Proyecto Basado en Clic Salud

Curso 2º CFGS A.S.I.R.

Centro Educativo: I.E.S Triana

Fecha: 20/03/2023 – 16/06/2023

Autor: Óscar Tarazaga Gálvez

Índice

[**Introducción**: 2](#_Toc137742928)

[**1º. ¿De dónde surge esta idea?** 2](#_Toc137742929)

[**2º. Idea general.** 2](#_Toc137742930)

[**3º. Herramientas necesarias para el proyecto.** 5](#_Toc137742931)

[**3.1º. Instalación del software.** 6](#_Toc137742932)

[**4º. Desarrollo del proyecto.** 10](#_Toc137742933)

[**4.1º. Desarrollo del apartado del médico.** 12](#_Toc137742934)

[**4.2º. Desarrollo del apartado del paciente.** 22](#_Toc137742935)

[**4.3º. Desarrollo del apartado del administrador.** 29](#_Toc137742936)

[**4.4º. Diseño de la base de datos.** 40](#_Toc137742937)

[**4.5º. Medidas de seguridad frente a posibles ataques.** 42](#_Toc137742938)

[**4.6º. Mapa de la página web.** 43](#_Toc137742939)

[**Problemas encontrados durante el desarrollo:** 44](#_Toc137742940)

[**¿Cómo expandiría esta idea?** 45](#_Toc137742941)

[**Conclusión:** 45](#_Toc137742942)

[**Bibliografía**: 46](#_Toc137742943)

# **Introducción**:

En este proyecto voy a mostrar el proceso y evolución sobre el proyecto, aparte de detallar las herramientas que usaré durante la construcción del mismo.

Este proyecto se basa en una versión simple en un entorno web de clicsalud (Aplicación medica de la Junta de Andalucía), en la cual no solo tendremos la perspectiva del paciente, sino también la del doctor y el administrador.

Este proyecto, lo tomaré como que es a pequeña escala, es decir, como si lo implantase en un solo centro médico y no en todos los de la comunidad autónoma como era la idea original.

# **1º. ¿De dónde surge esta idea?**

Esta idea surge desde hace un tiempo, ya que desde los últimos tiempos de la pandemia ha habido un incremento en el uso de las aplicaciones médicas que usa cada comunidad autónoma.

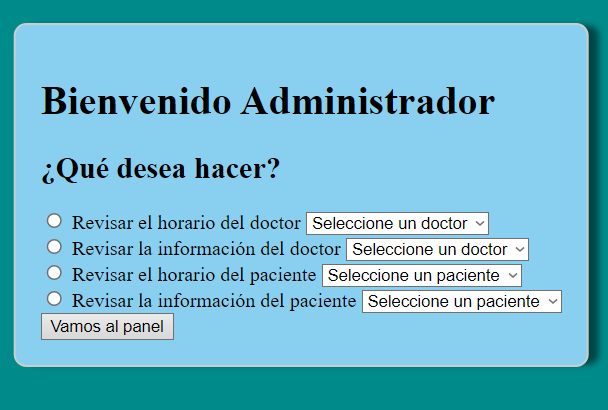
Desde mi punto de vista, este tipo de aplicaciones, tanto la versión web como la versión de móvil, suelen dar fallo, por lo que se me ocurrió el intentar replicar de alguna manera el funcionamiento de este tipo de aplicaciones, ya que me daba bastante curiosidad, por ello, desde que me surgió este pensamiento, estuve dándole vueltas a la idea de ver si yo podía implementar alguna mejora, lo cual voy a tratar de hacer en este proyecto.

# **2º. Idea general.**

Para este proyecto la idea preconcebida que tengo es un entorno web, en el cual, al logarse el usuario, el propio sistema, detecta que tipo de usuario es, en este caso, los tipos solo son 3: Usuario paciente, usuario doctor y usuario administrador.

Cada tipo tendrá su propio panel, es decir, el usuario administrador, por ejemplo, desde el entorno web podrá visualizar los datos tanto de los doctores como de los pacientes, pudiendo hacer algunos cambios, por ejemplo: Tenemos un paciente el cual ha mandado un ticket diciendo que su edad sale mal en el panel donde se le muestra su información, entonces el administrador, desde la opción concreta, podrá modificar esta información del paciente que envíe el ticket.

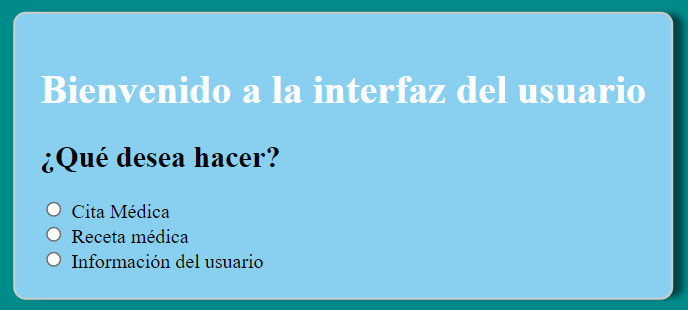
La idea es que cada tipo de usuario tenga varias opciones a realizar, ya sea, siendo administrador, paciente o doctor.



*Imagen de idea preliminar del panel del administrador.*

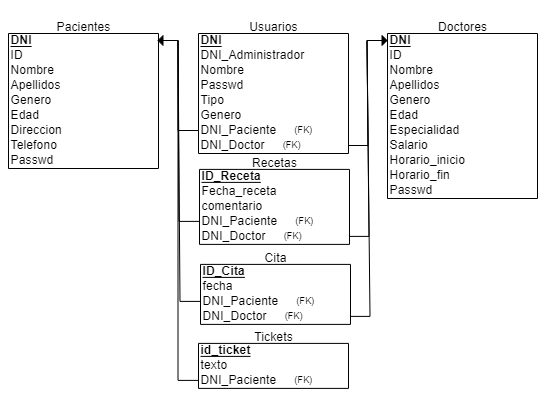


*Imagen de idea preliminar del panel del doctor.*



*Imagen de idea preliminar del panel del paciente.*

Toda esta interfaz web estará conectada con una base de datos, la cual estará documentada tanto en SQL como en su forma de esquema relacional.



*Imagen del esquema relacional de la base de datos*

# **3º. Herramientas necesarias para el proyecto.**

Para este proyecto vamos a usar varias herramientas tanto hardware como software, en mi caso, de hardware la única plataforma que usaré será mi equipo, el cual tiene las siguientes características:

-CPU: Intel Core i5-7300HQ

-RAM: 24 GB

-Almacenamiento: 1 TB en M.2

-Tarjeta gráfica: Nvidia 1050ti

-Placa base: Asus GL553VD

-Sistema operativo: Windows 10 pro

Con esto podemos pasar a hablar del software.

Para el desarrollo del código html, css, php y JS usare Visual Studio Code y para la base de datos, MYSQL Workbench, ya que son 2 aplicaciones que conozco bien como para sentirme cómodo para usarlas durante el proceso de desarrollo, además de poderle instalar plugins los cuales me ayudaran durante todo el proceso.

Para la visualización de las diferentes páginas desarrolladas usare XAMPP, del cual solo necesito php, pero al estar trabajando desde Windows es más sencillo usar esta aplicación directamente.

Como herramientas externas me beneficiaré de la aplicación web Erdplus, la cual permite el diseño de una base de datos desde modelo entidad/relación a SQL pasando por esquema relacional.

Por último, para la gestión de las tareas que tengo que realizar usare trello, y para el control de versiones de la aplicación usaré git hub.

[Enlace de GitHub aquí.](https://github.com/OscarTarazaga/ProyectoFCT)

## **3.1º. Instalación del software.**

En este apartado explicare como se han instalado las herramientas que se usarán durante todo el proceso de desarrollo.

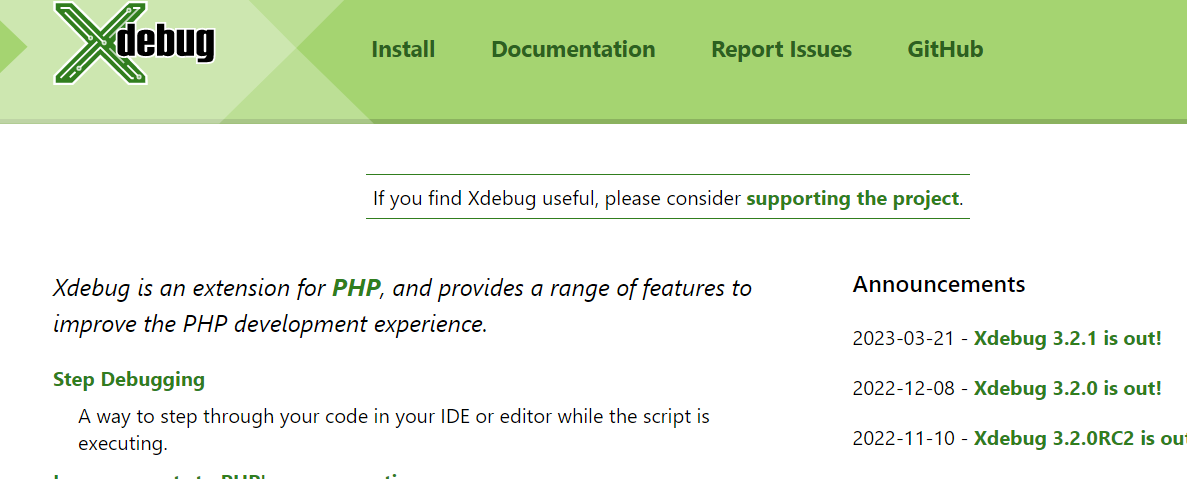
Por ello comenzaremos con MySQL Workbench, el cual es una herramienta visual de diseño y administración de bases de datos.

Este programa es muy sencillo de instalar, ya que tiene los propios ejecutables dispuestos por parte de Microsoft, por lo que solo hay que ejecutar y dar siguiente y siguiente, hasta definir el nombre del usuario que usará esta herramienta, su contraseña y que puerto usará, en este caso el usuario y contraseña son root y el puerto es el 3306, que es el por defecto de mysql.

También se ha instalado Visual Studio Code, el cual es un entorno de desarrollo el cual permite varios tipos de lenguajes, pero para lo que nos atañe, usaré html, css, js y php.

Este programa tiene varios plugins instalados, entre los que se encuentra xdebug, el cual permite de forma eficiente el poder depurar código de manera sencilla.

Este plugin se descarga primero desde la página oficial.



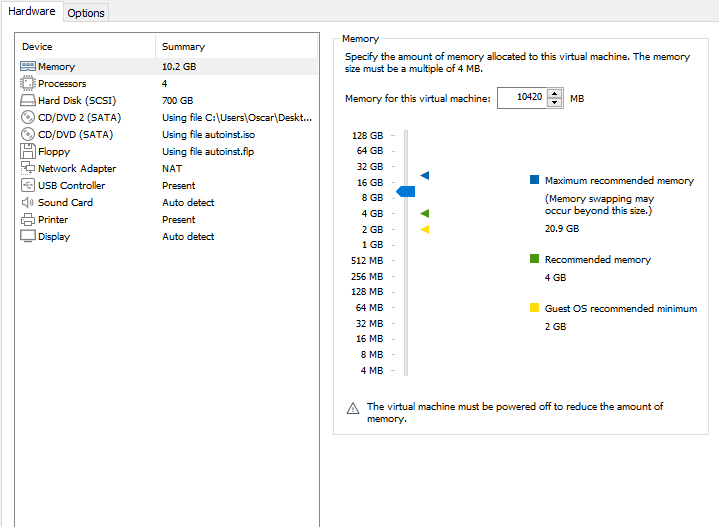
Este fichero con formato .dll se guarda en la ruta en la que este xampp, concretamente dentro de \php\ext y se le renombra como php\_xdebug.dll y se añade la línea Zend\_extension = xdebug, tras esto se reinicia el servicio de apache y con esto estaría instalado, para usarlo desde VScode, seleccionamos primero la opción de depurar, desde la cual se nos abrirá un desplegable para seleccionar el lenguaje que queremos depurar tras esto escribiremos lo siguiente.

