

Sanitech

Albert Mateos - Andres Rojas - Diego Amador - Samuel Garcia

Índice de contenido

1. Introducción	1
1.1. Miembros del equipo	1
1.2. Objetivos de la aplicación.....	1
1.3. Necesidades.....	1
1.4. Público objetivo.....	1
1.5. Tecnologías con las que hemos trabajado	2
2. Diseño de base de datos y diagramas de clases.....	3
3. Diseño y guía de estilos	4
3.1. Sketching y Mockups de interfaces de usuario.....	4
3.2. Guia de estilos.....	4
4. Manual de instalación, distribución y configuración.....	5
4.1. Despliegue en local	5
4.2. Instancia EC2 AWS (Amazon Web Services).....	9
5. Manual de usuario	16
5.1. Usuario Paciente	16
5.2. Usuario Médico	23
5.3. Usuario Administrador	27
6. Datos de prueba para testear la aplicación	34
7. Lineas futuras	35
7.1. Requerimientos pendientes	35
7.2. Posibles mejoras	35
8. Conclusiones	36
8.1. Desviaciones en la Planificación	36
8.2. Aportaciones del Proyecto a los Conocimientos del Alumno	36
9. Webgrafía	38

Chapter 1. Introducción

Os presentamos la memoria del tercer proyecto del curso 2023-2024 de 2º de Desarrollo de Aplicaciones Web del instituto Nicolau Copèrnic. Este proyecto consta del desarrollo de una página web de gestión de pruebas de un centro médico, desarrollado para los estudiantes del instituto Blanxart de Terrassa.

En este documento presentaremos el equipo de trabajo, las especificaciones y requerimientos del proyecto, el manual de instalación de la aplicación y diversas consideraciones para el futuro.

1.1. Miembros del equipo

- Albert Mateos, estudiante de 2º de DAW en instituto Nicolau Copèrnic.
- Andres Rojas, estudiante de 2º de DAW en instituto Nicolau Copèrnic.
- Diego Amador, estudiante de 2º de DAW en instituto Nicolau Copèrnic.
- Samuel Garcia, estudiante de 2º de DAW en instituto Nicolau Còpernic.

1.2. Objetivos de la aplicación.

Esta aplicación tiene como objetivo la creación de una aplicación para la gestión de citas y pruebas de un centro médico.

1.3. Necesidades.

Las necesidades a las que esta aplicación trata de dar respuesta son las siguientes:

1. Preparación de pruebas médicas. Permitirá a los pacientes tener acceso a diversas herramientas audiovisuales para la preparación de sus pruebas médicas agendadas.
2. Asignación de citas y pruebas médicas. Ofrecerá un sistema de gestión para la creación y asignación de citas y pruebas médicas para los pacientes por parte de estos mismos, también por parte de los médicos y los administradores.
3. Recordatorios y confirmación de citas médicas. La aplicación contará con sistemas de notificaciones, donde los pacientes podrán aceptar o cancelar sus citas médicas, una vez estas les han sido asignadas y también con recordatorios de las citas que ya fueron aceptadas.
4. Interfaces de usuario diferentes. La aplicación contará con interfaces de usuario diferentes para cada uno de los roles que existan dentro de la misma.
5. Acceso a los justificantes médicos. Desde la aplicación, los pacientes podrán descargar los justificantes médicos de sus citas o pruebas previamente realizadas.

1.4. Público objetivo.

Esta aplicación está diseñada considerando dos *targets* diferentes.

Por un lado, tenemos al personal médico que necesitan gestionar las pruebas y citas médicas de los

pacientes de un centro médico, podrán crear y asignar citas y pruebas medicas a sus pacientes, tambien ver los resultados de las pruebas e historial de citas de los pacientes y la agenda de cada uno de los médicos del centro. Se trata de un público profesional, que lo que necesita es realizar su trabajo de la forma más cómoda, eficiente y familiar.

En el otro lado, encontramos a los pacientes, pacientes con un amplio rango de edades, que necesitan crear, consultar, aceptar y cancelar sus citas y pruebas medicas, tambien generar de forma sencilla e intuitiva los justificantes de sus asistencias medicas realizadas, tambien recibir recordatorios de sus citas medicas a realizar, ademas de tener acceso a material audio visual para la preparación de sus pruebas medicas .

