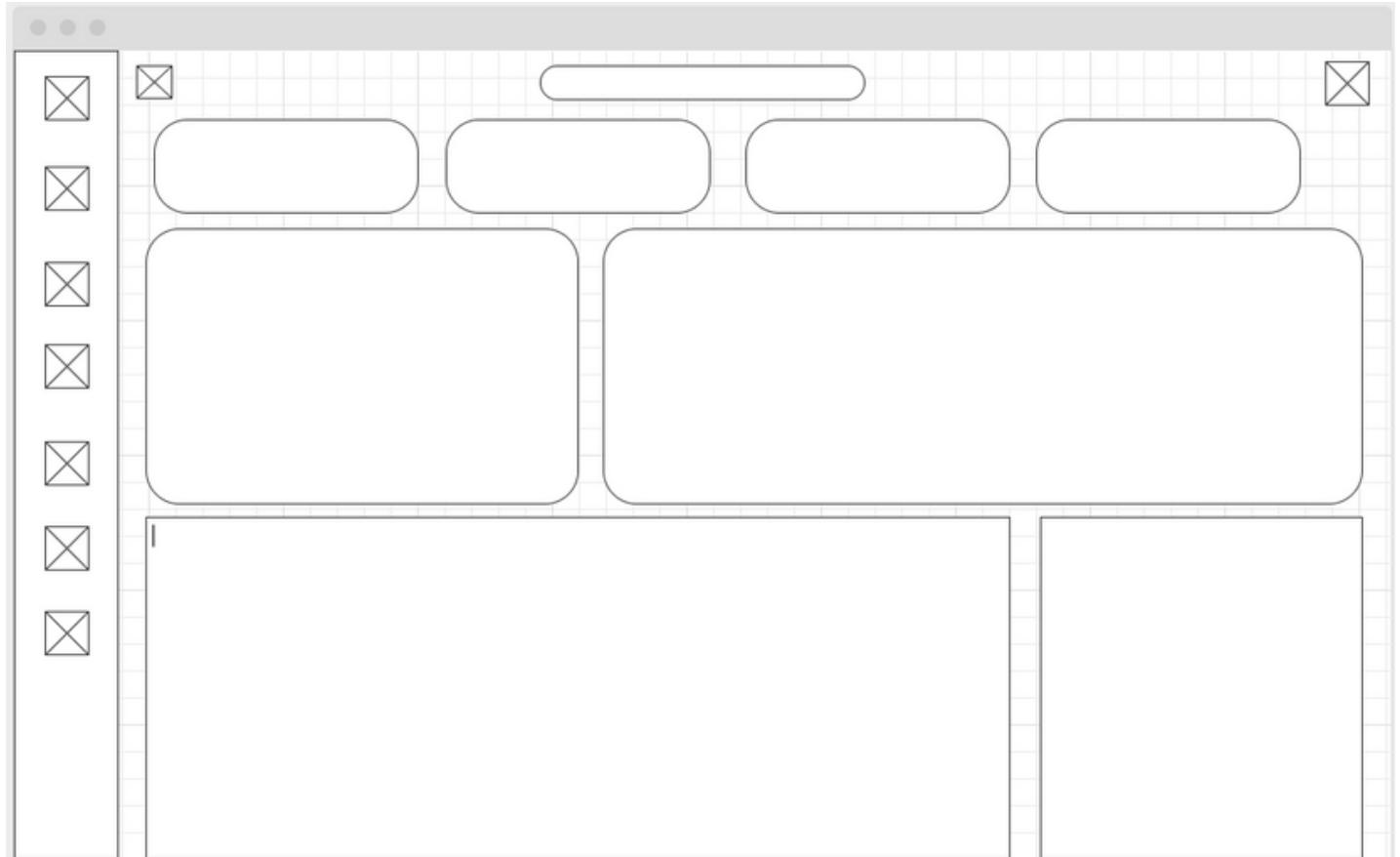


Security



3 WireFrame

Para hacer el wireframe he utilizado la herramienta de figma para crear como será la interfaz de usuario (UI) y teniendo en consideración la experiencia de usuario (UX), con la finalidad de crear un diseño simple y que te aporte la información precisa a el alcance de un click, primando los bordes redondeados y diseñando también teniendo muy en cuenta los dispositivos móviles, haciendo uso de diseño enteramente responsivo.



4 Guía de Estilos

Las guías de estilo, también denominadas libros de estilo, hacen referencia en el ámbito del marketing online a los manuales para el diseño uniforme de páginas web. Aquí, muestro la guía de estilo de este proyecto.

Logo



Colores:

Realizado #8de02c



Pendiente #f9ca3f

Cancelado #f00

Enprogreso #1795c9

--blue: #287bff

--white: #fff

--grey: #f5f5f5

--black1: #222

--black2: #999

Tipografía: 'Poppins', sans-serif;

Titulos

Parrafo

font-size: 2em

font-weight: 500

Iconos (ion-icon):



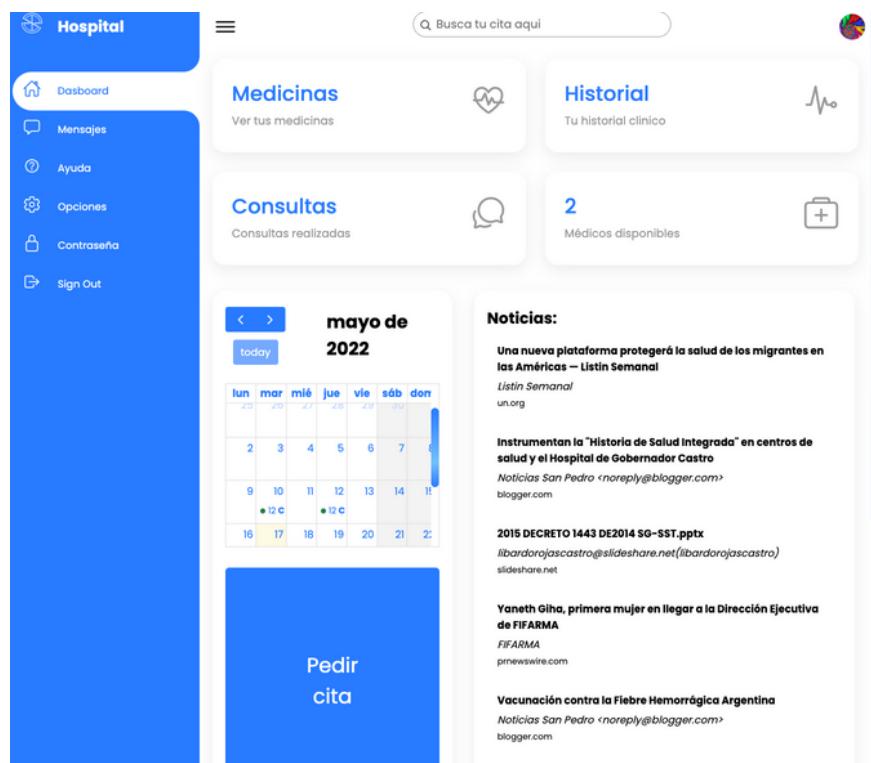
pulse-outline

<ion-icon name="pulse-outline"></ion-icon>

Estilo de cajas:

border-radius: 20px

box-shadow: 0 7px 25px rgb(0 0 0 / 8%)



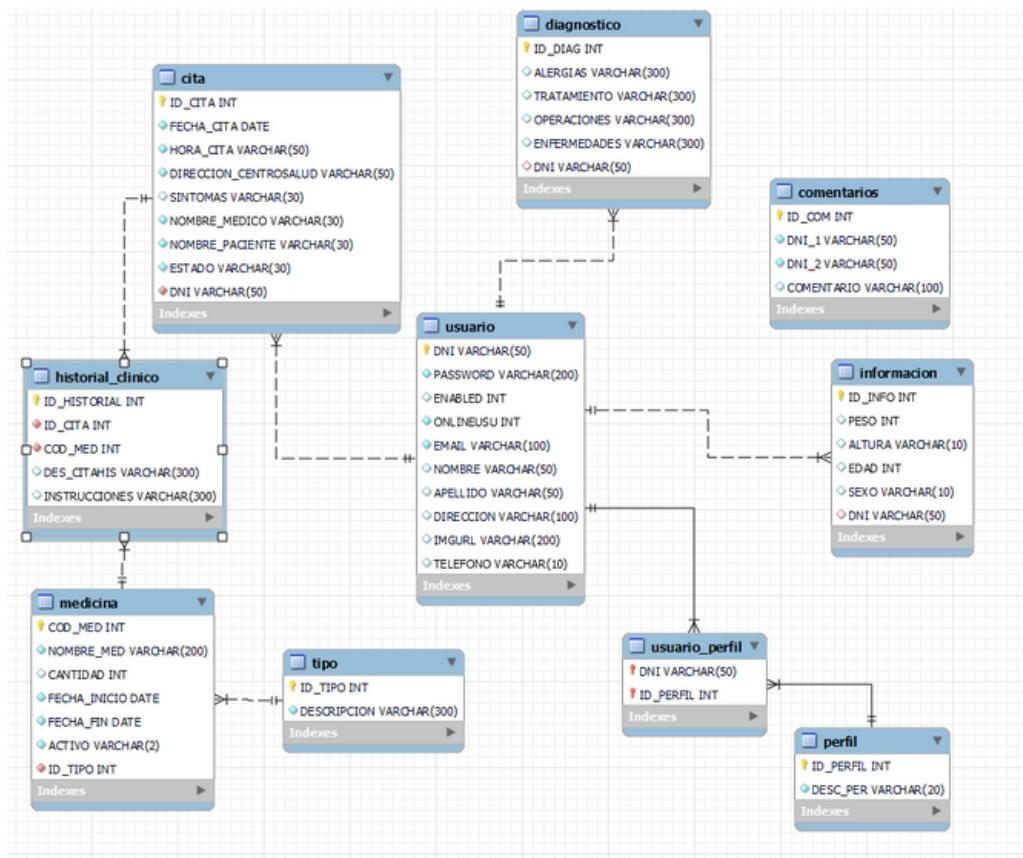
5 Bases de Datos

Como nuestra aplicación va a tener: usuarios con sus respectivos datos, medicinas que se pueden recetar, las citas de cuando te podrán atender, el historial medico de cada usuario, necesitaremos un sitio donde poder almacenar esos datos.



Por eso tenemos que crear una Base de datos utilizando SQL como lenguaje y MySQL Workbench como herramienta además de poder almacenar los datos también podemos: acceder, modificarlos, o guardar nuevos registros.

Para este caso se decidió crear un esquema de datos que reúna los datos que vemos necesarios para el mejor funcionamiento de la aplicación.



Hemos utilizado **DDL** (Lenguaje de Definición de Datos) para crear la base de datos y **DML** (Lenguaje de Manipulación de Datos) para insertar y manipular la información.

```
CREATE TABLE IF NOT EXISTS `hospital_bbdd`.`usuario` (
  `DNI` VARCHAR(50) NOT NULL,
  `PASSWORD` VARCHAR(200) NOT NULL,
  `ENABLED` TINYINT DEFAULT NULL,
  `ONLINEUSU` INT NOT NULL,
  `EMAIL` VARCHAR(100) NOT NULL,
  `NOMBRE` VARCHAR(50) NULL DEFAULT NULL,
  `APELLIDO` VARCHAR(50) NULL DEFAULT NULL,
  `DIRECCION` VARCHAR(100) NULL DEFAULT NULL,
  `IMGURL` VARCHAR(200) NULL DEFAULT NULL,
  `TELEFONO` VARCHAR(10) NULL DEFAULT NULL,
  PRIMARY KEY (`DNI`),
  UNIQUE INDEX `EMAIL` (`EMAIL` ASC) VISIBLE
)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_spanish_ci;
```

DDL

```
INSERT INTO `hospital_bbdd`.`informacion`(`PESO`, `ALTURA`, `EDAD`, `SEXO`, `DNI`) VALUES ('23', '1,23', '23', 'm', '34728920e');
INSERT INTO `hospital_bbdd`.`informacion`(`PESO`, `ALTURA`, `EDAD`, `SEXO`, `DNI`) VALUES ('34', '1,45', '32', 'f', '1232500');
INSERT INTO `hospital_bbdd`.`informacion`(`PESO`, `ALTURA`, `EDAD`, `SEXO`, `DNI`) VALUES ('78', '1,80', '32', 'm', '5833942');
INSERT INTO `hospital_bbdd`.`informacion`(`PESO`, `ALTURA`, `EDAD`, `SEXO`, `DNI`) VALUES ('65', '1,70', '32', 'm', '6574848e');
INSERT INTO `hospital_bbdd`.`informacion`(`PESO`, `ALTURA`, `EDAD`, `SEXO`, `DNI`) VALUES ('68', '1,78', '25', 'm', '71706550e');
```

DML

El código Sql que se utilizó para crear la bbdd se localiza en la misma carpeta donde está este pdf.

6 HTML

HTML (HyperText Markup Language) es un lenguaje de marcas utilizado para crear páginas web con este lenguaje podremos mostrar el contenido de la web.



Esto se hace utilizando como herramienta el Visual Studio Code, no es obligatorio usar esta herramienta pero es bastante cómoda.