En vez de doble contra barra, con una sola bastará y ya estaría el entorno de desarrollo listo.

También mostraré el proceso de instalación de la maquina virtual Ubuntu, la cual en un principio iba a tener la página web en contenedores, pero por problemas inesperados, no ha sido posible.

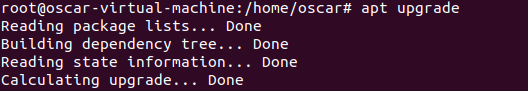
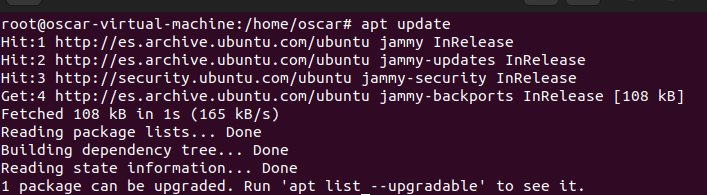
Por último, para la máquina virtual que contendrá los contenedores necesarios para levantar la página web, usariamos VMware Workstation pro, de la cual he adquirido una licencia ya que es una herramienta muy útil para la virtualización.

Primero crearemos una máquina virtual de Ubuntu con estas características:



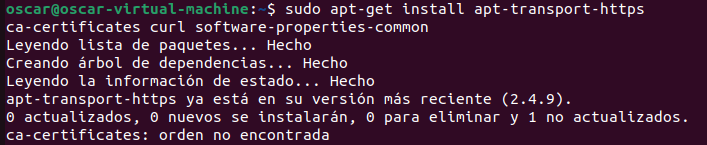
Esta máquina virtual dispondrá de Docker y Kubernetes, ya que son los gestores de contenedores que necesitamos, Docker es muy sencillo de instalar ya que son pocos pasos.

1º. Primero hacemos un update y un upgrade de la máquina.



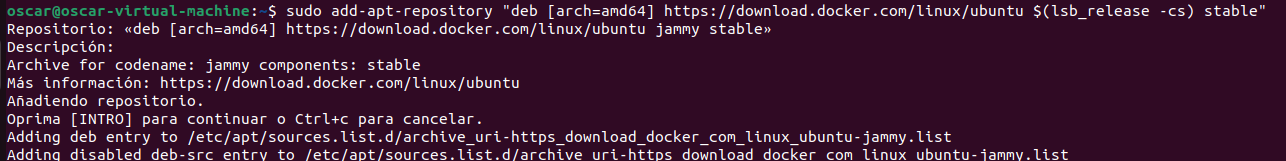
2º. Luego hacemos este comado.

sudo apt-get install \  
ca-certificates \  
curl \  
gnupg \  
lsb-release

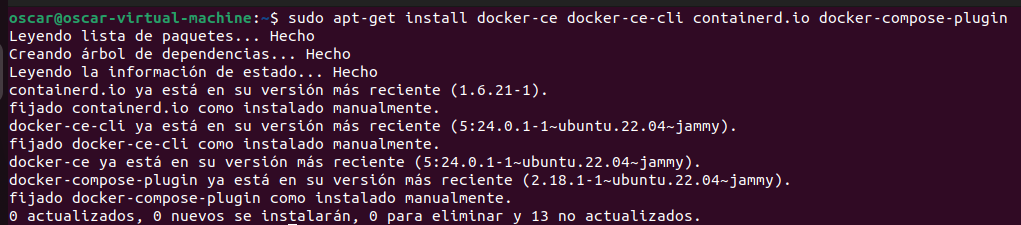


3º. Tras esto hacemos un curl con la siguiente ruta:  


4º. Nos bajamos el repositorio de Docker.



5º. Instalamos Docker-ce y algunos paquetes más de Docker.

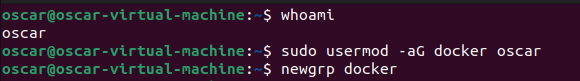


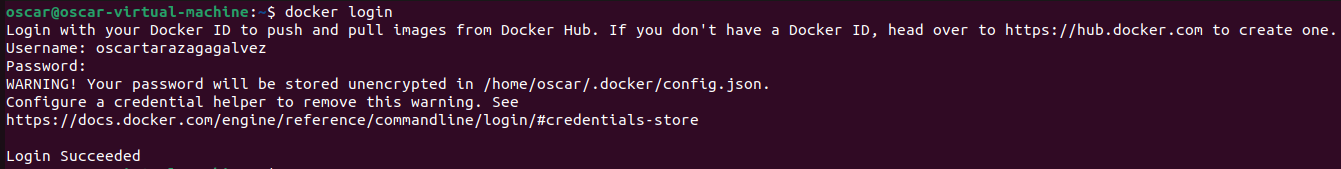
6º. Miramos la versión instalada

7º. Habilitamos Docker.



8º. Hacemos esto para que no haga falta poner sudo cada vez que ejecutamos alguna orden de Docker.





Con esto hecho, ya podemos instalar kubernetes.

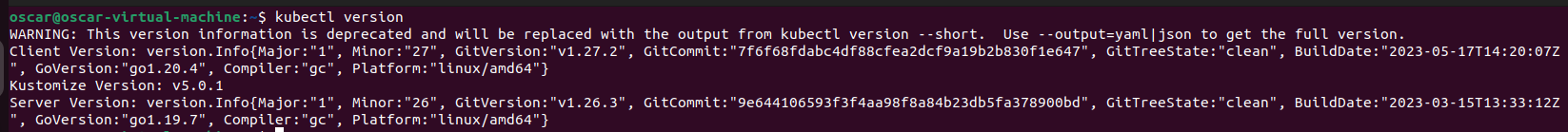
1º. Primero haremos un curl.



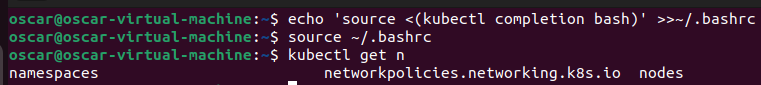
2º. Luego haremos un install del kubectl.



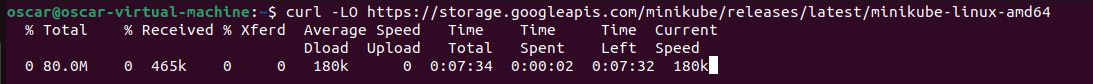
3º. Con esto ya podemos ver la versión instalada.

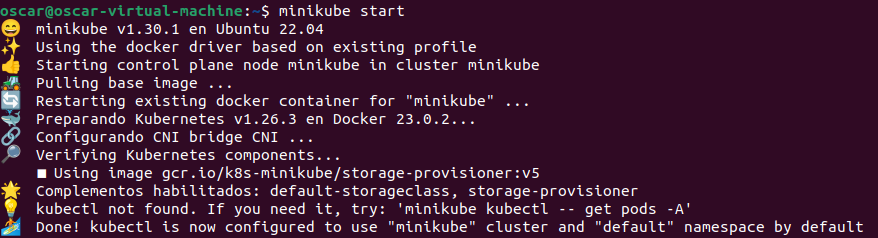


4º. Y por último añadimos kubectl al autocomplete.



5º. Tras esto instalamos minikube y lo iniciamos, esto lo hace automáticamente con docker.





# **4º. Desarrollo del proyecto.**

En la primera parte del desarrollo, empecé con un FrontEnd básico hecho con html y css, el cual está representado de la siguiente manera.



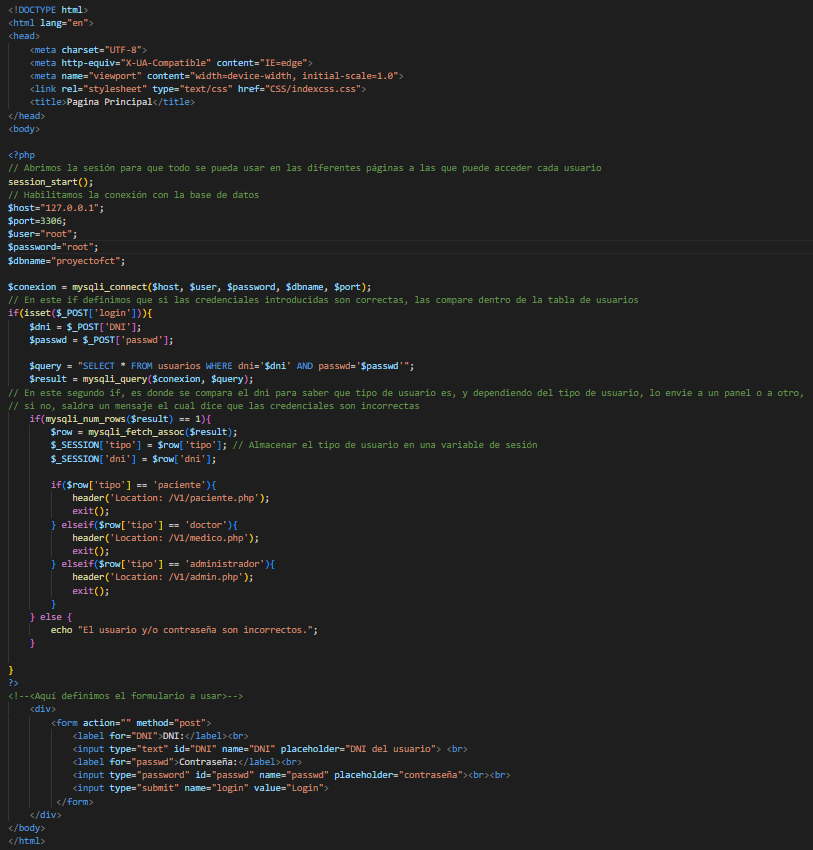
*Imagen de idea preliminar del LogIn.*

En este, tenemos un div el cual contiene el formulario para el sistema de LogIn el cual detecta que tipo de usuario es mediante el DNI.

Este código en php, mostrado a continuación, es el encargado de comparar el DNI introducido, y dependiendo de que usuario sea el DNI le llevará a un panel u otro, también si el usuario se equivoca con la contraseña o el DNI introducido saltará un mensaje de error.

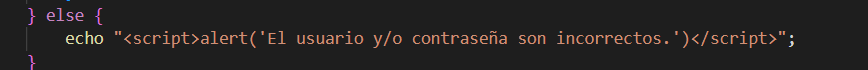
Todo el código desarrollado estará comentado de manera clara dentro del propio fichero con extensión “.php”.

Como podemos ver, hacemos uso de “*IFs*” cuya finalidad es la comparación del dni y la comparación del tipo de usuario que intenta acceder, también podemos observar el uso de headers que son para la redirección a cada página dependiendo del usuario y podemos distinguir el formulario hecho en html y la conexión con la base de datos.

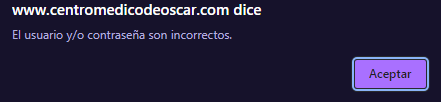


*Imagen del código del LogIn.*

A este código, también muestro como usando de lenguaje JavaScript, esto para la finalidad de que el navegador muestre como ventana emergente el error de que se han introducido erróneamente las credenciales.