1.5. Tecnologías con las que hemos trabajado

Para la realización de este proyecto hemos tenido que trabajar con tecnologías muy diversas, algunas de estas eran nuevas para nosotros.

Se trata de un proyecto Laravel, un framework de PHP con estructuración MVC del código de back-end. Este proyecto abarca el desarrollo de una aplicación web principal, que sería el sitio web de gestión de las citas y pruebas medicas y dos APIs internas que se encarguan de la búsqueda asíncrona de los médicos y pacientes registrados en el sitio web. Para la parte cliente, hemos trabajado con blade en html para la estructuración de las vistas y la comunicación entre el front-end y el back-end, la parte lógica se ha hecho mediante el uso de componentes del framework vue 3. En cuanto a la persistencia de datos hemos usado una base de datos Postgres.

Chapter 2. Diseño de base de datos y diagramas de clases.

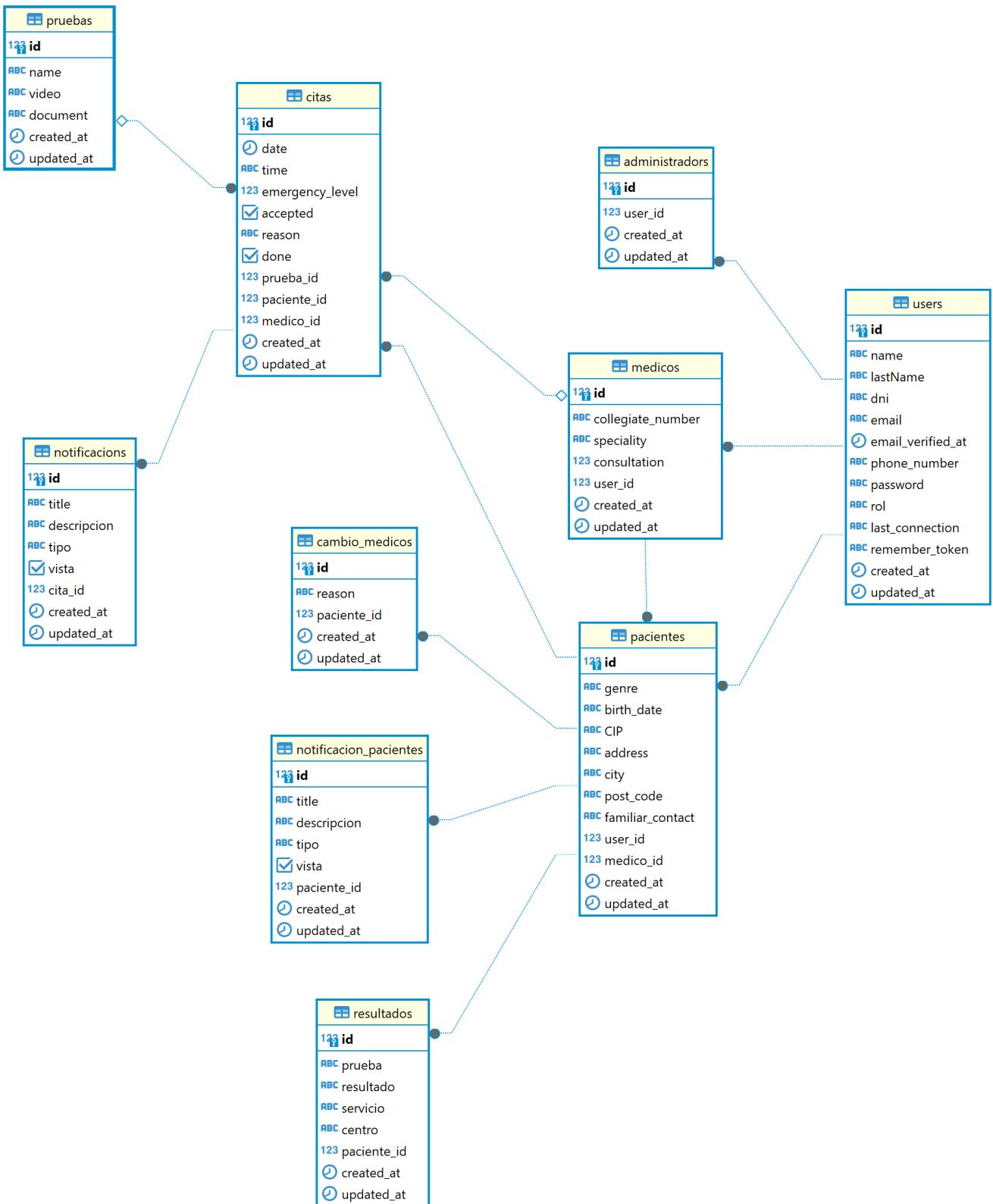


Imagen 1. Modelo MER base de datos

El diagrama de clases de modelos de la aplicación puede dirigirse al siguiente enlace: [Diagrama de clases de modelos](#)

Chapter 3. Diseño y guía de estilos

Tanto para el diseño de los mockups de las diferentes vistas como para la guía de estilos hemos usado FIGMA, encontraréis los enlaces en cada uno de los apartados.

3.1. Sketching y Mockups de interfaces de usuario.

Consejo:

Para ver los detalles de los Sketching y Mockups, puede dirigirse al siguiente enlace [Sketching y Mockups](#)

3.2. Guia de estilos.

En este apartado explicaremos brevemente la toma de decisiones a la hora de definir cada punto de la guía de estilos. Para poder acceder a la guía de estilos en FIGMA bastará con hacer click en el enlace que dejamos justo al terminar de explicar estos puntos.

1. **Paleta de colores.** Nuestra aplicación al estar orientada al personal que trabaja en el sector de la medicina, hemos decidido utilizar tonos azulados y verdosos como color de marca sobre un fondo blanco. Además el color azul refleja confianza y estabilidad, que es justo lo que necesitan los usuarios que hagan uso de una aplicación de este estilo.