Algunos ejemplos del código quedarían así,

Código del index.html

```
<body>
  <div class="container">
    <div class="navegation">
      <ul>
        <li>
          <a href="/index">
            <span class="icon"></span>
            <span class="title">
              <h2>Hospital</h2>
            </span>
          </a>
        </li>
        <li class="hovered">
          <a href="/index">
            <span class="icon">
              <ion-icon name="home-outline"></ion-icon>
            </span>
            <span class="title">Dashboard</span>
          </a>
        </li>
        <sec:authorize access="hasAuthority('Medico')">
        <li>
          <a href="/pacientes">
            <!-- Mostrar solo a médico -->
            <span class="icon">
              <ion-icon name="people-outline"></ion-icon>
            </span>
            <span class="title">Pacientes</span>
          </a>
        </li>
        </sec:authorize>
        <li>
          <a href="/todoschat">
            <span class="icon">
              <ion-icon name="chatbox-outline"></ion-icon>
            </span>
            <span class="title">Mensajes</span>
          </a>
        </li>
        <li>
```

Así se vería sin ningún tipo de estilo CSS

Hospital

- [Dashboard](#)
- [Pacientes](#)
- [Mensajes](#)
- [Ayuda](#)
- [Opciones](#)
- [Contraseña](#)
- [Sign Out](#)

Como se puede apreciar de esta manera no se vería muy agradable para el usuario por eso le tendríamos que añadir el estilo utilizando CSS

7

CSS

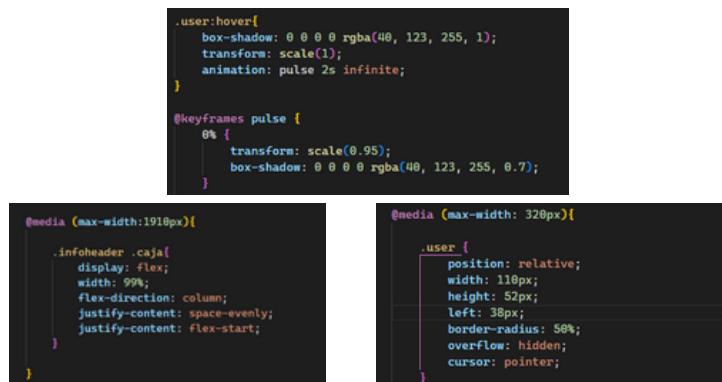
Para los estilos de este proyecto se contará con 3 archivos css los cuales son:

- style.css
- stylelogin.css
- styleusuario.css

También se han realizado pequeños ajustes puntuales e individuales utilizando <style> dentro del html, para realizar un diseño lo más responsive posible se ha utilizado el inspector del navegador para ir creando las mediaqueries necesarias.

Igualmente también se han utilizado keyframes para la animación del icono y efectos de hover para recalcar botones o campos interactivos.

Las mediaqueries van desde los @media (max-width:1910px) a los @media (max-width: 320px)



8

JavaScript

En este proyecto, hemos utilizado el JavaScript con el objetivo de dotar funcionalidad a la página, lo hemos utilizado para realizar validaciones de formularios tanto en la navegación del propio menú de la página, como para mostrar alertas personalizadas utilizando:

```

//----- Revisar BOTON Mostrar solo cuando no haya datos de informacion d
Swal.fire({
  title: "<strong>¿Quieres rellenar tus datos de <u>perfil</u>?</strong>",
  icon: "info",
  html: 
    "<strong>Rellene estos <b>datos</b>, " + "para mejorar nuestros analisis.</strong>",
  showCloseButton: true,
  focusConfirm: false,
  confirmButtonText:
    '<a href="/usuario.html" class="btn">Rellenar datos personales</a> ',
})

```

```

//ejecutar la funcion validar() cuando se pulse el boton de registro ----- NO VA!
function validar() {
  var dni = document.getElementById("dni").value;
  var password = document.getElementById("password").value;
  var email = document.getElementById("email").value;
  var nombre = document.getElementById("nombre").value;
  var apellidos = document.getElementById("apellidos").value;
  var direccion = document.getElementById("direccion").value;
  var telefono = document.getElementById("telefono").value;
  var check = document.getElementById("check").checked;
  if (dni == "" || password == "" || email == "" || nombre == "" || apellidos == "" || check == false) {
    Swal.fire({
      icon: 'error',
      title: 'Oops...',
      text: 'Debes llenar todos los campos',
    })
    return false;
  } else {
    return true;
  }
}

```

```

let toggle = document.querySelector(".toggle");
let navigation = document.querySelector(".navigation");
let main = document.querySelector(".main");

toggle.onclick = function () {
  navigation.classList.toggle("active");
  main.classList.toggle("active");
}

let list = document.querySelectorAll(".navigation li");

function activeLink() {
  list.forEach((item) => item.classList.remove("hovered"));
  this.classList.add("hovered");
}

list.forEach((item) => item.addEventListener("mouseover", activeLink));

```

9 API's

En la parte del desarrollo front hemos incorporado una api que muestra las citas que tiene cada usuario en un calendario utilizando la api: <https://fullcalendar.io/demos> y también realizamos otra llamada a otra api, que nos muestra un panel de noticias relacionadas con sanidad, con el fin de enriquecer el contenido de la web utilizando la librería axios para manejar las peticiones AJAX: <https://rapidapi.com/search/news> e insertando esta información utilizando la creación de elementos textos y nodos de js

The screenshot shows a calendar for May 2022. The days of the week are labeled from Monday (lun) to Sunday (dom). Specific events are marked: '● 13 Cita' on May 3rd, '● 11 Cita M' on May 11th, 'Tratamiento' on May 18th, and '● 20 Oper' on May 20th. A blue sidebar on the left contains a 'Pedir cita' button.

Noticias:

Una nueva plataforma protegerá la salud de los migrantes en las Américas — Listín Semanal

Listín Semanal
un.org

Instrumentan la "Historia de Salud Integrada" en centros de salud y el Hospital de Gobernador Castro

Noticias San Pedro <noreply@blogger.com>
blogger.com

2015 DECRETO 1443 DE2014 SG-SST.pptx

libardorojascastro@slideshare.net(libardorojascastro)
slideshare.net

Vacunación contra la Fiebre Hemorrágica Argentina

Noticias San Pedro <noreply@blogger.com>
blogger.com

[Editorial] Innovación revolucionaria para entusiastas de la salud y el bienestar