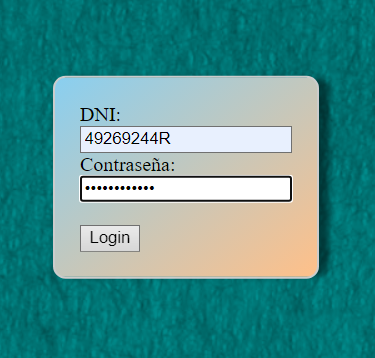
*Imagen del código javascript.*



*Imagen del mensaje que muestra el navegador.*

## **4.1º. Desarrollo del apartado del médico.**

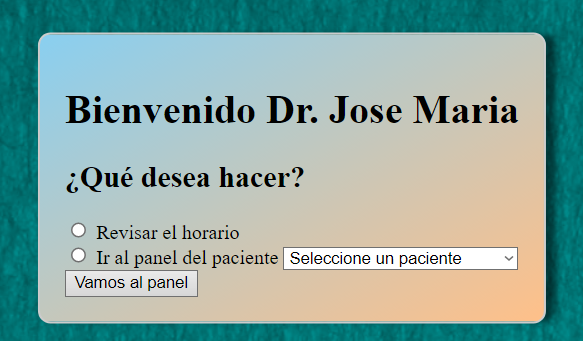
En este apartado explicare el proceso de desarrollo de los paneles/páginas que conforman las opciones que puede realizar el médico.



*Imagen del LogIn.*

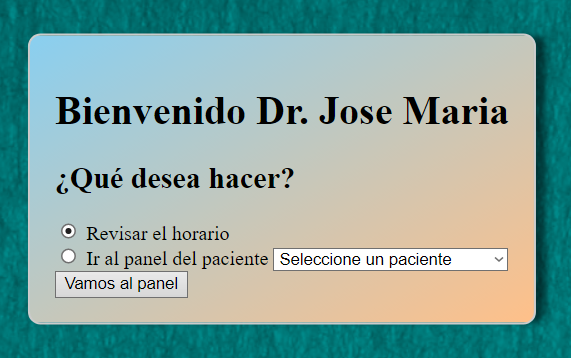
Para empezar al loguearnos como doctor, accederemos al siguiente panel, el cual tiene 2 opciones que son, revisar el horario del doctor para ver que citas tiene asignadas cada día durante la semana actual, y la selección del paciente, desde el cual podrá ver el motivo de la cita, la información del paciente y podrá hacer una receta.

Como podemos ver, al entrar la página usando el dni introducido anteriormente, nos saca el nombre del doctor y si es doctor o doctora.



*Imagen del panel del doctor.*

Si seleccionamos la opción de revisar el horario, nos llevara hasta la página donde seleccionando el campo de horario\_inicio y horario\_fin de la tabla de doctores, nos saca una tabla la cual se genera de lunes a viernes con el horario del doctor, mostrando en el futuro en que horas hay citas asignadas.



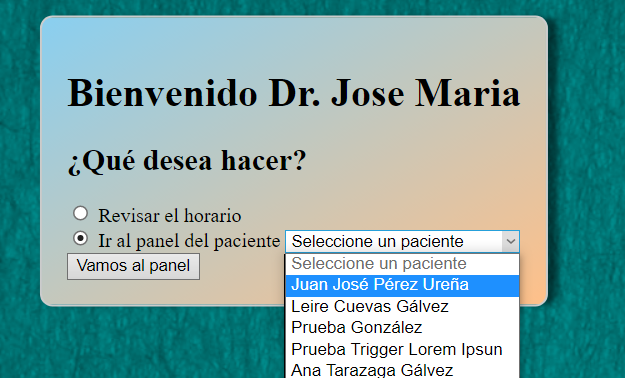
*Imagen de la selección del horario.*

En esta imagen vemos que el horario es desde las 8 de la mañana a las 14 de la tarde de lunes a viernes.



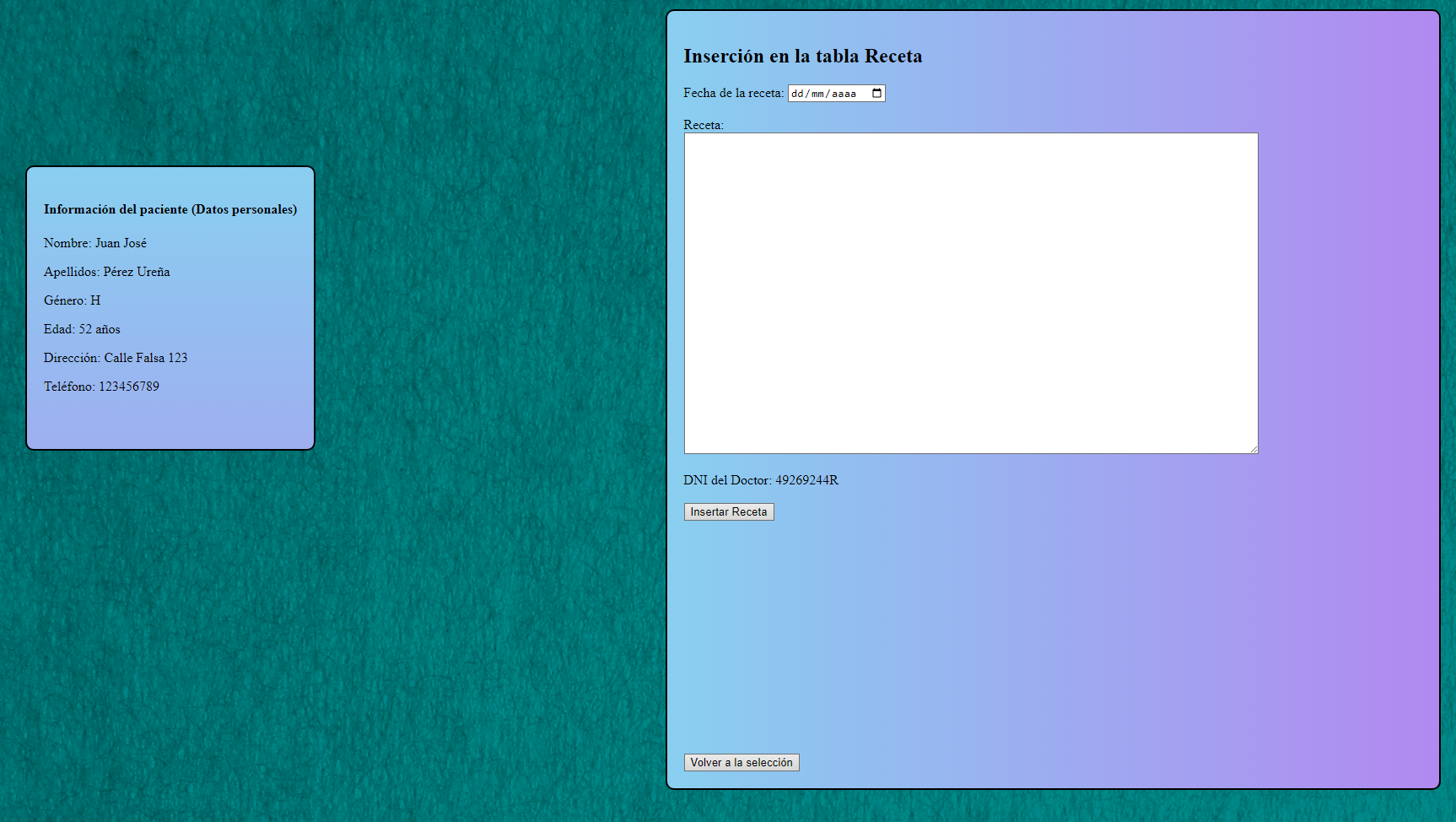
*Imagen del horario.*

Con la siguiente captura observaremos que tenemos un desplegable, el cual muestra a los pacientes registrados en la tabla pacientes, esto lo hace mediante un “SELECT dni, nombre, apellidos FROM pacientes” el cual es introducido en el selector del formulario, al seleccionar el usuario, guarda el dni del paciente en una variable llamada dni\_paciente para usarla en la siguiente página.



*Imagen de la selección del paciente*

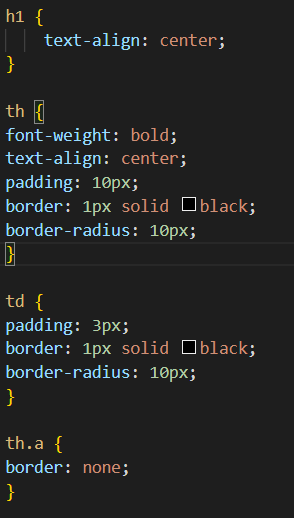
En esta otra captura vemos que tenemos 2 divs, en los cuales tenemos la información del paciente y otro en el que se escribirá la receta del paciente. que se almacenará en su tabla correspondiente.

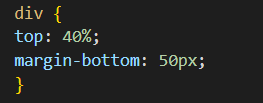


*Imagen del panel del paciente*

A continuación, mostrare el css usado para esta parte, puesto que me llevo algo más de tiempo del esperado por problemas de configuración.

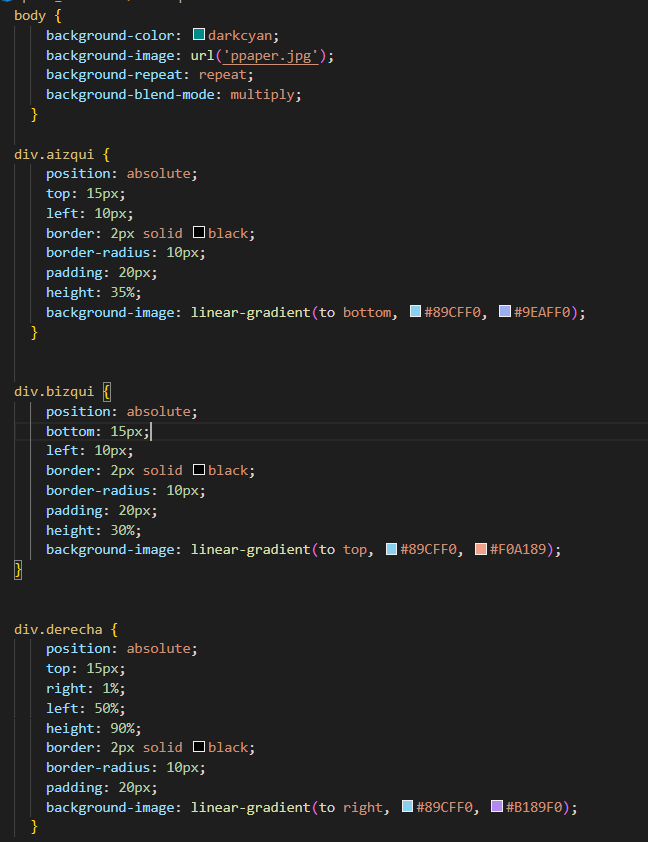
En este en cambio tenemos el formato para el titulo y para los bordes de la tabla, también vemos el del div corregido para esta página y el del botón para volver.



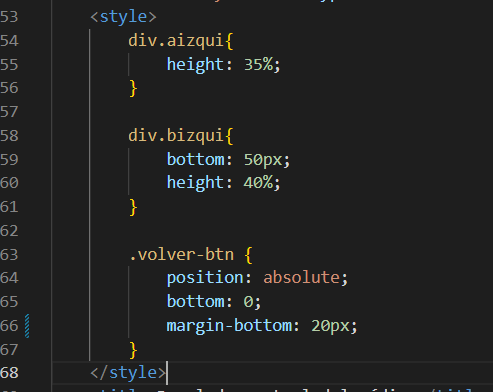


*Imagen del css de medico\_horario.php*

En este último css vemos que los ajustes son respecto al tamaño y la posición de los divs con respecto al css general y en este caso también al de panel\_css.css.



*Imagen del css general de medico\_panel.php*



*Imagen del css de medico\_panel.php*

A continuación, mostraré el código de los diferentes paneles que construyen este apartado.