2. **Tipografía.** Como paleta de colores, tal y como está especificado en la guía de estilos, hemos usado Outfit, ya que es una tipografía elegante a la par de sencilla, y queda bien tanto como para los títulos o la letra de los botones.
3. **Iconografía.** En cuanto a iconos hemos decidido optar por usar los de la librería font-awesome 5, que nos parecen quedan mejor con nuestro estilo y es gratis.
4. **Espacios y formas.** Hemos decidido usar medidas a partir del tamaño de la fuente (rem), ya que al cambiar de pantalla, cambiamos de tamaño de fuente y en teoría todo debería de quedar proporcionado. Buscamos cual era el espacio estándar entre el contenido y los bordes en pantallas móviles y a partir de ahí lo escalamos según la pantalla.
5. **Botones.** Los botones hemos decidido que usarían la misma fuente que el texto normal del sitio web. Tendrán un borde del color de la 'marca' de la web y un fondo azul que se rellenará al hacer hover con un color azulado más oscuro.

Chapter 4. Manual de instalación, distribución y configuración.

En esta sección se explicarán detalladamente los pasos a seguir para realizar el despliegue de la aplicación y extensiones necesarias para su funcionamiento. En primer lugar explicaremos como hacer el despliegue en local, y después en servidor, en este caso usaremos una **instancia EC2 de AWS (Amazon Web Services)** que hará la función de servidor.



Atención, ambos manuales explican el despliegue en una máquina con un sistema que utilice el shell *bash*. Para el despliegue en una máquina con Windows los pasos a seguir son los mismos pero la manera de instalar los diferentes *softwares* que necesitaremos es distinta.

4.1. Despliegue en local

Para realizar el despliegue del proyecto en un entorno local no necesitamos ningun servidor puesto que usaremos el que lleva "built-in" artisan para servir nuestro sitio web.