null
samsung.com

Yaneth Giha, primera mujer en llegar a la Dirección Ejecutiva de FIFARMA

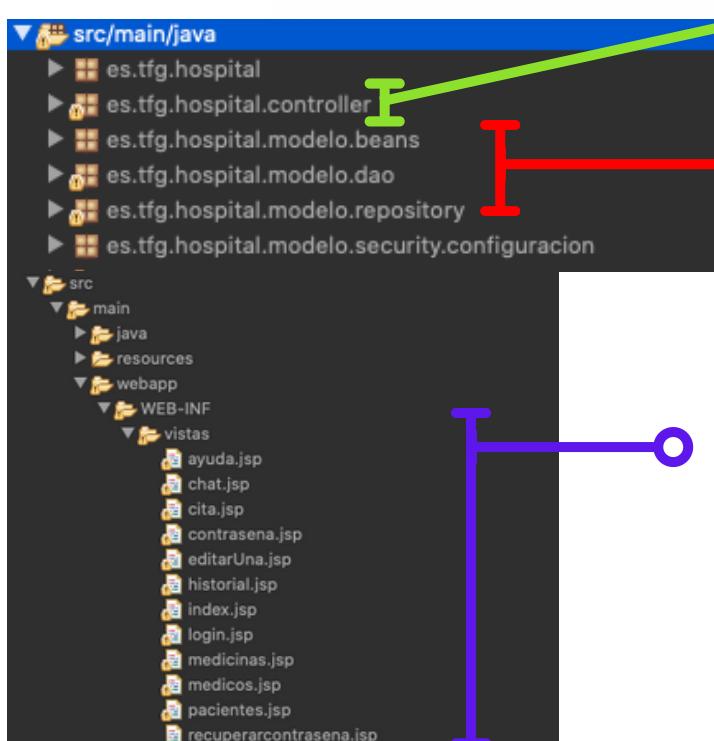
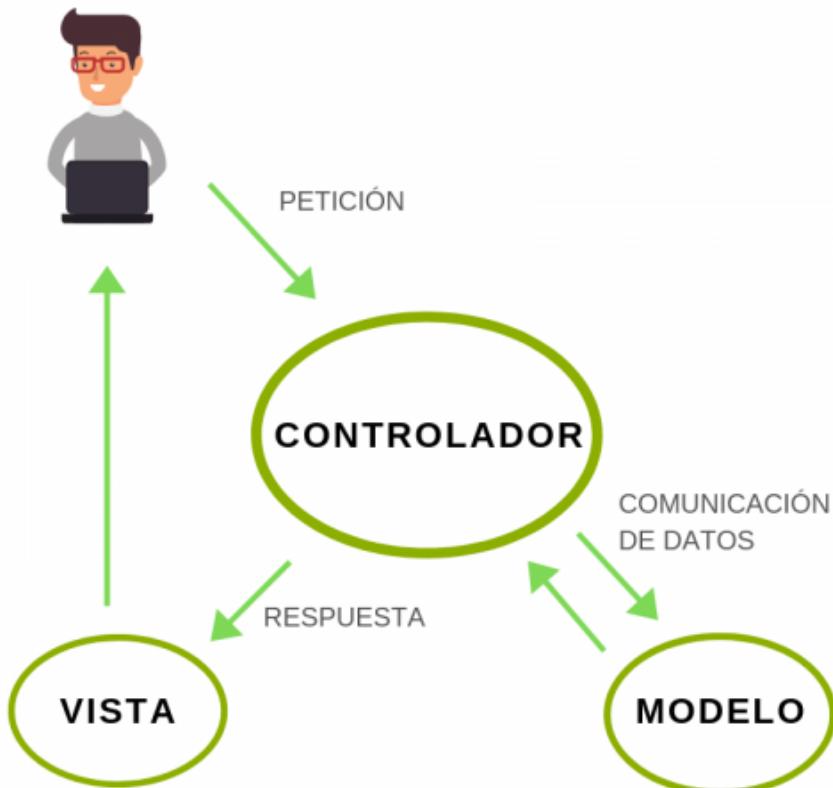
FIFARMA
prnewswire.com

```
C:\Users\Win10\Desktop>TGF>TGFFront> index.html
365  today = yyyy + "-" + mm + "-" + dd;
366
367  document.addEventListener("DOMContentLoaded", function () {
368    var calendarEl = document.getElementById("calendar");
369    //----- Revisar -----
370    var calendar = new FullCalendar.Calendar(calendarEl, {
371      locale: "es",
372      firstDay: 1,
373      headerToolbar: {
374        left: "prev,next today",
375        center: "title",
376        right: ""
377      },
378      //right: "dayGridMonth,timeGridWeek,timeGridDay,listMonth",
379      //right: "addEventButton",
380    },
381    customButtons: {
382      addEventButton: {
383        text: "Pide tu cita",
384        click: function () {
385          //Idea : añadir la funcion como externa (SE DERIVARA A UN JSP)
386          //Revisar poner opciones
387          var dateStr = prompt("Enter a date in YYYY-MM-DD format");
388          var titleStr = prompt("Enter a titulo");
389          var horaStr = prompt("Enter a hora");
390          var date = new Date(dateStr + "T" + horaStr + ":00"); // will be in local time
391          var titulo = titleStr + " " + horaStr;
392          location.href = "ir a una pagina de cita";
393          if (!isNaN(date.valueOf())) {
394            // valid?
395            calendar.addEvent({
396              title: titulo,
397              start: date,
398              allDay: true,
399            });
399            // Ejecutar solo si viene de poner cita
400            const Toast = Swal.mixin({
401              toast: true,
402              position: "top-end",
403              showConfirmButton: false,
404              timer: 3000,
405              timerProgressBar: true,
406              didOpen: (toast) => {
407                toast.addEventListener("mouseenter", Swal.stopTimer);
408                toast.addEventListener("mouseleave", Swal.resumeTimer);
409              },
410            });
411            // Ejecutar solo si viene de poner cita
412            Toast.fire();
413        }
414      }
415    }
416  });
417
418  // Ejecutar solo si viene de poner cita
419  const Toast = Swal.mixin({
420    toast: true,
421    position: "top-end",
422    showConfirmButton: false,
423    timer: 3000,
424    timerProgressBar: true,
425    didOpen: (toast) => {
426      toast.addEventListener("mouseenter", Swal.stopTimer);
427      toast.addEventListener("mouseleave", Swal.resumeTimer);
428    },
429  });
430
431  // Ejecutar solo si viene de poner cita
432  Toast.fire();
433
434  // Ejecutar solo si viene de poner cita
435  const Toast = Swal.mixin({
436    toast: true,
437    position: "top-end",
438    showConfirmButton: false,
439    timer: 3000,
440    timerProgressBar: true,
441    didOpen: (toast) => {
442      toast.addEventListener("mouseenter", Swal.stopTimer);
443      toast.addEventListener("mouseleave", Swal.resumeTimer);
444    },
445  });
446
447  // Ejecutar solo si viene de poner cita
448  const Toast = Swal.mixin({
449    toast: true,
450    position: "top-end",
451    showConfirmButton: false,
452    timer: 3000,
453    timerProgressBar: true,
454    didOpen: (toast) => {
455      toast.addEventListener("mouseenter", Swal.stopTimer);
456      toast.addEventListener("mouseleave", Swal.resumeTimer);
457    },
458  });
459
460  // Ejecutar solo si viene de poner cita
461  const Toast = Swal.mixin({
462    toast: true,
463    position: "top-end",
464    showConfirmButton: false,
465    timer: 3000,
466    timerProgressBar: true,
467    didOpen: (toast) => {
468      toast.addEventListener("mouseenter", Swal.stopTimer);
469      toast.addEventListener("mouseleave", Swal.resumeTimer);
470    },
471  });
472
473  // Ejecutar solo si viene de poner cita
474  const Toast = Swal.mixin({
475    toast: true,
476    position: "top-end",
477    showConfirmButton: false,
478    timer: 3000,
479    timerProgressBar: true,
480    didOpen: (toast) => {
481      toast.addEventListener("mouseenter", Swal.stopTimer);
482      toast.addEventListener("mouseleave", Swal.resumeTimer);
483    },
484  });
485
486  // Ejecutar solo si viene de poner cita
487  const Toast = Swal.mixin({
488    toast: true,
489    position: "top-end",
490    showConfirmButton: false,
491    timer: 3000,
492    timerProgressBar: true,
493    didOpen: (toast) => {
494      toast.addEventListener("mouseenter", Swal.stopTimer);
495      toast.addEventListener("mouseleave", Swal.resumeTimer);
496    },
497  });
498
499  // Ejecutar solo si viene de poner cita
500  const Toast = Swal.mixin({
501    toast: true,
502    position: "top-end",
503    showConfirmButton: false,
504    timer: 3000,
505    timerProgressBar: true,
506    didOpen: (toast) => {
507      toast.addEventListener("mouseenter", Swal.stopTimer);
508      toast.addEventListener("mouseleave", Swal.resumeTimer);
509    },
510  });
511
512  // Ejecutar solo si viene de poner cita
513  const Toast = Swal.mixin({
514    toast: true,
515    position: "top-end",
516    showConfirmButton: false,
517    timer: 3000,
518    timerProgressBar: true,
519    didOpen: (toast) => {
520      toast.addEventListener("mouseenter", Swal.stopTimer);
521      toast.addEventListener("mouseleave", Swal.resumeTimer);
522    },
523  });
524
525  // Ejecutar solo si viene de poner cita
526  const Toast = Swal.mixin({
527    toast: true,
528    position: "top-end",
529    showConfirmButton: false,
530    timer: 3000,
531    timerProgressBar: true,
532    didOpen: (toast) => {
533      toast.addEventListener("mouseenter", Swal.stopTimer);
534      toast.addEventListener("mouseleave", Swal.resumeTimer);
535    },
536  });
537
538  // Ejecutar solo si viene de poner cita
539  const Toast = Swal.mixin({
540    toast: true,
541    position: "top-end",
542    showConfirmButton: false,
543    timer: 3000,
544    timerProgressBar: true,
545    didOpen: (toast) => {
546      toast.addEventListener("mouseenter", Swal.stopTimer);
547      toast.addEventListener("mouseleave", Swal.resumeTimer);
548    },
549  });
550
551  // Ejecutar solo si viene de poner cita
552  const Toast = Swal.mixin({
553    toast: true,
554    position: "top-end",
555    showConfirmButton: false,
556    timer: 3000,
557    timerProgressBar: true,
558    didOpen: (toast) => {
559      toast.addEventListener("mouseenter", Swal.stopTimer);
560      toast.addEventListener("mouseleave", Swal.resumeTimer);
561    },
562  });
563
564  // Ejecutar solo si viene de poner cita
565  const Toast = Swal.mixin({
566    toast: true,
567    position: "top-end",
568    showConfirmButton: false,
569    timer: 3000,
570    timerProgressBar: true,
571    didOpen: (toast) => {
572      toast.addEventListener("mouseenter", Swal.stopTimer);
573      toast.addEventListener("mouseleave", Swal.resumeTimer);
574    },
575  });
576
577  // Ejecutar solo si viene de poner cita
578  const Toast = Swal.mixin({
579    toast: true,
580    position: "top-end",
581    showConfirmButton: false,
582    timer: 3000,
583    timerProgressBar: true,
584    didOpen: (toast) => {
585      toast.addEventListener("mouseenter", Swal.stopTimer);
586      toast.addEventListener("mouseleave", Swal.resumeTimer);
587    },
588  });
589
590  // Ejecutar solo si viene de poner cita
591  const Toast = Swal.mixin({
592    toast: true,
593    position: "top-end",
594    showConfirmButton: false,
595    timer: 3000,
596    timerProgressBar: true,
597    didOpen: (toast) => {
598      toast.addEventListener("mouseenter", Swal.stopTimer);
599      toast.addEventListener("mouseleave", Swal.resumeTimer);
600    },
601  });
602
603  // Ejecutar solo si viene de poner cita
604  const Toast = Swal.mixin({
605    toast: true,
606    position: "top-end",
607    showConfirmButton: false,
608    timer: 3000,
609    timerProgressBar: true,
610    didOpen: (toast) => {
611      toast.addEventListener("mouseenter", Swal.stopTimer);
612      toast.addEventListener("mouseleave", Swal.resumeTimer);
613    },
614  });
615
616  // Ejecutar solo si viene de poner cita
617  const Toast = Swal.mixin({
618    toast: true,
619    position: "top-end",
620    showConfirmButton: false,
621    timer: 3000,
622    timerProgressBar: true,
623    didOpen: (toast) => {
624      toast.addEventListener("mouseenter", Swal.stopTimer);
625      toast.addEventListener("mouseleave", Swal.resumeTimer);
626    },
627  });
628
629  // Ejecutar solo si viene de poner cita
630  const Toast = Swal.mixin({
631    toast: true,
632    position: "top-end",
633    showConfirmButton: false,
634    timer: 3000,
635    timerProgressBar: true,
636    didOpen: (toast) => {
637      toast.addEventListener("mouseenter", Swal.stopTimer);
638      toast.addEventListener("mouseleave", Swal.resumeTimer);
639    },
640  });
641
642  // Ejecutar solo si viene de poner cita
643  const Toast = Swal.mixin({
644    toast: true,
645    position: "top-end",
646    showConfirmButton: false,
647    timer: 3000,
648    timerProgressBar: true,
649    didOpen: (toast) => {
650      toast.addEventListener("mouseenter", Swal.stopTimer);
651      toast.addEventListener("mouseleave", Swal.resumeTimer);
652    },
653  });
654
655  // Ejecutar solo si viene de poner cita
656  const Toast = Swal.mixin({
657    toast: true,
658    position: "top-end",
659    showConfirmButton: false,
660    timer: 3000,
661    timerProgressBar: true,
662    didOpen: (toast) => {
663      toast.addEventListener("mouseenter", Swal.stopTimer);
664      toast.addEventListener("mouseleave", Swal.resumeTimer);
665    },
666  });
667
668  // Ejecutar solo si viene de poner cita
669  const Toast = Swal.mixin({
670    toast: true,
671    position: "top-end",
672    showConfirmButton: false,
673    timer: 3000,
674    timerProgressBar: true,
675    didOpen: (toast) => {
676      toast.addEventListener("mouseenter", Swal.stopTimer);
677      toast.addEventListener("mouseleave", Swal.resumeTimer);
678    },
679  });
680
681  // Ejecutar solo si viene de poner cita
682  const Toast = Swal.mixin({
683    toast: true,
684    position: "top-end",
685    showConfirmButton: false,
686    timer: 3000,
687    timerProgressBar: true,
688    didOpen: (toast) => {
689      toast.addEventListener("mouseenter", Swal.stopTimer);
690      toast.addEventListener("mouseleave", Swal.resumeTimer);
691    },
692  });
693
694  // Ejecutar solo si viene de poner cita
695  const Toast = Swal.mixin({
696    toast: true,
697    position: "top-end",
698    showConfirmButton: false,
699    timer: 3000,
700    timerProgressBar: true,
701    didOpen: (toast) => {
702      toast.addEventListener("mouseenter", Swal.stopTimer);
703      toast.addEventListener("mouseleave", Swal.resumeTimer);
704    },
705  });
706
707  // Ejecutar solo si viene de poner cita
708  const Toast = Swal.mixin({
709    toast: true,
710    position: "top-end",
711    showConfirmButton: false,
712    timer: 3000,
713    timerProgressBar: true,
714    didOpen: (toast) => {
715      toast.addEventListener("mouseenter", Swal.stopTimer);
716      toast.addEventListener("mouseleave", Swal.resumeTimer);
717    },
718  });
719
720  // Ejecutar solo si viene de poner cita
721  const Toast = Swal.mixin({
722    toast: true,
723    position: "top-end",
724    showConfirmButton: false,
725    timer: 3000,
726    timerProgressBar: true,
727    didOpen: (toast) => {
728      toast.addEventListener("mouseenter", Swal.stopTimer);
729      toast.addEventListener("mouseleave", Swal.resumeTimer);
730    },
731  });
732
733  // Ejecutar solo si viene de poner cita
734  const Toast = Swal.mixin({
735    toast: true,
736    position: "top-end",
737    showConfirmButton: false,
738    timer: 3000,
739    timerProgressBar: true,
740    didOpen: (toast) => {
741      toast.addEventListener("mouseenter", Swal.stopTimer);
742      toast.addEventListener("mouseleave", Swal.resumeTimer);
743    },
744  });
745
746  // Ejecutar solo si viene de poner cita
747  const Toast = Swal.mixin({
748    toast: true,
749    position: "top-end",
750    showConfirmButton: false,
751    timer: 3000,
752    timerProgressBar: true,
753    didOpen: (toast) => {
754      toast.addEventListener("mouseenter", Swal.stopTimer);
755      toast.addEventListener("mouseleave", Swal.resumeTimer);
756    },
757  });
758
759  // Ejecutar solo si viene de poner cita
760  const Toast = Swal.mixin({
761    toast: true,
762    position: "top-end",
763    showConfirmButton: false,
764    timer: 3000,
765    timerProgressBar: true,
766    didOpen: (toast) => {
767      toast.addEventListener("mouseenter", Swal.stopTimer);
768      toast.addEventListener("mouseleave", Swal.resumeTimer);
769    },
770  });
771
772  // Ejecutar solo si viene de poner cita
773  const Toast = Swal.mixin({
774    toast: true,
775    position: "top-end",
776    showConfirmButton: false,
777    timer: 3000,
778    timerProgressBar: true,
779    didOpen: (toast) => {
780      toast.addEventListener("mouseenter", Swal.stopTimer);
781      toast.addEventListener("mouseleave", Swal.resumeTimer);
782    },
783  });
784
785  // Ejecutar solo si viene de poner cita
786  const Toast = Swal.mixin({
787    toast: true,
788    position: "top-end",
789    showConfirmButton: false,
790    timer: 3000,
791    timerProgressBar: true,
792    didOpen: (toast) => {
793      toast.addEventListener("mouseenter", Swal.stopTimer);
794      toast.addEventListener("mouseleave", Swal.resumeTimer);
795    },
796  });
797
798  // Ejecutar solo si viene de poner cita
799  const Toast = Swal.mixin({
800    toast: true,
801    position: "top-end",
802    showConfirmButton: false,
803    timer: 3000,
804    timerProgressBar: true,
805    didOpen: (toast) => {
806      toast.addEventListener("mouseenter", Swal.stopTimer);
807      toast.addEventListener("mouseleave", Swal.resumeTimer);
808    },
809  });
810
811  // Ejecutar solo si viene de poner cita
812  const Toast = Swal.mixin({
813    toast: true,
814    position: "top-end",
815    showConfirmButton: false,
816    timer: 3000,
817    timerProgressBar: true,
818    didOpen: (toast) => {
819      toast.addEventListener("mouseenter", Swal.stopTimer);
820      toast.addEventListener("mouseleave", Swal.resumeTimer);
821    },
822  });
823
824  // Ejecutar solo si viene de poner cita
825  const Toast = Swal.mixin({
826    toast: true,
827    position: "top-end",
828    showConfirmButton: false,
829    timer: 3000,
830    timerProgressBar: true,
831    didOpen: (toast) => {
832      toast.addEventListener("mouseenter", Swal.stopTimer);
833      toast.addEventListener("mouseleave", Swal.resumeTimer);
834    },
835  });
836
837  // Ejecutar solo si viene de poner cita
838  const Toast = Swal.mixin({
839    toast: true,
840    position: "top-end",
841    showConfirmButton: false,
842    timer: 3000,
843    timerProgressBar: true,
844    didOpen: (toast) => {
845      toast.addEventListener("mouseenter", Swal.stopTimer);
846      toast.addEventListener("mouseleave", Swal.resumeTimer);
847    },