*Imagen del panel de medico.php*

Este código php abre la sesión y conecta con la base de datos, tras esto, inicializa vacías cuatro variables, las cuales se usarán para almacenar el nombre del doctor, el genero su pronombre y el dni, esto con el fin de que el panel del doctor este un poco personalizado para cada doctor.

Tras esto, verificamos que el tipo de usuario sea doctor y compara el dni que le llega desde el LogIn para ver si de verdad es un doctor.

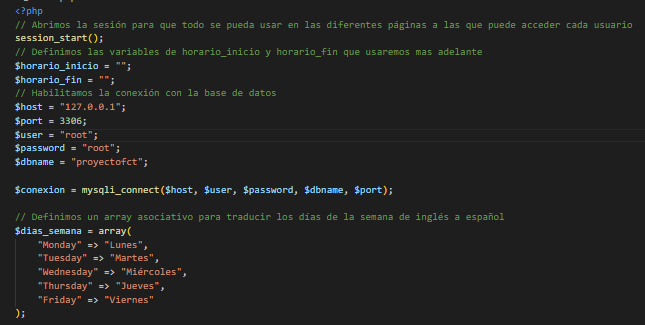
Con esto, lo siguiente que se hace es una consulta la cual devuelve el nombre y el genero del doctor que se haya logueado para mas adelante, comparar el genero para imprimir por pantalla el nombre con el genero con el mensaje de bienvenida.

Tras estas verificaciones lo que hace el código es un if que si se selecciona la primera opción redireccionará al horario, mientras que si se selecciona la segunda y el usuario que se quiere ver, nos redireccionará al panel para la consulta con el paciente.

Si no se seleccionara un paciente, la página dará un mensaje de error avisando de que se debe seleccionar un paciente.

Tras esto pasamos al horario del doctor.

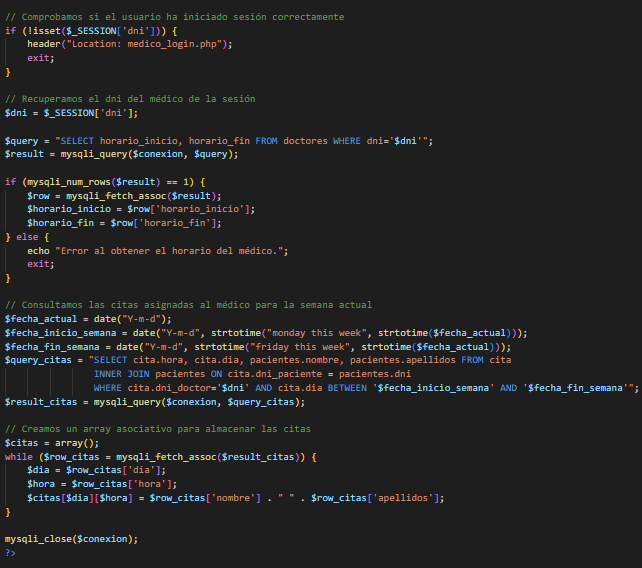
Para empezar, volvemos a abrir la sesión y definimos las variables para el inicio y el fin del horario, también creamos un array asociativo para traducir los nombres de los días de la semana.



*Imagen 1 del horario del medico*

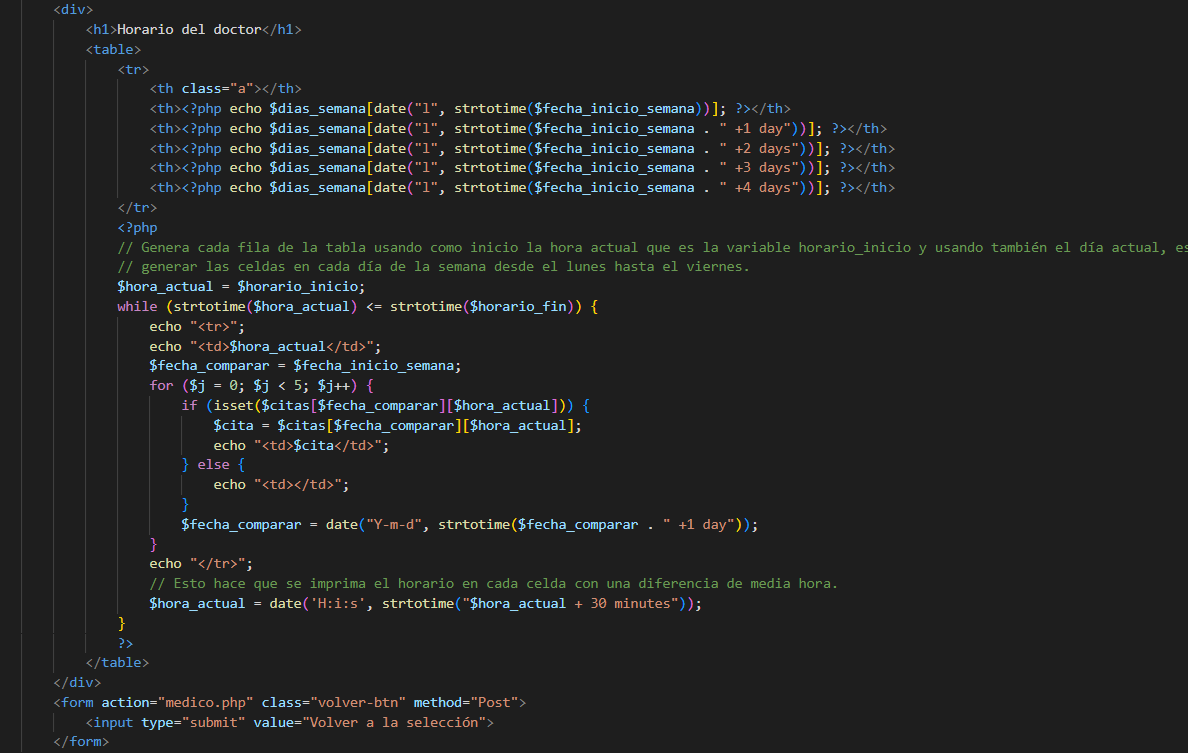
­­­ En esta otra parte del código comprobamos primero que el usuario haya iniciado sesión correctamente.

Luego recuperamos el dni del doctor, ya que dependiendo de que doctor entre, tendrá unas citas asociadas u otras, si no se encontrara el doctor el cual se ha logueado, saldrá un mensaje de error, pero tras 2 comprobaciones esto es muy poco probable.

Con esto, solo queda el tema de imprimir en el horario los nombres y los apellidos de los pacientes que la hayan solicitados y por último creamos otro array asociativo para que se asocien las columnas y filas con los nombres de los pacientes.

*Imagen 2 del horario de medico*

Este otro apartado del código, se genera el horario del doctor



*Imagen 3 del horario de medico*

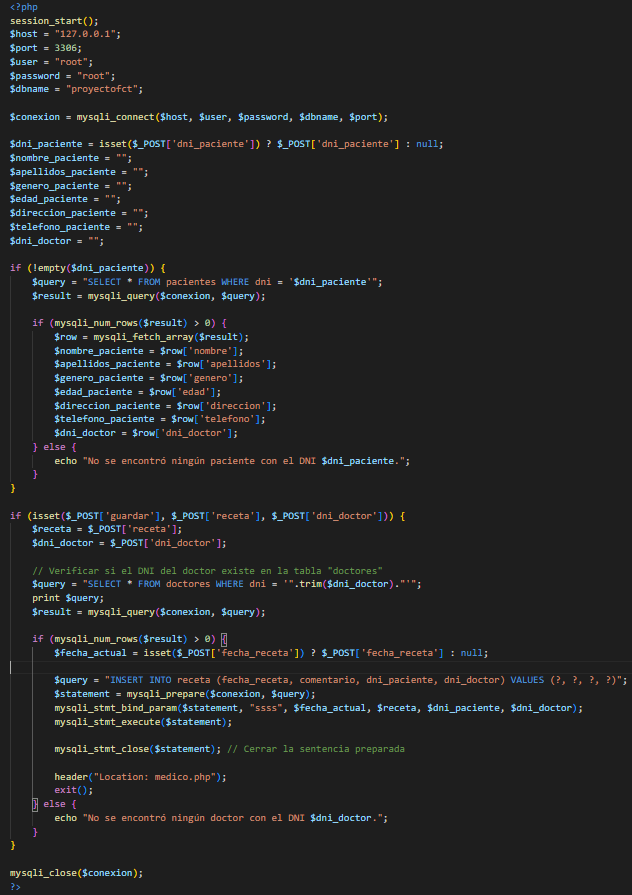
Por último, mostraré el panel del médico.

Para empezar, volvemos a iniciar la sesión y la conexión con la base de datos, tras esto, declaramos las variables necesarias para imprimir por pantalla los datos del paciente.

Tras esto, la consulta trae todos los datos del paciente y los guarda en su respectiva variable.

También podemos ver la configuración detrás del panel el cual insertará la receta en la tabla recetas, la cual coge el dni del doctor desde la sesión y tiene delante la función trim, la cual eliminará los espacios delante del dni del doctor, esto para evitar problemas de caracteres.

Con todo esto, la inserción en la tabla cita se hace mediante un statement para mantener algunas medidas de seguridad a la hora de que no puedan hacer una inyección de SQL mediante la url.

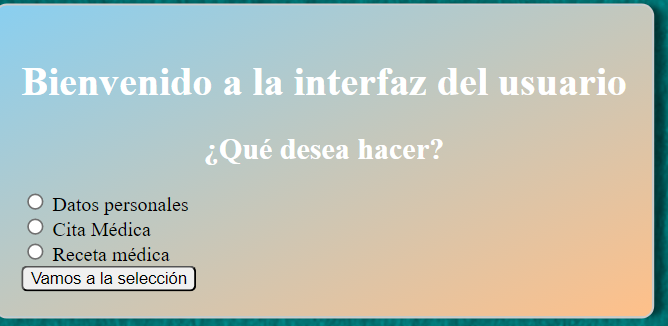


*Imagen del panel del medico*

## **4.2º. Desarrollo del apartado del paciente.**

En este apartado explicare el proceso de desarrollo de los paneles/páginas que conforman las opciones que puede realizar el paciente.

Para comenzar, tenemos el panel con las 3 opciones que puede realizar el paciente.



*Imagen del panel del paciente.*

Como podemos ver, tenemos la opción de visualizar los datos personales, la selección de cita médica y la de ver las recetas médicas que tenga el paciente, a modo de histórico.

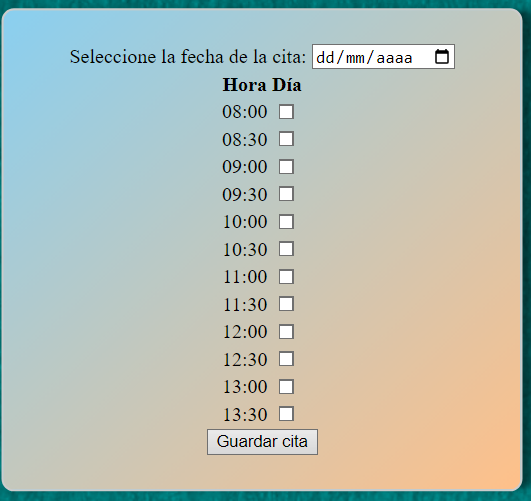
Para comenzar hablaremos de la opción de ver los datos personales, la cual únicamente tiene los datos del paciente y en el futuro una opción para que, si hay algún apartado erróneo, se le pueda enviar un ticket al administrador para que cambie ese dato y asi restringir también para que el paciente no pueda cambiar asi como así sus datos.



*Imagen del panel con la información del paciente.*

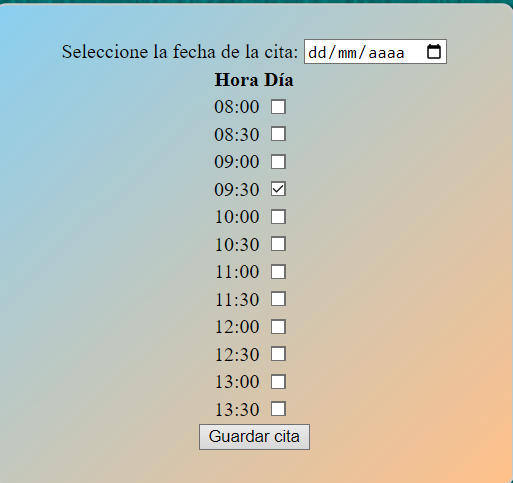
En este caso, podemos ver que este contiene 3 elementos, el elemento date, para seleccionar la fecha, que restringe la selección de fines de semana, la tabla con las horas y el botón de guardar la cita.

A continuación, pasaremos a hablar de la selección de la cita.



*Imagen del panel del horario 1.*

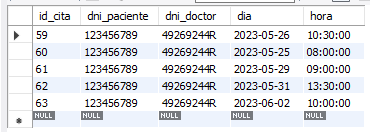
En el caso del panel de la selección de la hora de la cita, este tiene un poco más de miga, pues para que solo se pueda seleccionar un checkbox, he usado JavaScript para que detecte cuando se pulsa una caja y desactive esa columna.



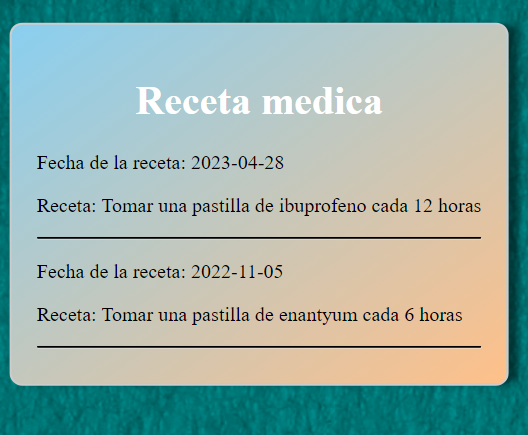
*Imagen del panel del horario 2.*

En este caso, dentro del div que agrupa esta tabla, tenemos el botón de guardar cita, mientras que fuera, tenemos el de volver a la selección, este es, ya que, si el paciente se ha equivocado al seleccionar la opción, que no tenga que cerrar la página y volverla a abrir para seleccionar la otra opción.

Al guardar la cita, la hora y el día se guardarán en la tabla cita de la base de datos usada durante todo el proyecto.



Por último, explicare cómo funciona el histórico de recetas.

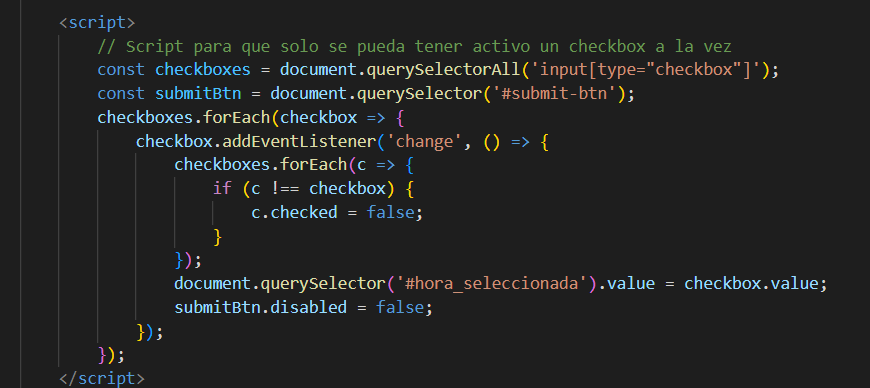


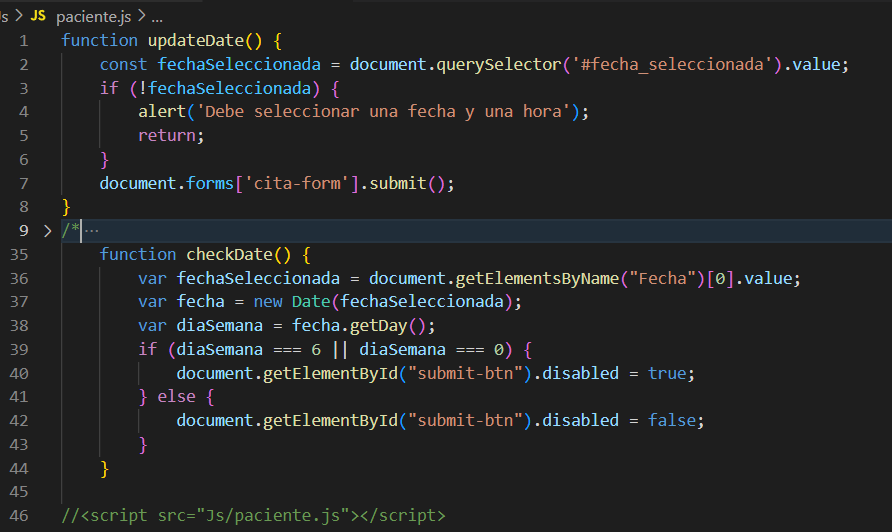
*Imagen del panel del histórico de recetas.*

Como podemos ver en la captura anterior, este paciente consta ahora mismo con 2 recetas, una de 2023 y otra de 2022, como vemos están separadas entre si mediante la etiqueta de HTML llamada “hr”, la intención ya que este es una mera prueba, es que se impriman solo las 3 ultimas recetas, para así no ocupar toda la pantalla con todas las recetas del paciente.

Para el css en este caso es muy similar, por no decir el mismo, que se usa en el resto de entornos.

A continuación, mostrare el código más interesante que tiene esta parte que es para mí el JavaScript, ya que, es la parte que más trabajo me ha costado.





*Imagen del panel del código JavaScript.*

En este caso, mostraré solo el código de Paciente\_citamedica.php ya que es el que tiene mas contenido y al que más horas he dedicado, puesto que se ha intentado que los checkboxes se deshabiliten de manera dinámica pero solo he podido hacer verificaciones para que no se puedan insertar campos vacíos.

Para comenzar, verificamos que el usuario que acceda a este panel sea de verdad un paciente.

Lo que hace este código es seleccionar el dni del doctor, para que se asocie a la cita al igual que el dni del paciente para saber a que paciente debe atender el doctor, tras esto comenzamos con las verificaciones.

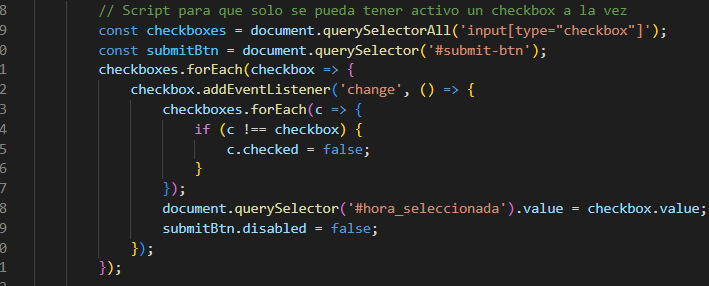
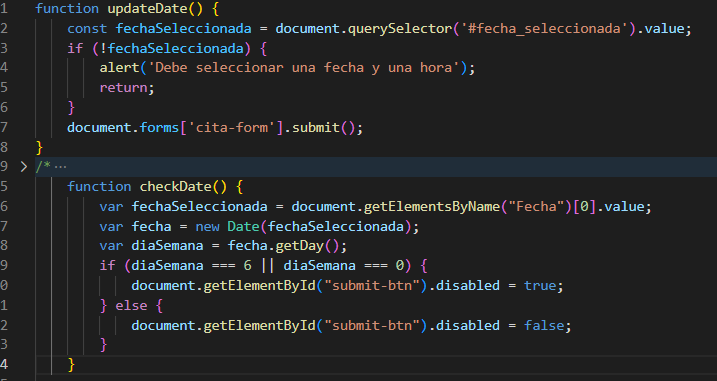
La primera verifica si se ha seleccionado una fecha y un día, si no, no se permitirá hacer la inserción.

La segunda verifica si se ha seleccionado un sábado o domingo, ya que, en este supuesto, no se contempla el que se atienda a los pacientes durante el fin de semana.

La tercera verificación tiene en cuenta la fecha actual para ver si se intenta seleccionar una fecha o una hora anterior a la actual, ya que no tendría sentido que esto se pudiera hacer.

La última verificación impide la inserción de citas duplicadas ya que esto podría causar problemas a la larga.

Este código también consta de código JavaScript, un bloque para impedir la selección de múltiples horas en una misma inserción, otra para ver si la fecha es de un fin de semana y la última poder insertar la fecha en la tabla.

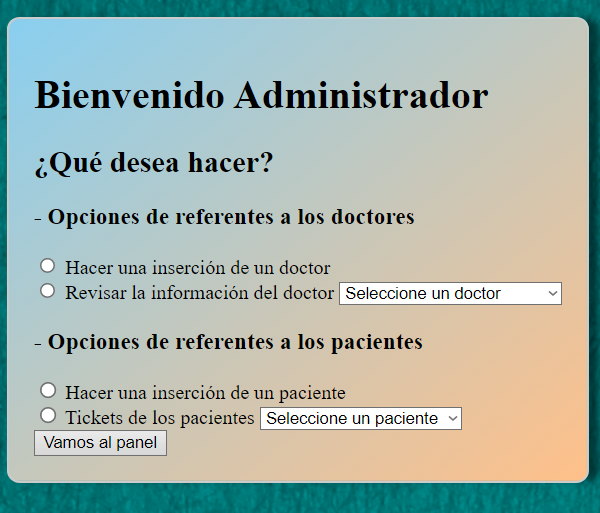




## **4.3º. Desarrollo del apartado del administrador.**

Dentro de este apartado explicare el desarrollo de los paneles del administrador, los cuales son conformados por la inserción de un doctor, la revisión de la información del doctor, la inserción del paciente y la actualización y visión de tickets de un paciente.

Para empezar, tenemos el panel del administrador el cual tiene varias opciones que al ser seleccionadas nos redirigen al panel deseado.



*Imagen del panel del administrador.*

Como podemos observar, el panel está dividido en 2 partes, en las opciones para los doctores y otras para las opciones de los pacientes.

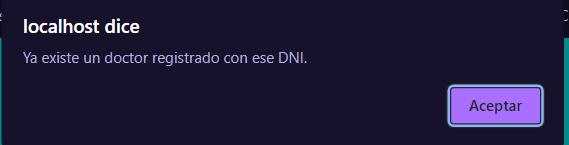
Si nos vamos a la primera opción veremos que tenemos un formulario, el cual al ser rellenado nos insertará en la base de datos un usuario doctor.



*Imagen del panel de inserción de un doctor.*

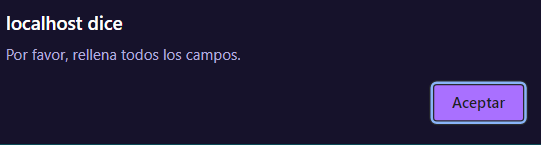
Este formulario al ser rellenado nos devuelve de nuevo al panel principal del administrador.

También si intentamos insertar un usuario cuyo dni ya existe, este dará error y nos pedirá que rellenemos todos los campos y que no repitamos el dni.



*Imagen de error 1.*

Cuando intentemos enviar algún campo vacío nos aparecerá este mensaje de error.