848  });
849
850  // Ejecutar solo si viene de poner cita
851  const Toast = Swal.mixin({
852    toast: true,
853    position: "top-end",
854    showConfirmButton: false,
855    timer: 3000,
856    timerProgressBar: true,
857    didOpen: (toast) => {
858      toast.addEventListener("mouseenter", Swal.stopTimer);
859      toast.addEventListener("mouseleave", Swal.resumeTimer);
860    },
861  });
862
863  // Ejecutar solo si viene de poner cita
864  const Toast = Swal.mixin({
865    toast: true,
866    position: "top-end",
867    showConfirmButton: false,
868    timer: 3000,
869    timerProgressBar: true,
870    didOpen: (toast) => {
871      toast.addEventListener("mouseenter", Swal.stopTimer);
872      toast.addEventListener("mouseleave", Swal.resumeTimer);
873    },
874  });
875
876  // Ejecutar solo si viene de poner cita
877  const Toast = Swal.mixin({
878    toast: true,
879    position: "top-end",
880    showConfirmButton: false,
881    timer: 3000,
882    timerProgressBar: true,
883    didOpen: (toast) => {
884      toast.addEventListener("mouseenter", Swal.stopTimer);
885      toast.addEventListener("mouseleave", Swal.resumeTimer);
886    },
887  });
888
889  // Ejecutar solo si viene de poner cita
890  const Toast = Swal.mixin({
891    toast: true,
892    position: "top-end",
893    showConfirmButton: false,
894    timer: 3000,
895    timerProgressBar: true,
896    didOpen: (toast) => {
897      toast.addEventListener("mouseenter", Swal.stopTimer);
898      toast.addEventListener("mouseleave", Swal.resumeTimer);
899    },
900  });
901
902  // Ejecutar solo si viene de poner cita
903  const Toast = Swal.mixin({
904    toast: true,
905    position: "top-end",
906    showConfirmButton: false,
907    timer: 3000,
908    timerProgressBar: true,
909    didOpen: (toast) => {
910      toast.addEventListener("mouseenter", Swal.stopTimer);
911      toast.addEventListener("mouseleave", Swal.resumeTimer);
912    },
913  });
914
915  // Ejecutar solo si viene de poner cita
916  const Toast = Swal.mixin({
917    toast: true,
918    position: "top-end",
919    showConfirmButton: false,
920    timer: 3000,
921    timerProgressBar: true,
922    didOpen: (toast) => {
923      toast.addEventListener("mouseenter", Swal.stopTimer);
924      toast.addEventListener("mouseleave", Swal.resumeTimer);
925    },
926  });
927
928  // Ejecutar solo si viene de poner cita
929  const Toast = Swal.mixin({
930    toast: true,
931    position: "top-end",
932    showConfirmButton: false,
933    timer: 3000,
934    timerProgressBar: true,
935    didOpen: (toast) => {
936      toast.addEventListener("mouseenter", Swal.stopTimer);
937      toast.addEventListener("mouseleave", Swal.resumeTimer);
938    },
939  });
940
941  // Ejecutar solo si viene de poner cita
942  const Toast = Swal.mixin({
943    toast: true,
944    position: "top-end",
945    showConfirmButton: false,
946    timer: 3000,
947    timerProgressBar: true,
948    didOpen: (toast) => {
949      toast.addEventListener("mouseenter", Swal.stopTimer);
950      toast.addEventListener("mouseleave", Swal.resumeTimer);
951    },
952  });
953
954  // Ejecutar solo si viene de poner cita
955  const Toast = Swal.mixin({
956    toast: true,
957    position: "top-end",
958    showConfirmButton: false,
959    timer: 3000,
960    timerProgressBar: true,
961    didOpen: (toast) => {
962      toast.addEventListener("mouseenter", Swal.stopTimer);
963      toast.addEventListener("mouseleave", Swal.resumeTimer);
964    },
965  });
966
967  // Ejecutar solo si viene de poner cita
968  const Toast = Swal.mixin({
969    toast: true,
970    position: "top-end",
971    showConfirmButton: false,
972    timer: 3000,
973    timerProgressBar: true,
974    didOpen: (toast) => {
975      toast.addEventListener("mouseenter", Swal.stopTimer);
976      toast.addEventListener("mouseleave", Swal.resumeTimer);
977    },
978  });
979
980  // Ejecutar solo si viene de poner cita
981  const Toast = Swal.mixin({
982    toast: true,
983    position: "top-end",
984    showConfirmButton: false,
985    timer: 3000,
986    timerProgressBar: true,
987    didOpen: (toast) => {
988      toast.addEventListener("mouseenter", Swal.stopTimer);
989      toast.addEventListener("mouseleave", Swal.resumeTimer);
990    },
991  });
992
993  // Ejecutar solo si viene de poner cita
994  const Toast = Swal.mixin({
995    toast: true,
996    position: "top-end",
997    showConfirmButton: false,
998    timer: 3000,
999    timerProgressBar: true,
1000    didOpen: (toast) => {
1001      toast.addEventListener("mouseenter", Swal.stopTimer);
1002      toast.addEventListener("mouseleave", Swal.resumeTimer);
1003    },
1004  });
1005
1006  // Ejecutar solo si viene de poner cita
1007  const Toast = Swal.mixin({
1008    toast: true,
1009    position: "top-end",
1010    showConfirmButton: false,
1011    timer: 3000,
1012    timerProgressBar: true,
1013    didOpen: (toast) => {
1014      toast.addEventListener("mouseenter", Swal.stopTimer);
1015      toast.addEventListener("mouseleave", Swal.resumeTimer);
1016    },
1017  });
1018
1019  // Ejecutar solo si viene de poner cita
1020  const Toast = Swal.mixin({
1021    toast: true,
1022    position: "top-end",
1023    showConfirmButton: false,
1024    timer: 3000,
1025    timerProgressBar: true,
1026    didOpen: (toast) => {
1027      toast.addEventListener("mouseenter", Swal.stopTimer);
1028      toast.addEventListener("mouseleave", Swal.resumeTimer);
1029    },
1030  });
1031
1032  // Ejecutar solo si viene de poner cita
1033  const Toast = Swal.mixin({
1034    toast: true,
1035    position: "top-end",
1036    showConfirmButton: false,
1037    timer: 3000,
1038    timerProgressBar: true,
1039    didOpen: (toast) => {
1040      toast.addEventListener("mouseenter", Swal.stopTimer);
1041      toast.addEventListener("mouseleave", Swal.resumeTimer);
1042    },
1043  });
1044
1045  // Ejecutar solo si viene de poner cita
1046  const Toast = Swal.mixin({
1047    toast: true,
1048    position: "top-end",
1049    showConfirmButton: false,
1050    timer: 3000,
1051    timerProgressBar: true,
1052    didOpen: (toast) => {
1053      toast.addEventListener("mouseenter", Swal.stopTimer);
1054      toast.addEventListener("mouseleave", Swal.resumeTimer);
1055    },
1056  });
1057
1058  // Ejecutar solo si viene de poner cita
1059  const Toast = Swal.mixin({
1060    toast: true,
1061    position: "top-end",
1062    showConfirmButton: false,
1063    timer: 3000,
1064    timerProgressBar: true,
1065    didOpen: (toast) => {
1066      toast.addEventListener("mouseenter", Swal.stopTimer);
1067      toast.addEventListener("mouseleave", Swal.resumeTimer);
1068    },
1069  });
1070
1071  // Ejecutar solo si viene de poner cita
1072  const Toast = Swal.mixin({
1073    toast: true,
1074    position: "top-end",
1075    showConfirmButton: false,
1076    timer: 3000,
1077    timerProgressBar: true,
1078    didOpen: (toast) => {
1079      toast.addEventListener("mouseenter", Swal.stopTimer);
1080      toast.addEventListener("mouseleave", Swal.resumeTimer);
1081    },
1082  });
1083
1084  // Ejecutar solo si viene de poner cita
1085  const Toast = Swal.mixin({
1086    toast: true,
1087    position: "top-end",
1088    showConfirmButton: false,
1089    timer: 3000,
1090    timerProgressBar: true,
1091    didOpen: (toast) => {
1092      toast.addEventListener("mouseenter", Swal.stopTimer);
1093      toast.addEventListener("mouseleave", Swal.resumeTimer);
1094    },
1095  });
1096
1097  // Ejecutar solo si viene de poner cita
1098  const Toast = Swal.mixin({
1099    toast: true,
1100    position: "top-end",
1101    showConfirmButton: false,
1102    timer: 3000,
1103    timerProgressBar: true,
1104    didOpen: (toast) => {
1105      toast.addEventListener("mouseenter", Swal.stopTimer);
1106      toast.addEventListener("mouseleave", Swal.resumeTimer);
1107    },
1108  });
1109
1110  // Ejecutar solo si viene de poner cita
1111  const Toast = Swal.mixin({
1112    toast: true,
1113    position: "top-end",
1114    showConfirmButton: false,
1115    timer: 3000,
1116    timerProgressBar: true,
1117    didOpen: (toast) => {
1118      toast.addEventListener("mouseenter", Swal.stopTimer);
1119      toast.addEventListener("mouseleave", Swal.resumeTimer);
1120    },
1121  });
1122
1123  // Ejecutar solo si viene de poner cita
1124  const Toast = Swal.mixin({
1125    toast: true,
1126    position: "top-end",
1127    showConfirmButton: false,
1128    timer: 3000,
1129    timerProgressBar: true,
1130    didOpen: (toast) => {
1131      toast.addEventListener("mouseenter", Swal.stopTimer);
1132      toast.addEventListener("mouseleave", Swal.resumeTimer);
1133    },
1134  });
1135
1136  // Ejecutar solo si viene de poner cita
1137  const Toast = Swal.mixin({
1138    toast: true,
1139    position: "top-end",
1140    showConfirmButton: false,
1141    timer: 3000,
1142    timerProgressBar: true,
1143    didOpen: (toast) => {
1144      toast.addEventListener("mouseenter", Swal.stopTimer);
1145      toast.addEventListener("mouseleave", Swal.resumeTimer);
1146    },
1147  });
1148
1149  // Ejecutar solo si viene de poner cita
1150  const Toast = Swal.mixin({
1151    toast: true,
1152    position: "top-end",
1153    showConfirmButton: false,
1154    timer: 3000,
1155    timerProgressBar: true,
1156    didOpen: (toast) => {
1157      toast.addEventListener("mouseenter", Swal.stopTimer);
1158      toast.addEventListener("mouseleave", Swal.resumeTimer);
1159    },
1160  });
1161
1162  // Ejecutar solo si viene de poner cita
1163  const Toast = Swal.mixin({
1164    toast: true,
1165    position: "top-end",
1166    showConfirmButton: false,
1167    timer: 3000,
1168    timerProgressBar: true,
1169    didOpen: (toast) => {
1170      toast.addEventListener("mouseenter", Swal.stopTimer);
1171      toast.addEventListener("mouseleave", Swal.resumeTimer);
1172    },
1173  });
1174
1175  // Ejecutar solo si viene de poner cita
1176  const Toast = Swal.mixin({
1177    toast: true,
1178    position: "top-end",
1179    showConfirmButton: false,
1180    timer: 3000,
1181    timerProgressBar: true,
1182    didOpen: (toast) => {
1183      toast.addEventListener("mouseenter", Swal.stopTimer);
1184      toast.addEventListener("mouseleave", Swal.resumeTimer);
1185    },
1186  });
1187
1188  // Ejecutar solo si viene de poner cita
1189  const Toast = Swal.mixin({
1190    toast: true,
1191    position: "top-end",
1192    showConfirmButton: false,
1193    timer: 3000,
1194    timerProgressBar: true,
1195    didOpen: (toast) => {
1196      toast.addEventListener("mouseenter", Swal.stopTimer);
1197      toast.addEventListener("mouseleave", Swal.resumeTimer);
1198    },
1199  });
1200
1201  // Ejecutar solo si viene de poner cita
1202  const Toast = Swal.mixin({
1203    toast: true,
1204    position: "top-end",
1205    showConfirmButton: false,
1206    timer: 3000,
1207    timerProgressBar: true,
1208    didOpen: (toast) => {
1209      toast.addEventListener("mouseenter", Swal.stopTimer);
1210      toast.addEventListener("mouseleave", Swal.resumeTimer);
1211    },
1212  });
1213
1214  // Ejecutar solo si viene de poner cita
1215  const Toast = Swal.mixin({
1216    toast: true,
1217    position: "top-end",
1218    showConfirmButton: false,
1219    timer: 3000,
1220    timerProgressBar: true,
1221    didOpen: (toast) => {
1222      toast.addEventListener("mouseenter", Swal.stopTimer);
1223      toast.addEventListener("mouseleave", Swal.resumeTimer);
1224    },
1225  });
1226
1227  // Ejecutar solo si viene de poner cita
1228  const Toast = Swal.mixin({
1229    toast: true,
1230    position: "top-end",
1231    showConfirmButton: false,
1232    timer: 3000,
1233    timerProgressBar: true,
1234    didOpen: (toast) => {
1235      toast.addEventListener("mouseenter", Swal.stopTimer);
1236      toast.addEventListener("mouseleave", Swal.resumeTimer);
1237    },
1238  });
1239
1240  // Ejecutar solo si viene de poner cita
1241  const Toast = Swal.mixin({
1242    toast: true,
1243    position: "top-end",
1244    showConfirmButton: false,
1245    timer: 3000,
1246    timerProgressBar: true,
1247    didOpen: (toast) => {
1248      toast.addEventListener("mouseenter", Swal.stopTimer);
1249      toast.addEventListener("mouseleave", Swal.resumeTimer);
1250    },
1251  });
1252
1253  // Ejecutar solo si viene de poner cita
1254  const Toast = Swal.mixin({
1255    toast: true,
1256    position: "top-end",
1257    showConfirmButton: false,
1258    timer: 3000,
1259    timerProgressBar: true,
1260    didOpen: (toast) => {
1261      toast.addEventListener("mouseenter", Swal.stopTimer);
1262      toast.addEventListener("mouseleave", Swal.resumeTimer);
1263    },
1264  });
1265
1266  // Ejecutar solo si viene de poner cita
1267  const Toast = Swal.mixin({
1268    toast: true,
1269    position: "top-end",
1270    showConfirmButton: false,
1271    timer: 3000,
1272    timerProgressBar: true,
1273    didOpen: (toast) => {
1274      toast.addEventListener("mouseenter", Swal.stopTimer);
1275      toast.addEventListener("mouseleave", Swal.resumeTimer);
1276    },
1277  });
1278
1279  // Ejecutar solo si viene de poner cita
1280  const Toast = Swal.mixin({
1281    toast: true,
1282    position: "top-end",
1283    showConfirmButton: false,
1284    timer: 3000,
1285    timerProgressBar: true,
1286    didOpen: (toast) => {
1287      toast.addEventListener("mouseenter", Swal.stopTimer);
1288      toast.addEventListener("mouseleave", Swal.resumeTimer);
1289    },
1290  });
1291
1292  // Ejecutar solo si viene de poner cita
1293  const Toast = Swal.mixin({
1294    toast: true,
1295    position: "top-end",
1296    showConfirmButton: false,
1297    timer: 3000,
1298    timerProgressBar: true,
1299    didOpen: (toast) => {
1300      toast.addEventListener("mouseenter", Swal.stopTimer);
1301      toast.addEventListener("mouseleave", Swal.resumeTimer);
1302    },
1303  });
1304
1305  // Ejecutar solo si viene de poner cita
1306  const Toast = Swal.mixin({
1307    toast: true,
1308    position: "top-end",
1309    showConfirmButton: false,
1310    timer: 3000,
1311    timerProgressBar: true,
1312    didOpen: (toast) => {
1313      toast.addEventListener("mouseenter", Swal.stopTimer);
1314      toast.addEventListener("mouseleave", Swal.resumeTimer);
1315    },
1316  });
1317
1318  // Ejecutar solo si viene de poner cita
1319  const Toast = Swal.mixin({
1320    toast: true,
1321    position: "top-end",
1322    showConfirmButton: false,
1323    timer: 3000,
1324    timerProgressBar: true,
1325    didOpen: (toast) => {
1326      toast.addEventListener("mouseenter", Swal.stopTimer);
1327      toast.addEventListener("mouseleave", Swal.resumeTimer);
1328    },
1329  });
1330
1331  // Ejecutar solo si viene de poner cita
1332  const Toast = Swal.mixin({
1333    toast: true,
1334    position: "top-end",
1335    showConfirmButton: false,
1336    timer: 3000,
1337    timerProgressBar: true,
1338    didOpen: (toast) => {
1339      toast.addEventListener("mouseenter", Swal.stopTimer);
1340      toast.addEventListener("mouseleave", Swal.resumeTimer);
1341    },
1342  });
1343
1344  //
```