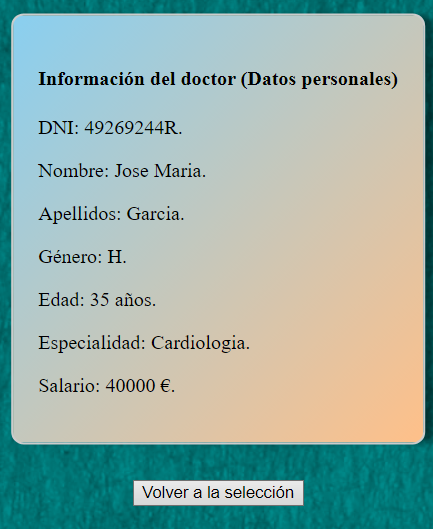


*Imagen de error 2.*

Con la siguiente opción podremos verificar la información de los doctores.



*Imagen de la selección del doctor.*



*Imagen de la información del doctor.*

Pasando ya a la siguiente opción, es lo mismo que la primera solo que para los pacientes.



*Imagen de la inserción de un paciente.*

Cuando los datos son rellenados correctamente son insertados en la base de datos, mientras que si se deja un campo vacío o ya existe un paciente con el mismo dni saldrán errores parecidos a los que salen en el panel de insertar un doctor.

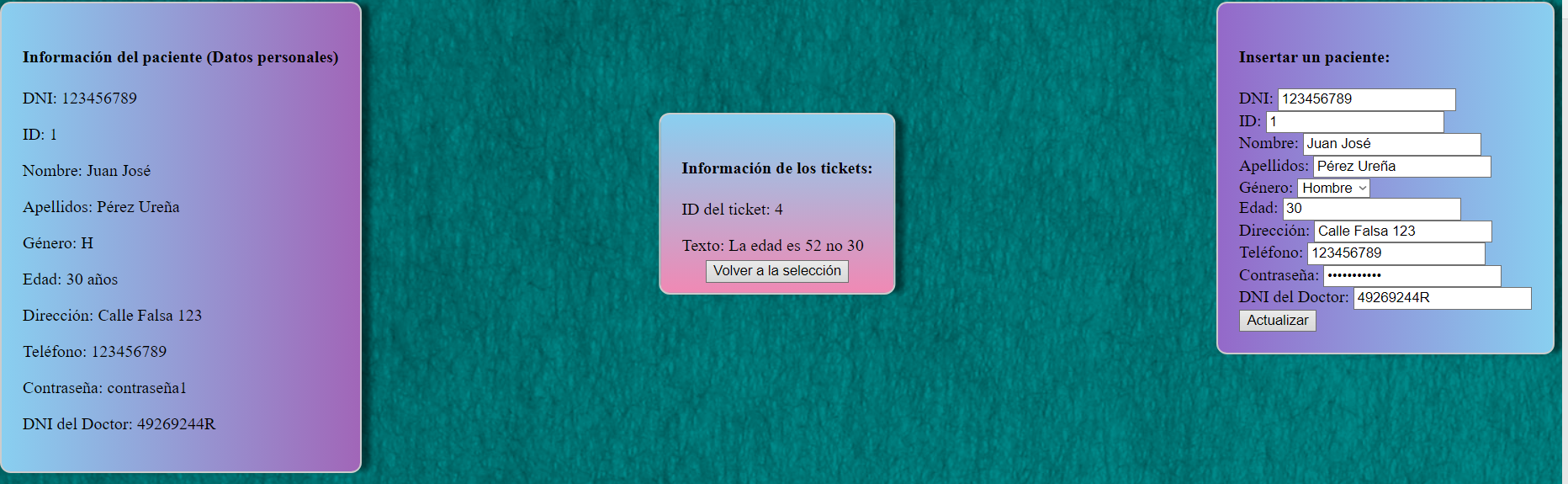
En el siguiente panel tenemos el visor de tickets, por si algún dato de un paciente está mal, que este pueda enviar un aviso al administrador y que este lo cambie.



*Imagen de la selección del ticket del paciente.*

A continuación, en el panel de los tickets tenemos la información actual del paciente, el mensaje donde pide la corrección y el formulario que pide los datos para la actualización.

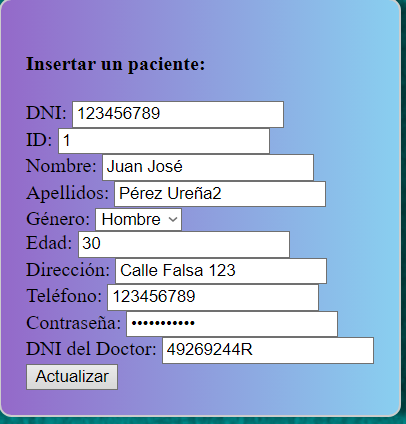
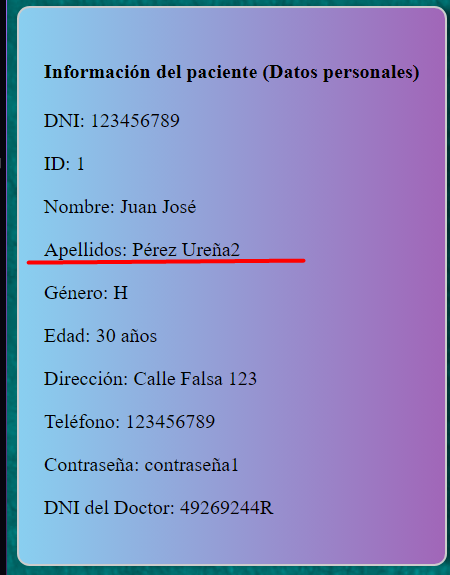
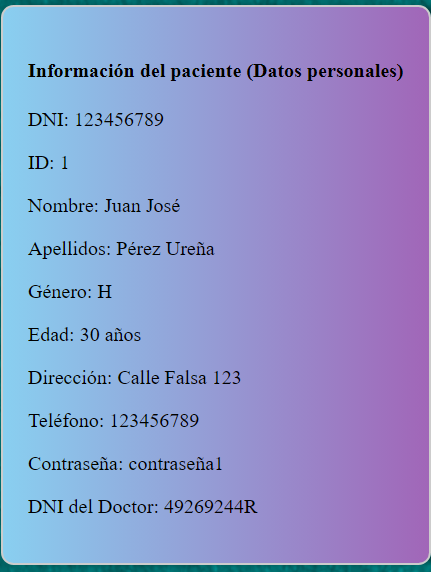
Como podemos ver en la imagen, en el div que actualiza los datos del paciente, mete automáticamente los datos tal y como están en la tabla, para que no quede ningún campo vacio.



*Imagen del panel de los tickets.*

Al hacer alguna actualización, los datos se actualizan tanto en la tabla de pacientes como en la tabla de usuarios.

A continuación, mostrare como se actualizan los datos de los pacientes.

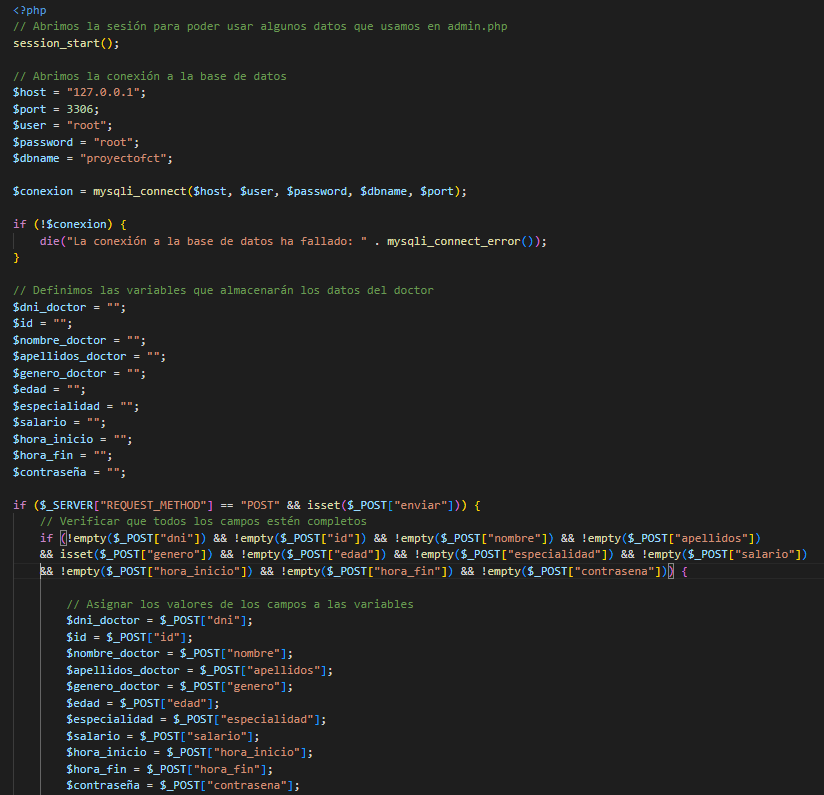


*Conjunto de imágenes de como se actualizan los datos del paciente.*

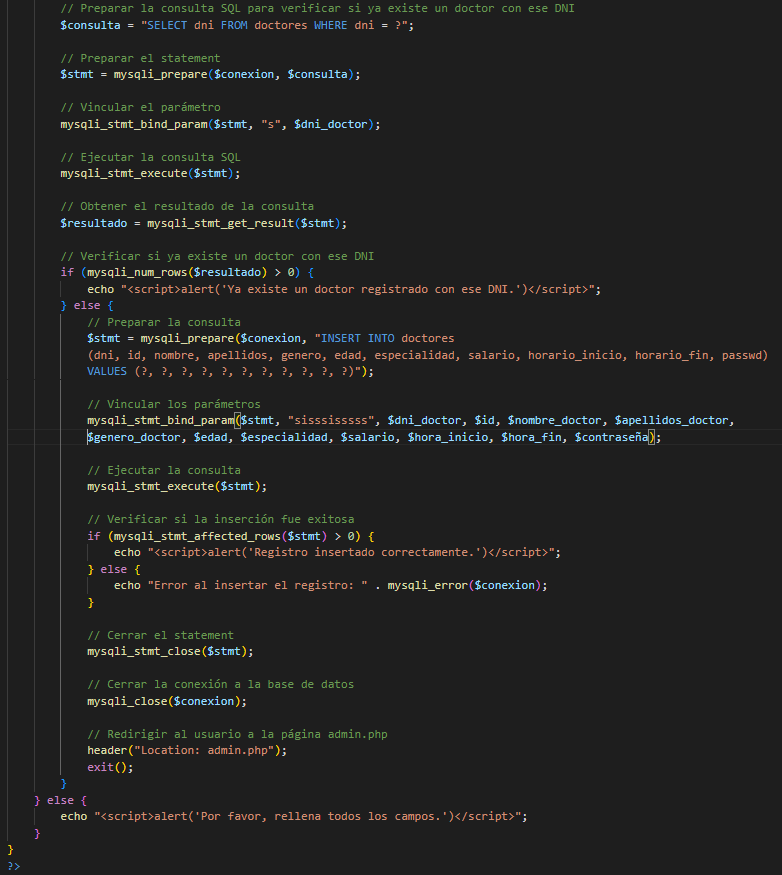
A continuación, pasare a explicar los códigos de cada panel menos el principal del administrador y del que muestra la información del doctor, puesto que es igual al resto de paneles principales de cada tipo de usuario y es igual al resto de paneles que muestran la información del usuario.

Comenzando por el panel de la inserción de un doctor, en este caso comenzamos el código inicializando las variables vacías que contendrán los datos del doctor a insertar, tras esto, tenemos una verificación para que ninguno de los datos se quede vacío, si esto ocurriera, saltaría un error.