10 Modelo, Vista, Controlador (MVC)

El Modelo-Vista-Controlador es una arquitectura software.

Se encarga de recibir en el controlador la petición enviada por usuario, una vez recibida la petición el controlador analizara que tiene que hacer y pedirá tanto los datos como las funciones que necesita de los modelo para así poder procesar la información y pasársela a la vista que esta será recibida por el cliente, el cual podrá visualizarlo en su buscador



El controlador se encuentra en la ruta de directorio: **es.tfg.hospital.controller**

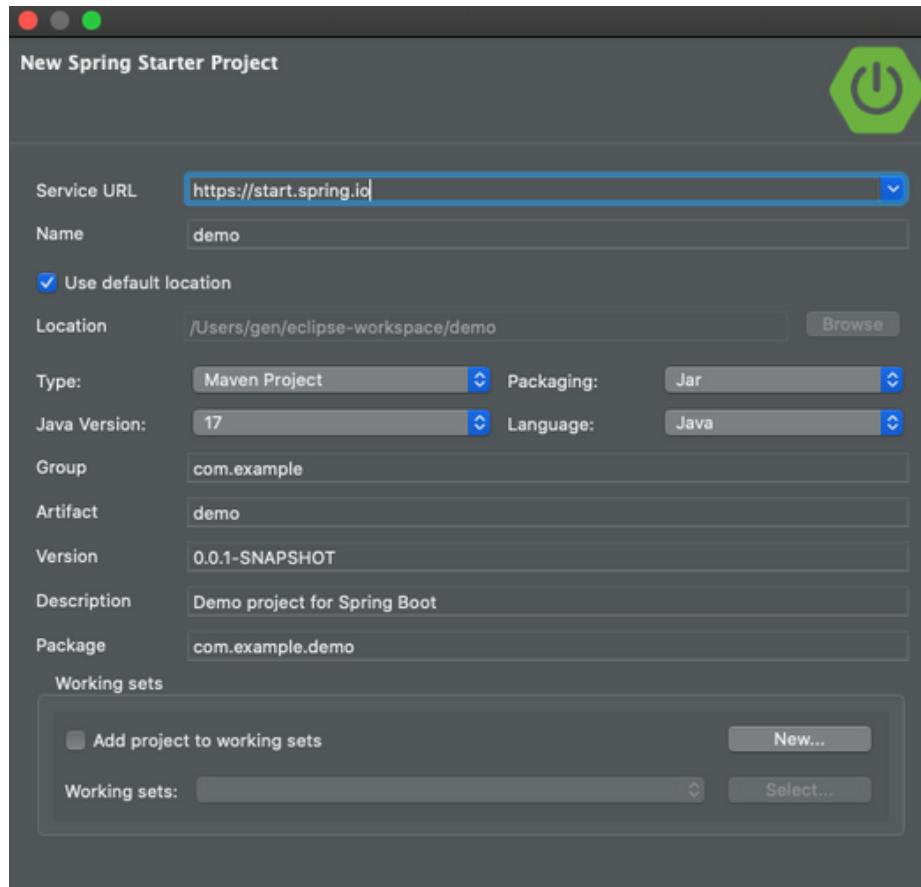
El modelo se compone de beans , dao y el repository, estos directorios se encuentran en la ruta: **es.tfg.hospital.modelo.beans**, **es.tfg.hospital.modelo.dao**, **es.tfg.hospital.modelo.repository**.

Las vistas se componen de archivos .jsp y se encuentran en el directorio:
src.main.webapp.WEB-INF.vistas

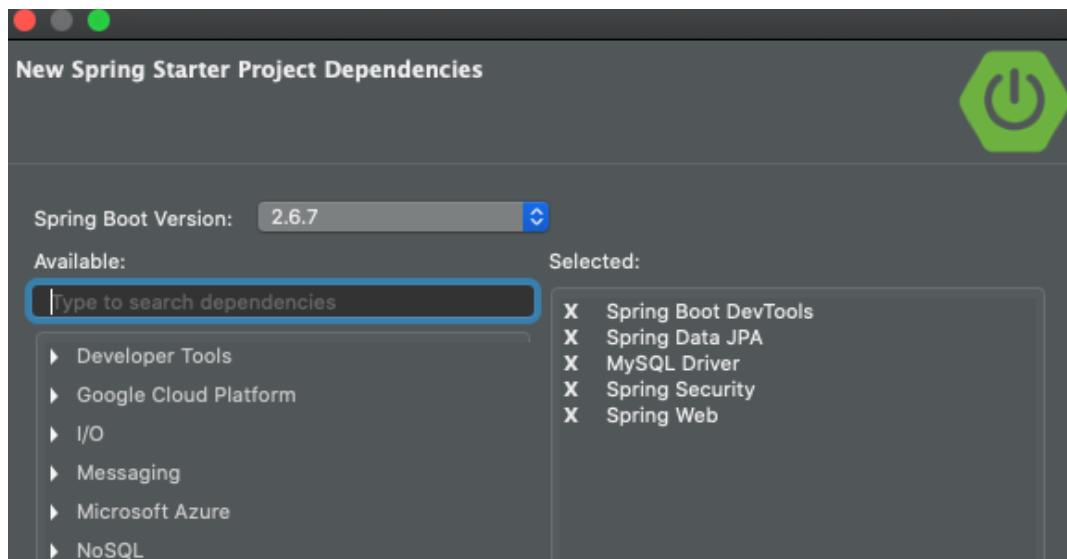
Configuración del proyecto

Para generar el proyecto seguiremos los siguientes pasos:

- File -> New Spring starter project



- Elegimos las dependencias de Spring que vamos a utilizar



Configuramos el archivo application.properties (**src/main/resources**)

```
1 server.port=8086
2 spring.mvc.view.prefix=/WEB-INF/vistas/
3 spring.mvc.view.suffix=.jsp
4
5 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
6 spring.datasource.url=jdbc:mysql://localhost:3306/hospital_bbdd?serverTimezone=UTC
7 spring.datasource.username=uemedico
8 spring.datasource.password=uemedico
9
10 spring.jpa.generate-ddl=false
11 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
12 spring.jpa.show-sql=true
13 spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl
14
```

Aquí configuramos el puerto , la conexión a la base de datos y la ruta donde se encuentran nuestras vistas.

Botón derecho sobre el propio proyecto:

- Properties -> Project Facets->Convert faceted from ->JPA
- Further configuration available

JPA

- Plataforma Generic 2.2
- Conexión -> Añadimos conexión, especificamos my sql
- Metemos el driver mysql Conektor J 8.0 (jar)
- Propiedades:
 - jdbc:mysql://localhost:3306/base_datos?Timezone=UTC
 - base_datos
 - com.mysql.cj.jdbc.Driver
 - Contraseña
 - UserID
 - Test conexión

Botón derecho proyecto->JPA Tools -> new JPA Entities from Tables

- Seleccionar conexión
- Seleccionamos las tablas
- Elegir las asociaciones
- Si es una referencia a una colección de objetos, generar como cascade persist

En el archivo pom.xml también añadiremos otras dependencias necesarias, como por ejemplo el taglib.

```
<dependency>
  <groupId>org.springframework.security</groupId>
  <artifactId>spring-security-taglibs</artifactId>
  <version>5.4.5</version>
</dependency>
```

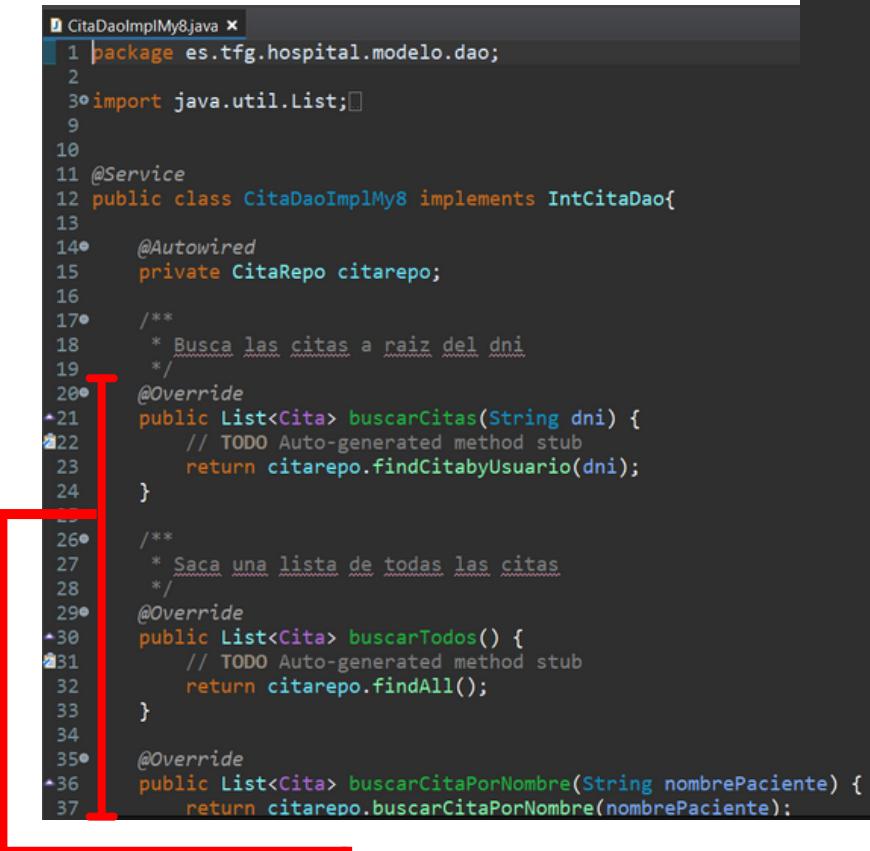
En el directorio dentro de la ruta **src/main/resources static** pondremos nuestros archivos css y js e imágenes y contenido estático.