Tras esto, tenemos otra verificación, esta para ver si el dni del doctor que se intenta introducir ya existe, con todo esto, preparamos la inserción, vinculamos los parámetros para el statement y por ultimo ejecutamos la consulta, con esto, verificamos si la inserción se ha hecho bien, si no lo hiciera, saldría un error por pantalla.

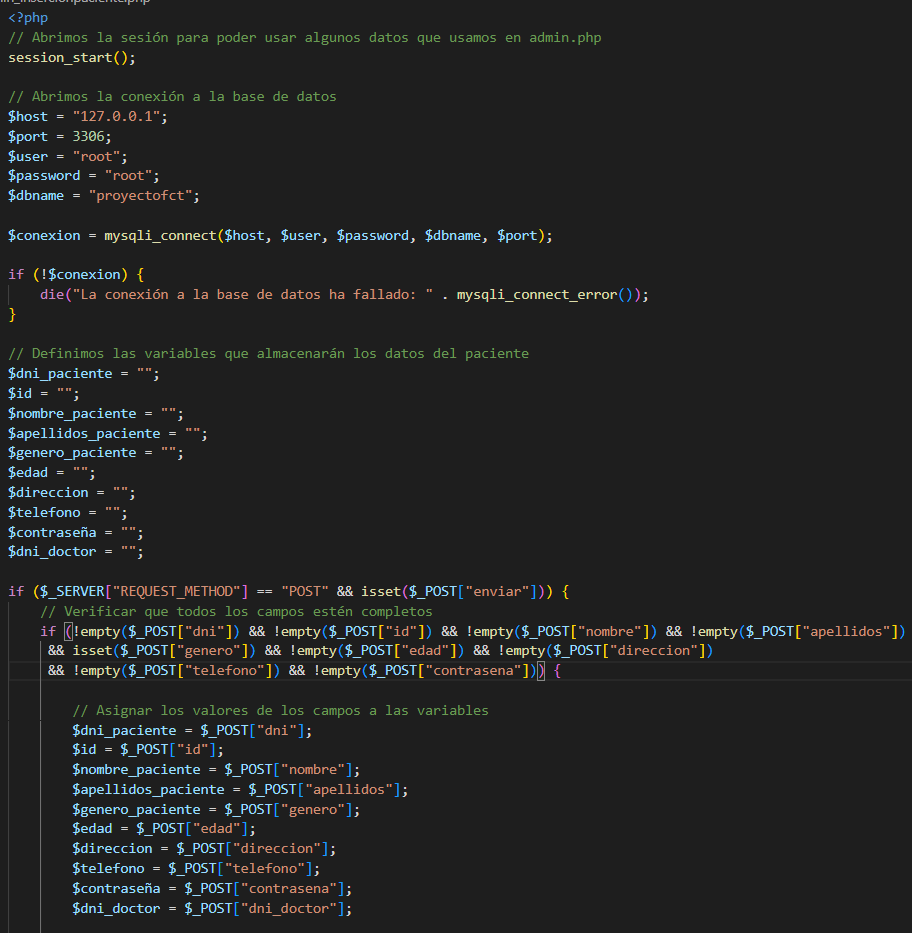


*Imagen 1 del código de admin\_insercion.php.*

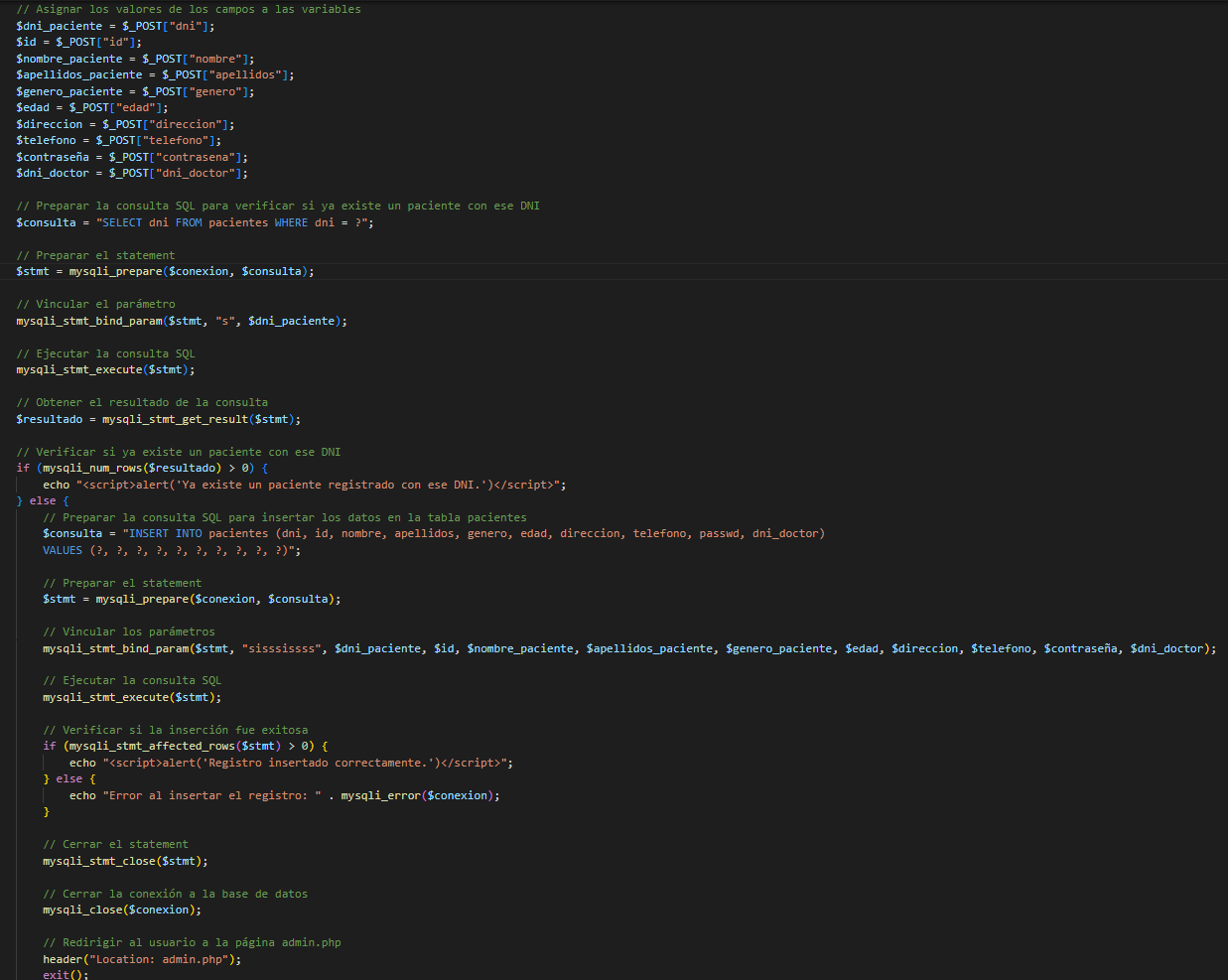


*Imagen 2 del código de admin\_insercion.php.*

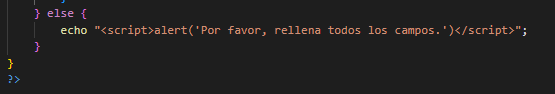
Con esto podemos pasar al panel de inserción de un paciente, el cual es igual que el del doctor solo que cambian los datos por los del paciente.



*Imagen 1 del código de admin\_insercionpaciente.php.*



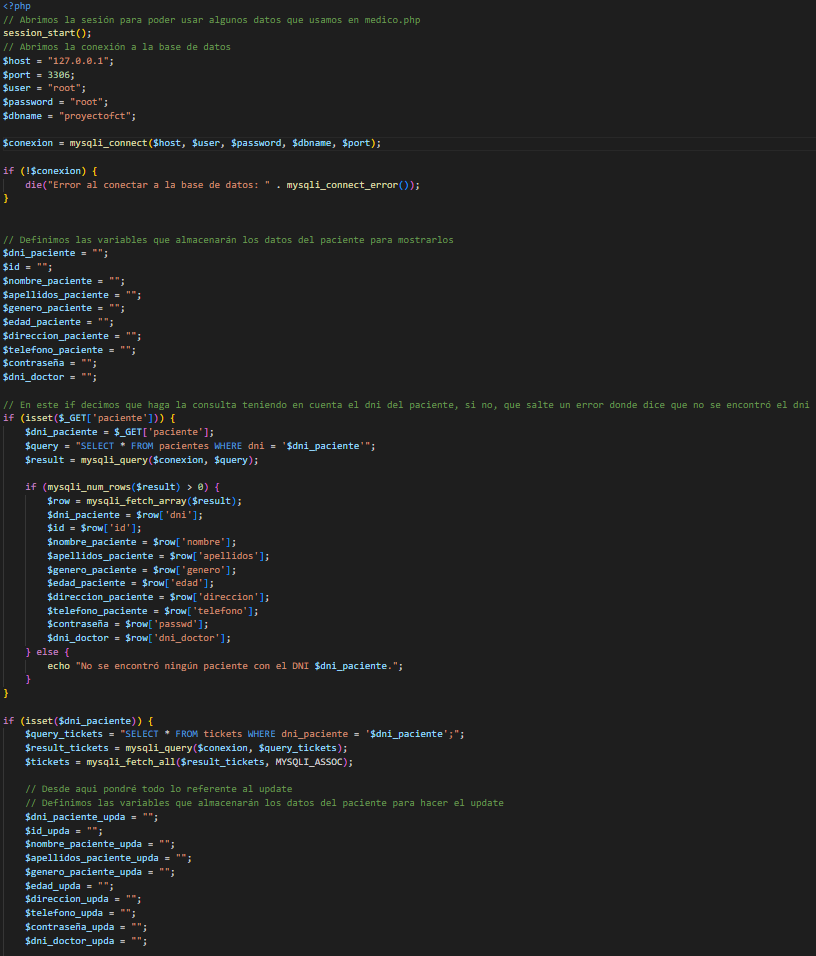
*Imagen 2 del código de admin\_insercionpaciente.php.*



*Imagen 3 del código de admin\_insercionpaciente.php.*

Por último, tenemos el código del panel en el que el administrador puede cambiar los datos de un paciente.

La primera parte del código es básicamente para mostrar los datos actuales del paciente el cual ha hecho un ticket para que le cambien sus datos, por ello, teniendo en cuenta el dni del paciente que haga el ticket, imprimimos sus datos, después de esto, los mostramos también en el panel que permite la modificación de los datos y hacemos la operación update mediante statements.



*Imagen 1 del código de admin\_tickets.php.*



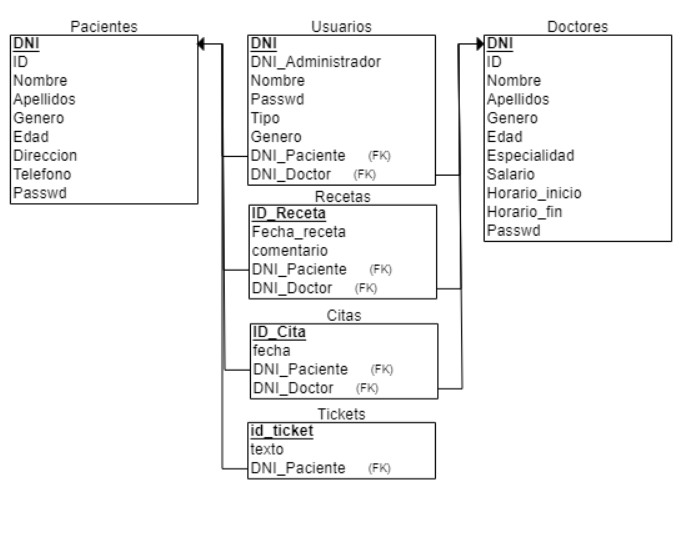
*Imagen 2 del código de admin\_tickets.php.*



*Imagen 3 del código de admin\_tickets.php.*

## **4.4º. Diseño de la base de datos.**

Este proyecto usa una base de datos en MYSQL, esta contiene las tablas de paciente, doctores, usuarios, recetas, horarios, citas y tickets.