11 Modelo DAO

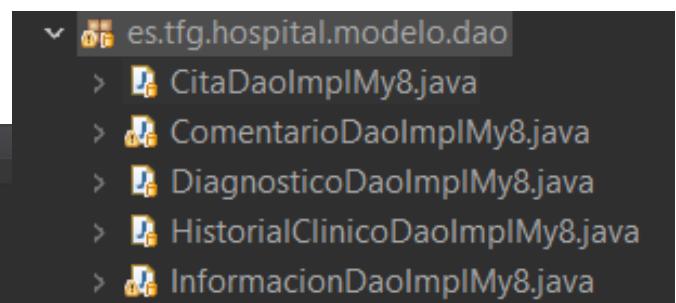
El Modelo DAO (Data Access Object) es lo que nos permite acceder a los datos utilizando métodos definidos por el propio DAO

En los archivos Dao se tiene que poner @Service arriba de la clase esto se usa para construir una clase de Servicio que habitualmente se conecta a varios repositorios.

Ejemplos casos Modelo DAO en el proyecto:

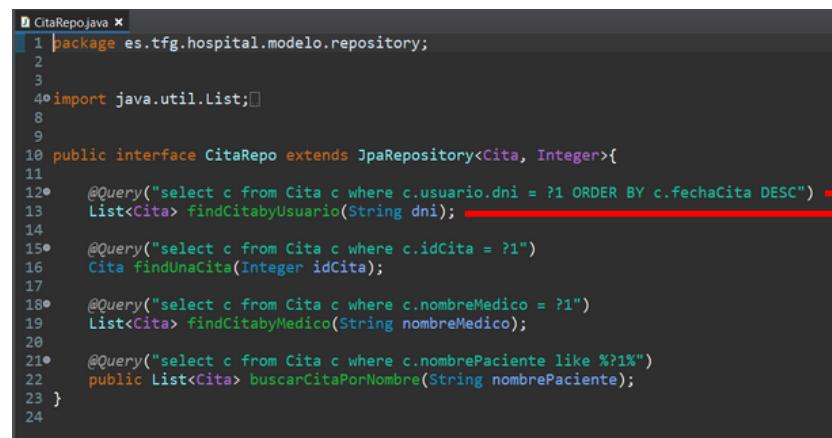


```
1 package es.tfg.hospital.modelo.dao;
2
3 import java.util.List;
4
5
6 @Service
7 public class CitaDaoImplMy8 implements IntCitaDao{
8
9     @Autowired
10    private CitaRepo citarepo;
11
12
13    /**
14     * Busca las citas a raiz del dni
15     */
16    @Override
17    public List<Cita> buscarCitas(String dni) {
18        // TODO Auto-generated method stub
19        return citarepo.findCitabyUsuario(dni);
20    }
21
22
23    /**
24     * Saca una lista de todas las citas
25     */
26    @Override
27    public List<Cita> buscarTodos() {
28        // TODO Auto-generated method stub
29        return citarepo.findAll();
30    }
31
32
33    @Override
34    public List<Cita> buscarCitaPorNombre(String nombrePaciente) {
35        return citarepo.buscarCitaPorNombre(nombrePaciente);
36    }
37}
```



En esta parte del código lo que hacemos son diferentes métodos para poder obtener los datos que queremos tanto la lista completa como solo uno.

Esto se hace llamando a los métodos de CitaRepo que es un archivo que manda las sentencias SQL a la base de dato con el fin de obtenerlos



```
1 package es.tfg.hospital.modelo.repository;
2
3
4 import java.util.List;
5
6
7 public interface CitaRepo extends JpaRepository<Cita, Integer>{
8
9
10    @Query("select c from Cita c where c.usuario.dni = ?1 ORDER BY c.fechaCita DESC")
11    List<Cita> findCitabyUsuario(String dni);
12
13
14    @Query("select c from Cita c where c.idCita = ?1")
15    Cita findUnaCita(Integer idCita);
16
17
18    @Query("select c from Cita c where c.nombreMedico = ?1")
19    List<Cita> findCitabyMedico(String nombreMedico);
20
21
22    @Query("select c from Cita c where c.nombrePaciente like %?1%")
23    public List<Cita> buscarCitaPorNombre(String nombrePaciente);
24}
```

Sentencia SQL

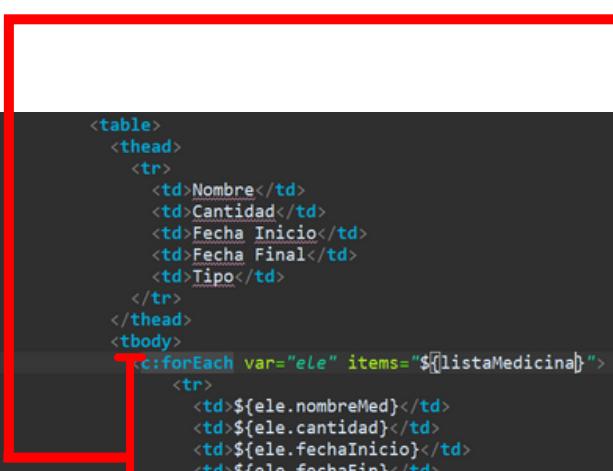
Método

12 Vista

La Vista es el archivo donde se recibe la información proveniente del Controlador para que este archivo se vea bien se utiliza

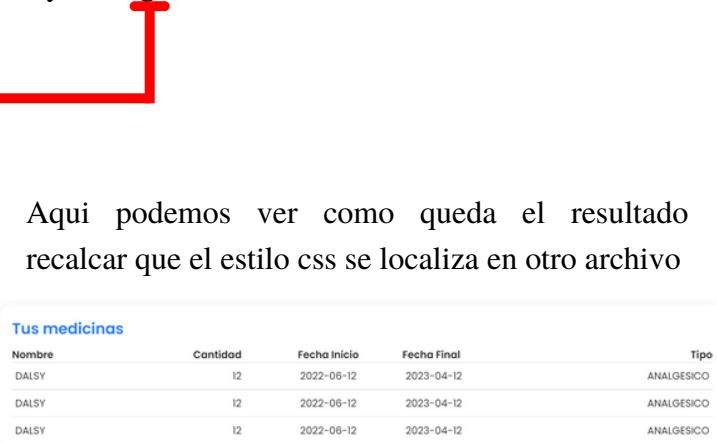
- Html
- Css
- JavaScript o para el pdf
- Taglib (son unas etiquetas que permiten realizar algunas acciones como introducir datos de una lista)

En la foto de la izquierda se hace la union del html y del taglib



```
174<table>
175  <thead>
176    <tr>
177      <td>Nombre</td>
178      <td>Cantidad</td>
179      <td>Fecha Inicio</td>
180      <td>Fecha Final</td>
181      <td>Tipo</td>
182    </tr>
183  </thead>
184  <tbody>
185    <c:forEach var="ele" items="${listaMedicina}">
186      <tr>
187        <td>${ele.nombreMed}</td>
188        <td>${ele.cantidad}</td>
189        <td>${ele.fechaInicio}</td>
190        <td>${ele.fechaFin}</td>
191        <td>${ele.tipo.descripcion}</td>
192      </tr>
193    </c:forEach>
194  </tbody>
195</table>
```

Aqui podemos ver como queda el resultado
recalcar que el estilo css se localiza en otro archivo



Tus medicinas				
Nombre	Cantidad	Fecha Inicio	Fecha Final	Tipo
DALSY	I2	2022-06-12	2023-04-12	ANALGESICO
DALSY	I2	2022-06-12	2023-04-12	ANALGESICO
DALSY	I2	2022-06-12	2023-04-12	ANALGESICO

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib prefix="sec" uri="http://www.springframework.org/security/tags" %>
```

13 Controllador

El Controlador es el cerebro de todo el programa, este se encarga de unir las peticiones del usuario para así poder tramitar la información de tal manera, que si el usuario quiere acceder a una ventana el Controllador

1. Recibe la información de a que pestaña quiere acceder



2. Realiza la petición de los datos necesarios, los pasa en una etiqueta para que el jsp los inserte y devuelve el archivo .jsp

```
526  
527* @GetMapping("/medicina")  
528 public String mostrarMedicinas(Model model, HttpSession misesion) {  
529     List<Medicina> listaMedicina= mdao.buscarMedicinasUsuario((String) misesion.getAttribute("dni"));  
530     model.addAttribute("listaMedicina",listaMedicina);  
531     return "medicinas";  
532 }
```

Petición de datos

Nombre del archivo .jsp

Etiqueta de los datos

3. Así es como lo visualizaríamos

Tus medicinas				
Nombre	Cantidad	Fecha Inicio	Fecha Final	Tipo
DALSY	12	2022-06-12	2023-04-12	ANALGESICO
DALSY	12	2022-06-12	2023-04-12	ANALGESICO
DALSY	12	2022-06-12	2023-04-12	ANALGESICO

14 Apache Tomcat

Apache Tomcat es un contenedor de servlets que se puede usar para compilar y ejecutar aplicaciones web realizadas en Java. Implementa y da soporte tanto a servlets como a páginas JSP (Java Server Pages) o Java Sockets. Además, Tomcat es compatible con otras tecnologías como Java Expression Language y Java WebSocket, del ecosistema Java.

Estructura de directorios:

- **bin** - arranque, cierre, y otros scripts y ejecutables.
- **common** - clases comunes.
- **conf** - ficheros XML y los correspondientes DTD
- **logs** - logs de Catalina y de las aplicaciones
- **server** - clases utilizadas solamente por Catalina
- **shared** - clases compartidas por las aplicaciones web.
- **webapps** - directorio que contiene las aplicaciones web.
- **work** - almacenamiento temporal de ficheros y directorio

Despliegue de la aplicación

En caso de tener que desplegar la aplicación solo tendremos que seguir los siguientes pasos

Para realizar el despliegue de la aplicación se podía utilizar Heroku que es una plataforma como servicio de computación en la Nube que soporta distintos lenguajes de programación.

Para ello necesitaremos:

- Un proyecto Spring Boot existente con base de datos MySQL
- Una cuenta verificada de Heroku
- Heroku CLI instalado
- Git instalado
- Servidor comunitario MySQL con cliente de línea de comandos MySQL

1- Implementar un proyecto Spring Boot en Heroku

- Abra un nuevo símbolo del sistema en Windows o terminal en Mac o Linux.
- Cambie el directorio actual al directorio raíz de su proyecto
- Cree un nuevo repositorio local de Git: **git init**

```
@)git init
Initialized empty Git repository in G:/Heroku/SpringBootInMemoryFormAuth/.git/
```

- Agregue los archivos del proyecto al índice del repositorio local: **git add**.
- Confirme los archivos en el repositorio local de Git: **git commit -m "first commit"**

```
@)git commit -m "commit to deploy"
[master (root-commit) 906adc1] commit to deploy
 20 files changed, 939 insertions(+)
```

- Iniciar sesión en Heroku: **heroku login**

```
@)heroku login
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/4cce92c4-
2gDbQAAAA4xMTYuMTEwLjE3LjIwMG4GAGjfdC98AWIAAVGA.BU38Zb0pCZxgSgFp_cyCI_No
Logging in... done
Logged in as [REDACTED] your email address
```

- Crea una nueva aplicación de Heroku: **heroku create nombre-app**

```
@)heroku create my-spring-app-1
Creating my-spring-app-1... done
https://my-spring-app-1.herokuapp.com/ | https://git.heroku.com/my-spring-app-1.git
```

- Implemente el proyecto a través de Git: **git push heroku master**

```
remote:      Default types for buildpack -> web
remote:
remote: -----> Compressing...
remote:          Done: 69.7M
remote: -----> Launching...
remote:          Released v3
remote:          https://my-spring-app-1.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/my-spring-app-1.git
 * [new branch]      master -> master
```

2- Instalar el complemento ClearDB MySQL

Utilizaremos estos comandos para transladar nuestra base del mySQLWorkbench

```
heroku addons:create cleardb:ignite -a nombre-app
```

Una vez instalado, este complemento crea una nueva variable de configuración para su aplicación, llamada **CLEARDB_DATABASE_URL** , que almacena información de conexión a la base de datos. Es algo como esto:

```
CLEARDB_DATABASE_URL: mysql://username:password@hostname/schema_name
```

3- Exporte la base de datos MySQL local a un archivo SQL

```
mysqldump -u nombre_de_usuario -p nombre_esquema > mysql_dump_file.sql
```

Este comando exporta el esquema de base de datos dado a un archivo .sql. Reemplace nombre de usuario, schema_name y mysql_dump_file por valores reales. Tenga en cuenta que debe ingresar la contraseña de MySQL para ejecutar este comando.

4- Importar archivo SQL al servidor MySQL remoto

A continuación, le mostraré cómo conectarse al servidor MySQL remoto para importar el archivo .sql. Escriba este comando para ver las variables de configuración de su aplicación:

```
heroku config -a nombre_de_tu_aplicación
```

Extraiga el nombre de usuario, la contraseña, el nombre de host y el nombre del esquema de la variable de configuración CLEARDB_DATABASE_URL . Y escriba el siguiente comando para conectarse al servidor MySQL remoto con el programa cliente MySQL Command Line:

```
mysql -u nombre_de_usuario -p -h nombre_de_host
```

Es necesario introducir la contraseña para iniciar sesión. Luego, en el indicador del cliente de línea de comandos de MySQL, escriba este comando para conectarse al esquema de la base de datos creado por el complemento ClearDB:

```
connect schema_name ;
```

A continuación, escriba el siguiente comando para obtener el juego de caracteres y la intercalación de la base de datos actual:

```
select @@character_set_database, @@collation_database
```

Luego reemplace todos los valores del conjunto de caracteres por utf8; cotejar/cotejar por utf8_general_ci.

Guarde el archivo .sql. Luego escriba exit para salir del programa cliente MySQL Command Line. Ahora escriba este comando para importar el archivo .sql al servidor MySQL remoto:

```
mysql -u username -p -h hostname < mysql_dump_file.sql
```

Debe ingresar la contraseña cuando se le solicite. Y espere un momento hasta que el comando termine en silencio. Luego, puede conectarse nuevamente al servidor MySQL remoto y escribir los siguientes comandos para verificar que los datos se importaron correctamente:

```
connect schema_name;  
show tables;  
desc table table_name;  
select * from table_name;
```

5- Acceda a la base de datos MySQL en la aplicación Spring Boot

Tenga en cuenta que si su aplicación implementada es una aplicación Spring Boot, el complemento ClearDB MySQL también agregará las siguientes variables de entorno:

SPRING_DATASOURCE_URL
SPRING_DATASOURCE_NOMBRE DE USUARIO
SPRING_DATASOURCE_PASSWORD

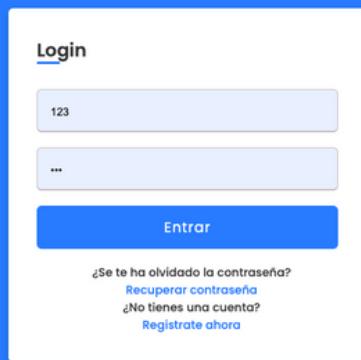
6 Conclusiones

Como conclusión final en el proyecto se observa que el uso de la aplicación web es más que intuitiva y accesible que las existentes citadas anteriormente mejorando su funcionalidad y la experiencia de usuario.

En cuanto a los resultados se han creado los métodos necesarios para cumplir las funciones que ha de tener la aplicación, también añadimos nuevas mejoras sobre la idea original como puede ser la exportación a pdf, el sistema de chat entre usuario o la capacidad de ver si los usuarios están conectados o no.

En cuanto a las posibles mejoras del proyecto podrían ser tales como una optimización del chat para que recargue la conversación en el chat o incluir y mejorar querys de búsqueda de citas y listas de usuario o médico.

Visuales



Hospital

Dashboard Mensajes Ayuda Opciones Contraseña Sign Out

Medicinas Ver tus medicinas Historial Tu historial clínico Consultas Consultas realizadas 2 Médicos disponibles

Noticias:

- Salud urbana: el barrio determina tu bienestar o tus enfermedades**
Manuel Franco
elpais.com
- Instrumentan la "Historia de Salud Integrada" en centros de salud y el Hospital de Gobernador Castro**
Noticias San Pedro <noreply@blogger.com>
blogger.com
- Una nueva plataforma protegerá la salud de los migrantes en las Américas — Listín Semanal**
Listín Semanal
un.org
- Terrel Guerrero Karina.pdf**
KarinaFiorelaTERRELG@slideshare.net(KarinaFiorelaTERRELG)
slideshare.net
- Anatomía.pptx**
yrianamartinez@slideshare.net(yrianamartinez)
slideshare.net
- Nace EL PAÍS Salud y Bienestar**
El País
elpais.com

Pedir cita

Últimas citas

Fecha	Dirección	Estado	Información
2022-06-11	Dire 1	Cancelado	Ver más
2022-06-20	Direccion1	Pendiente	Ver más

Conectados

- jaume jaume
- maria

Hospital

Dashboard Pacientes Mensajes Ayuda Opciones Contraseña Sign Out

Todas las citas

Nombre Paciente	Fecha	Dirección	Estado	Información
rodrigo	2022-04-28	Direccion rontonda 2	Realizado	Ver mas
rodrigo	2022-06-12	Direccion1	Cancelado	Ver mas
jaume	2022-04-23	Direccion rontonda 1	Pendiente	Ver mas
l23	2022-05-12	Dire 1	Realizado	Ver mas
l23	2022-06-11	Dire 1	Cancelado	Ver mas
l23	2022-04-12	Dire 2	Realizado	Ver mas
l23	2022-05-10	calle 1º 3d	Cancelado	Ver mas
nuevo	2022-05-20	Direccion2	Realizado	Ver mas