En esta imagen podemos ver el modelo relacional de la base de datos, viendo cada conexión de manera clara y concisa.

Para la funcionalidad del LogIn, usamos la tabla usuarios, la cual reúne a los 3 tipos de usuarios para poder verificar el tipo de usuario que intenta entrar.

Para la tabla recetas, almacena el id de la receta, que es auto incremental, tenemos la fecha de la receta, para poder ordenarla en el panel del paciente, también almacena la receta dentro del campo comentario y los dni del paciente y del doctor.

La tabla citas es usada para la asignación de una cita por parte del paciente y la revisión de esta por parte del doctor.

Por último, la tabla tickets almacena la incidencia que tiene el paciente con sus datos, para que más adelante el administrador pueda cambiar los datos erróneos del paciente.

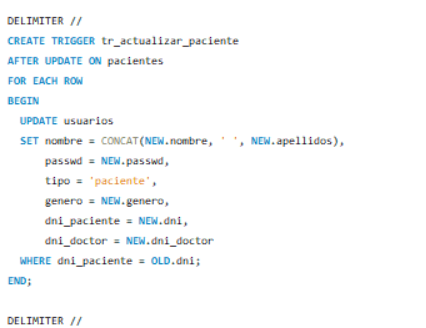
Para la funcionalidad de la base de datos en conjunto con la página web, se han diseñado algunos triggers, los cuales explicare a continuación.

Este trigger insertará el doctor que se acaba de insertar en la tabla doctores en la tabla usuarios.

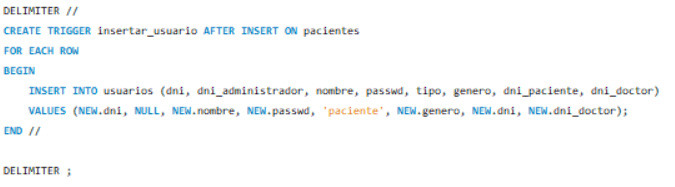
Esto con el fin de no tener que hacerlo de forma manual.



Este otro trigger se creó con la intención de que, al actualizar los datos del paciente en la tabla pacientes, también se actualicen en la tabla pacientes.



El ultimo trigger inserta los pacientes que se inserten en la tabla pacientes en la tabla usuarios.



## **4.5º. Medidas de seguridad frente a posibles ataques.**

Como medidas de seguridad para la página web desarrollada he añadido las siguientes características:  
-Validaciones: A la hora de hacer una inserción en la base de datos mediante la página he añadido validaciones, es decir, cada vez que se intenta insertar algo por ejemplo en la tabla pacientes, comparará la clave primaria que en este caso es dni, para que no haya redundancia de datos, ya que esto podría generar problemas graves al realizar las consultas.

-Inyecciones SQL: Para impedir las inyecciones de SQL mediante la url he añadido statements, los cuales son comandos que se ejecutan dentro de la página para realizar acciones específicas, en este caso, prepara la consulta la cual insertará los datos, esto ha sido definido para decirle a las variables que recibirán los datos que tipo de dato deben esperar, y si reciben otro tipo de dato saltará un error.

## **4.6º. Mapa de la página web.**

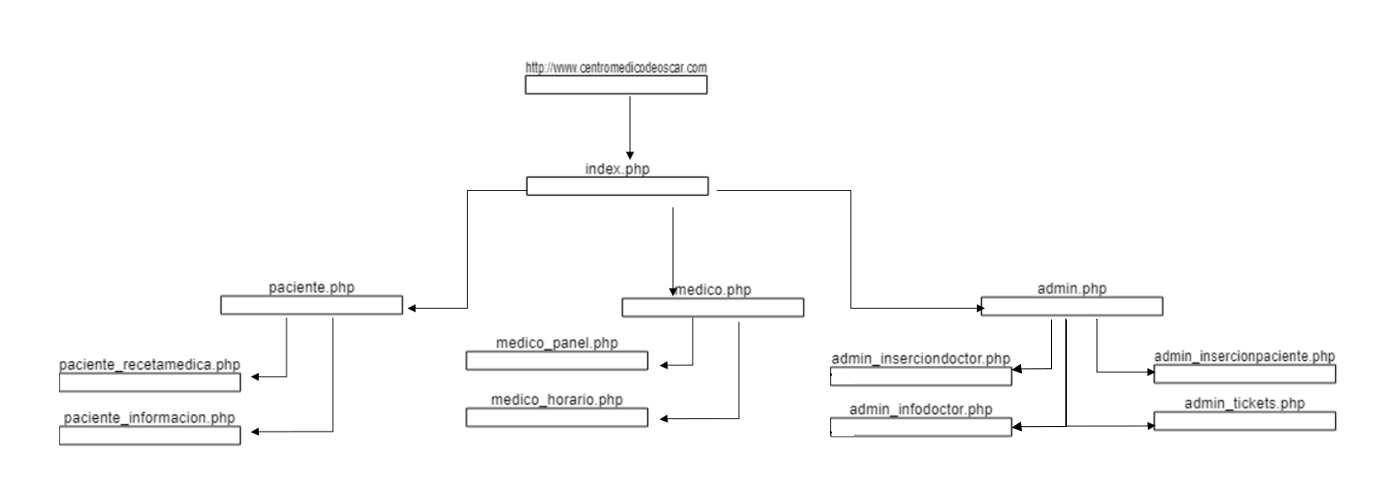
En este apartado mostrare el mapa del sitio web al que he llamado [www.centromedicodeoscar.com](http://www.centromedicodeoscar.com) como vemos en la imagen, al escribir esta dirección en el navegador, entraremos al index de la web, desde donde podremos acceder a cada panel dependiendo de que usuario loguee a la página, dependiendo de que usuario entre en la página entraremos al panel principal de cada usuario.

Por ejemplo, si entráramos el administrador quisiera revisar los tickets de los pacientes que han destacado que sus datos están mal, el camino que seguirá seria:  
1º. Entrar en la web donde llegará al Index.

2º. Loguearse y entrar al panel del administrador.

3º. Seleccionar el panel de tickets.

Desde aquí, podremos acceder a los subpaneles que contienen la verdadera funcionalidad de la página.



# **Problemas encontrados durante el desarrollo:**

1º. A la hora de imprimir por pantalla los datos del paciente en la página medico\_panel.php, no conseguía sacar los datos del paciente y esto era debido a que no se estaba guardando el dni del doctor desde medico\_php y por lo tanto al pasarlo no se detectaba, esto fue corregido creando la variable $dni\_paciente e indicando que su valor será el dni seleccionado en el selector del formulario.

2º. A la hora de implementar la tabla con el horario para que pidan cita los pacientes, no se insertaban los datos, el problema era que el formulario estaba fuera de la tabla, por lo que no se enviaban los horarios seleccionados, concretamente la hora seleccionada mediante el checkbox, para ello moví todo el formulario dentro del div y le tuve que agregar el atributo name a los checkboxes para que se pudieran enviar correctamente y con todo esto ya funcionó.

3º. A la hora de implementar la función de insertar la receta en la tabla he estado teniendo errores, puesto que no hacia correctamente la inserción.

El problema era la sentencia del statement que he estado usando, ya que estaba definiendo el campo fecha como int en vez de como string, también había otro error que no conseguí solucionar que al recargar la página los datos del paciente desaparecían, al final también tuve algunos errores que no supe identificar y la solución fue crear desde cero el panel, el cual ha quedado mejor optimizado y con una funcionalidad completa.

4º. Al momento de intentar hacer los despliegues de los contenedores cuya idea tenía contemplada desde el inicio, no he podido hacerlo, puesto que me daban muchos problemas a la hora de configurarlos o intentar hacer la instalación de los servicios necesarios para el despliegue de la aplicación web, y al llegar a conseguirlo todo, no se visualizaba bien la pagina desde los contenedores por varios problemas con la resolución y algunos otros problemas que no he sido capaz de solucionar.

5º. Al intentar hacer que la página tuviera componentes dinámicos, he intentado implementar Ajax, el cual me ha dado varios quebraderos de cabeza, ya que no he conseguido el que se comuniquen bien los códigos involucrados, y cuando he conseguido que funcionen, ha deshabilitado todos los checkboxes en vez de solo los que tienen la misma hora y fecha que los que están almacenados en la tabla cita.

# **¿Cómo expandiría esta idea?**

Para expandir este proyecto o como lo haría de otra manera, para empezar, seguramente montaría todo usando Python y Django, los cuales son un lenguaje de programación de alto nivel, y Django que es un framework de desarrollo web de alto nivel escrito en Python.

Esto ayudaría y optimizaría la velocidad de las consultas a la base de datos y una mejora del LogIn bastante contundente.

También me gustaría poder implementar la funcionalidad del calendario dinámico con Ajax, ya que es algo con lo que me he quedado con ganas de poder practicar y que realmente sería más útil que lo que está implantado ahora mismo, ya que, si están todas las citas ocupadas, el usuario deberá ir una por una para ver si hay una libre.

# **Conclusión:**

La conclusión de que saco de este proyecto es que igual intente ser demasiado ambicioso, pues creí que podría hacer la página y meterla en contenedores para emular el funcionamiento de un servidor, de lo que he aprendido que el desarrollo de una página web es más complejo de lo que parece incluso para una web sencilla como esta, como bien explico en el apartado anterior esta idea seguramente estaría mejor implementada si se implantara usando el framework Django, a pesar de que esto seria ya un proyecto enfocado a un modulo de grado superior de desarrollo de aplicaciones web (DAW).

Aunque si que saco en claro algunos conocimientos más exhaustivos de JavaScript y php, a parte de como implementar medidas de seguridad para una página web para evitar la inyección de SQL.

# **Bibliografía**:

-Enlaces seguidos para la redirección a las diferentes páginas:

<https://code.tutsplus.com/es/tutorials/how-to-redirect-with-php--cms-34680>

<https://help.dreamhost.com/hc/es/articles/214395288-Crear-un-redireccionamiento-de-PHP>

<https://www.factoriacreativabarcelona.es/blog/redireccion-php/>

-Enlaces seguidos para instalar Docker y Kubernetes:  
 -Docker:

<https://sergioss.net/instalar-docker-ce-en-ubuntu-22-04-lts-jammy/>

-Kubernetes:  
 <http://localhost:51235/unidad2_2> (Manuales dados por el profesor

Nicolás José del Pozo Madroñal durante el segundo trimestre)

-Enlaces seguidos para el balanceo de carga con Kubernetes:

<https://www.youtube.com/watch?v=0iMEcrcfG5A&ab_channel=PeladoNerd>

<https://www.youtube.com/watch?v=2i5t6bfgWvE&ab_channel=PeladoNerd>

<https://kubernetes.io/es/docs/concepts/services-networking/service/>

<https://www.ovhcloud.com/es-es/public-cloud/kubernetes/kubernetes-load-balancer/>

-Enlaces seguidos para el uso de Ajax:

<https://www.youtube.com/watch?v=DsI7V49JFQY>

<https://www.youtube.com/watch?v=j3vK2I5LoBw>

<https://www.youtube.com/watch?v=MeCEJOVJyDs>

<https://developer.mozilla.org/es/docs/Web/Guide/AJAX>

http://thinkcoderepeat.blogspot.com/2006/02/tutorial-de-ajax-con-php-y-json.html