nuevo2	2022-05-28	Direccion1	Realizado	Ver mas
nuevo2	2022-05-28	Direccion1	Cancelado	Ver mas
l23	2022-05-27	Direccion1	Pendiente	Ver mas
l23	2022-05-21	Direccion1	Pendiente	Ver mas
l23	2022-05-21	Direccion1	Pendiente	Ver mas
l23	2022-05-28	Direccion1	Pendiente	Ver mas

The screenshot shows a mobile application interface for a hospital. The top navigation bar includes a logo, the word "Hospital", a search bar with placeholder text "Busca tu cita aquí", and a user profile icon.

The left sidebar contains the following menu items:

- Dashboard
- Mensajes
- Ayuda
- Opciones
- Contraseña
- Sign Out

The main content area is titled "Pide tu cita" (Request an appointment). It features fields for:

- Fecha: dd/mm/aaaa (with a calendar icon)
- Centro: Dirección1 (with a dropdown arrow)
- Médico: medi123 (with a dropdown arrow)
- Hora: 8:30 (with a dropdown arrow)
- Motivo: (empty input field)

A blue button labeled "Realizar cita" (Book appointment) is located at the bottom right of the form.

The screenshot shows a mobile application interface for a hospital, similar to the one above, but with a different main content area.

The top navigation bar includes a logo, the word "Hospital", a search bar with placeholder text "Busca tu cita aquí", and a user profile icon.

The left sidebar contains the following menu items:

- Dashboard
- Mensajes
- Ayuda
- Opciones
- Contraseña
- Sign Out

The main content area is titled "Chats" (Chats). It displays a list of chat participants with their names and profile icons:

- jaume (jaume)
- maria (maria)
- pepe (pepe)
- rodrigo (sendino)
- nuevo (nuevo)

Hospital

Dashboard Mensajes Ayuda Opciones Contraseña Sign Out

Busca tu cita aquí

Datos usuario

DNI
123

Email
123@123.123

Nombre
123

Apellido
123

Domicilio
123

Teléfono
123

Imagen URL
<https://empresas.blogthinkbig.com/wp-content/uploads/2019/11/Imagen3-245003649.jpg>

Actualizar

Datos físicos

Peso
23

Altura
45

Edad
36

Sexo

Hospital

Dashboard Mensajes Ayuda Opciones Contraseña Sign Out

Busca tu cita aquí

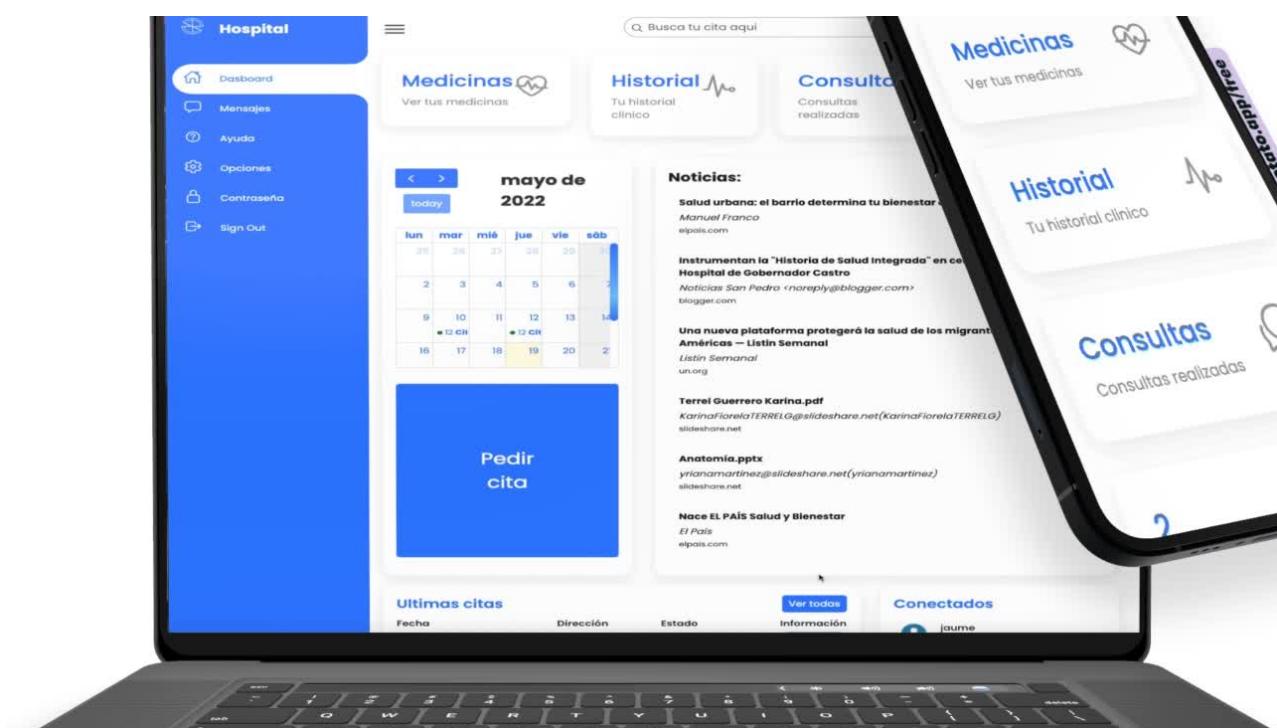
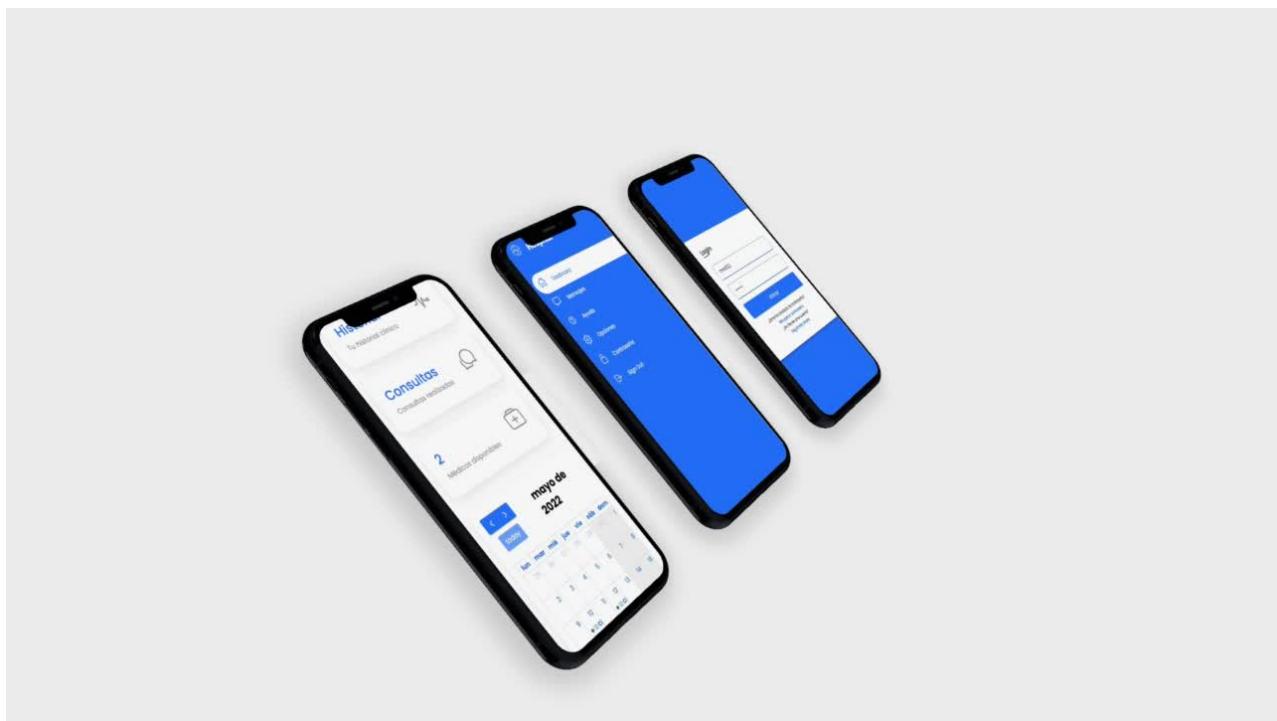
Cambiar contraseña

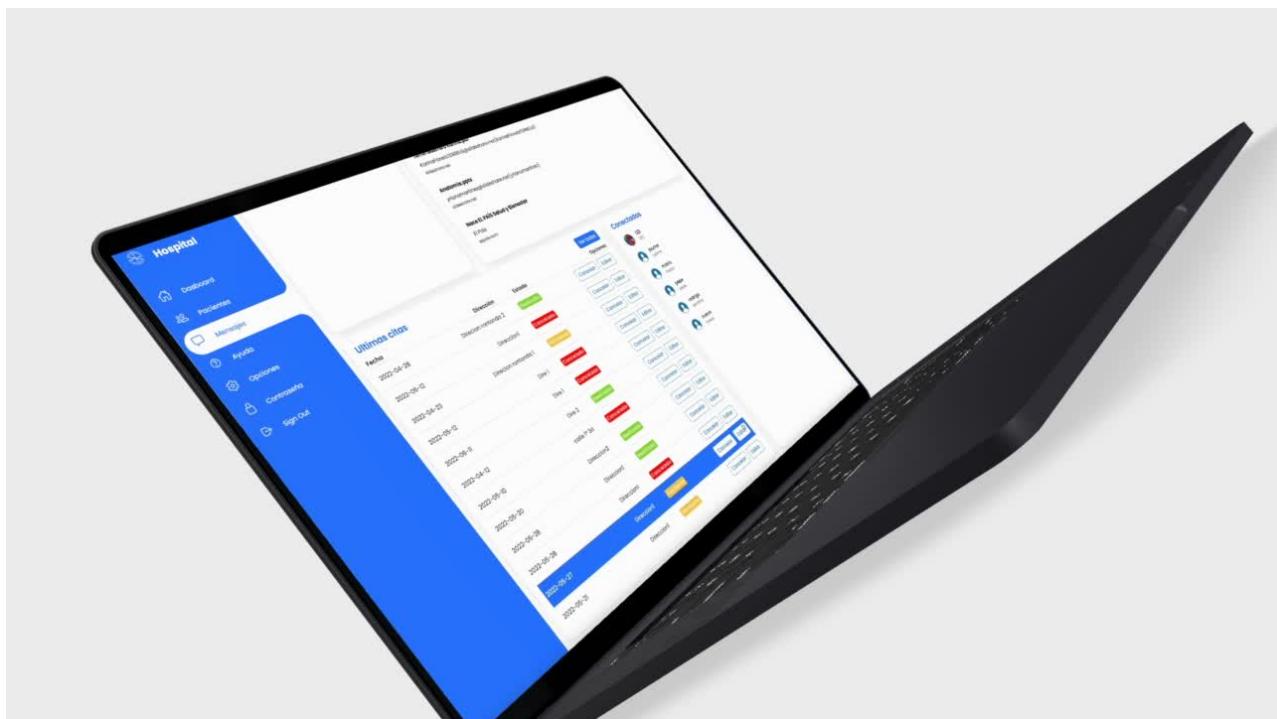
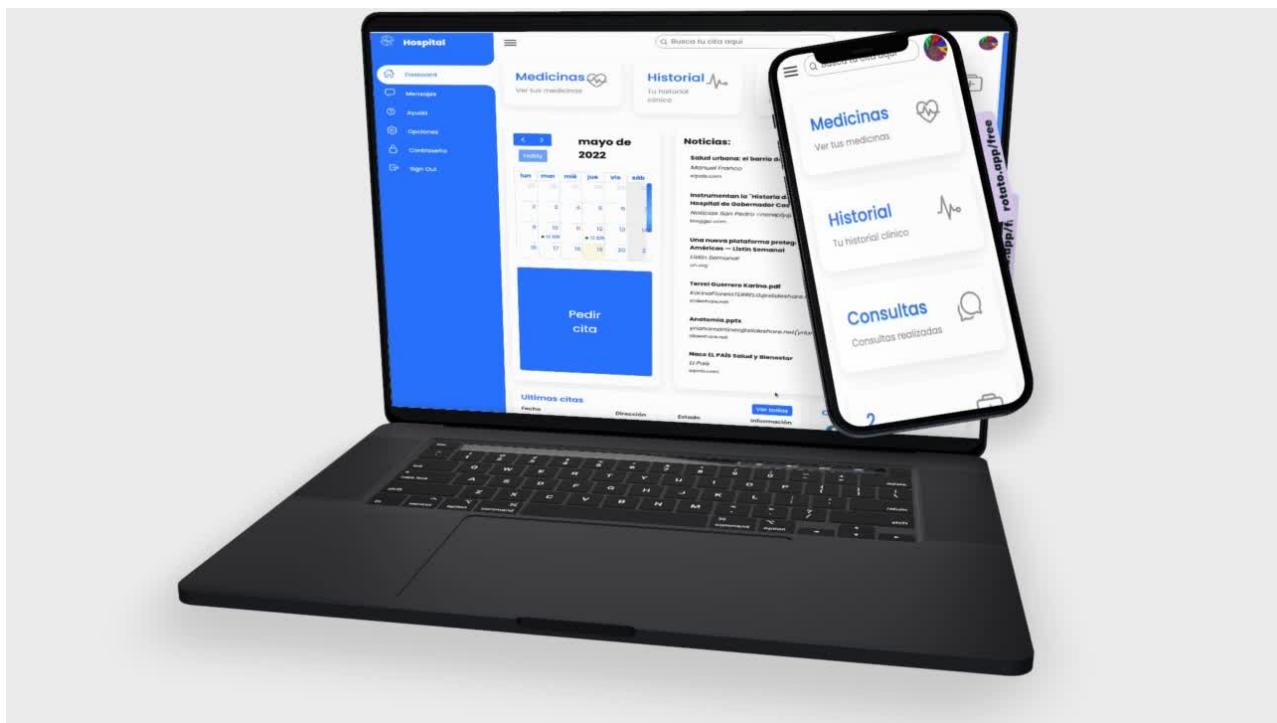
Contraseña actual

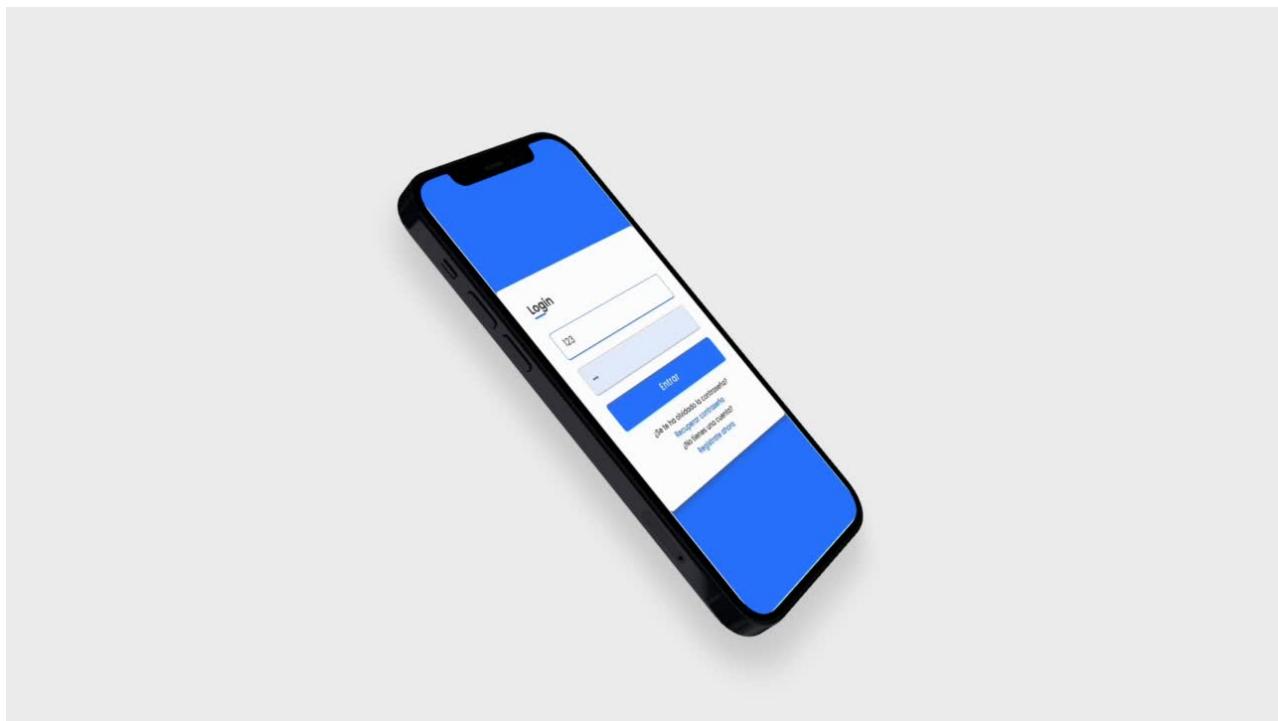
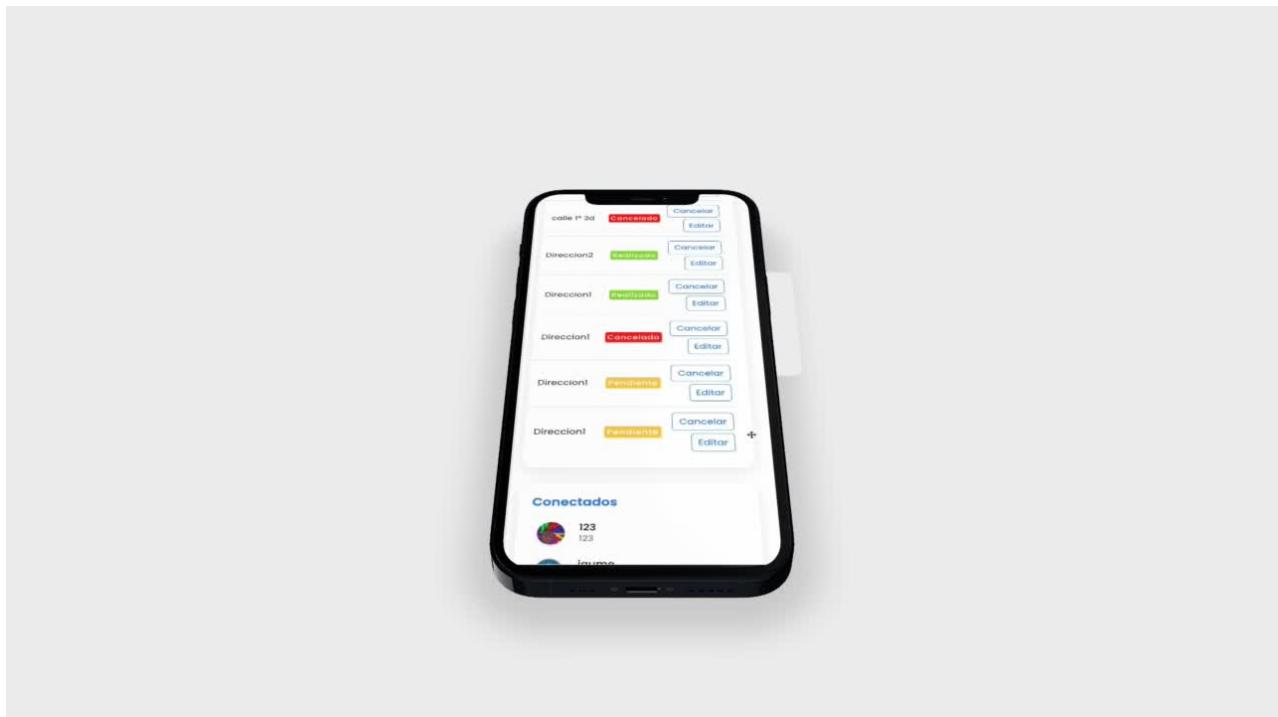
Nueva contraseña

Respte la nueva contraseña

Cambiar Contraseña







7 Agradecimientos

En estos 2 años nos hemos esforzado todos para conseguir una formación de calidad pese a los cambios y la incertidumbre.

Queríamos agradecer por el apoyo a las siguientes personas :

A Ascensión Blázquez y Daniel Díaz.

A los profesores:

- Raúl Salgado Vilas
- Félix de Pablo
- María de Gracia Carrión
- Alfredo Cordero
- Tomás Escudero Delgado

y a nuestra familia y amigos.

8 Anexos

Para testear el proyecto navegamos a el repositorio:

https://github.com/RodrigoSendinoSanz/01_Hospital_TFG

Para crear la base de datos utilizaremos el archivo que se encuentra en la carpeta
https://github.com/RodrigoSendinoSanz/01_Hospital_TFG/blob/main/MemoriaTFG/Parte-Back/CodigoBBDD.sql

- Creamos la base de datos con el usuario y contraseña **umedico**
- Importamos la carpeta descargada del github en el programa eclipse
- Una vez totalmente importado hacemos click en restart
- y en cualquier navegador escribir en su barra de búsqueda:
http://localhost:8086/



9 Bibliografía

[Textos]

Los textos se han basado en su mayoría en la plataforma de educación de unir y relación propia en base a la formación adquirida.

Aun que también se ha obtenido información en Wikipedia, google y codejava.net

[Imágenes]

La mayoría de las imágenes presentes en el proyecto son de google imágenes, recortes del mismo proyecto o imágenes de canva.

[Diseño web]

Inspiración YouTube OnlineTutorials4Designers