Suponiendo que la máquina en la que hagamos la instalación tiene lo básico instalado y configurado como el php, composer, nodeJS, npm y algún editor de código fuente como *Visual Studio Code (VSCode)*, los pasos a seguir son los siguientes para un sistema operativo windows:

4.1.1. Preparación del entorno

Crearemos un nuevo directorio en donde alojaremos la aplicación; Abrimos el *Visual Studio Code* y nos dirigimos al directorio creado anteriormente para clonar el repositorio donde se encuentra la aplicación.

Abrimos una nueva terminal en el *VSCode*, para

Ctrl + ñ

4.1.2. Clonar el Repositorio GIT

El siguiente paso es descargar todo el proyecto desde el repositorio de GIT. Para hacer esto, basta con ir al directorio de la máquina en el que se quiera instalar el proyecto y ejecutar el siguiente comando.

```
git clone https://git.copernic.cat/garcia.dominguez.samuel/blanxart-m12-amador-garcia-rojas-mateos.git .
```



El punto del final del comando significa en la ruta actual. Es decir en el directorio en el que nos encontramos en el momento de ejecutarlo. Si se quiere se puede cambiar por una ruta física o por una relativa.

4.1.3. Copiar y configurar el archivo .env del proyecto

Despues de realizar el cloando de la aplicación desde el repositorio a nuestra maquina local, empezaremos con la configuración local. Pese a que hemos dedicado un apartado entero más adelante para la explicación del archivo `.env`, en este apartado hablaremos de él y explicaremos brevemente qué se ha de hacer para que funcione la aplicación.

El siguiente paso consiste en localizar el archivo `.env` de la aplicación web, para ello nos hemos de ubicar en la carpeta raíz del repositorio, es decir allá donde lo hayamos clonado.



El archivo `.env` en Laravel es un archivo de configuración que contiene variables de entorno para ajustar la configuración del proyecto. Debes configurar debidamente este archivo para que tu aplicación funcione correctamente.

```
cd /ruta/a/tu/proyecto
```

Desde este punto hemos de entrar en la siguiente ruta: `blanxart/`, que sería la carpeta raíz del proyecto Laravel y allí localizar el archivo `.env.example` y copiarlo en el mismo lugar pero con el nombre `.env`, de esta manera crearemos el archivo de variables de entorno de nuestra aplicación a partir de una plantilla preconfigurada guardada en el repositorio GIT.

```
cd blanxart
copy .env.example .env
```

Una vez hecho esto hemos de abrir el nuevo archivo `.env` desde el `VSCode`, dando doble click al archivo recien creado.

Una vez abierto el archivo buscamos hasta encontrar estas líneas:

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=laravel
DB_USERNAME=root
DB_PASSWORD=
```

Y las cambiamos a:

```
DB_CONNECTION=pgsql
DB_HOST=127.0.0.1
DB_PORT=5432
DB_DATABASE=blanxart
DB_USERNAME=usuario
DB_PASSWORD=1234
```

Guardamos los cambios con "Ctrl + s" y cerramos nano.

Con esto habremos configurado lo necesario para que al levantar el contenedor de docker donde está nuestra base de datos, la aplicación conecte con ésta.

4.1.4. Instalación de dependencias del proyecto Laravel

Una vez configurado el archivo `.env`, el siguiente paso es instalar todas las dependencias necesarias para el proyecto mediante el gestor Composer.

Ubicándonos de nuevo en la carpeta `fairy_tickets/`, la carpeta raíz del proyecto Laravel, en la que deberíamos de encontrarnos, si se ha seguido la guía hasta este punto, lanzamos el siguiente comando:

```
composer update
```

Este comando instalará y/o actualizarán todas las dependencias especificadas en el archivo `composer.json`, que son las que necesita nuestro proyecto.

Llegados aquí, en cuanto a la parte web del proyecto sólo nos quedaría generar una `APP_KEY` de Laravel para poder funcionar, Para esto, seguimos en la carpeta raíz del proyecto Laravel `blanxart/` y lanzamos el siguiente comando:

```
php artisan key:generate
```

4.1.5. Configuración de nodeJS

Ubicados en el directorio de la aplicación, debemos instalar el gestor de paquetes de node (npm) dentro del proyecto, para ello ejecutamos el comando:

```
npm install
```

Por último debemos construir los componentes vue en el servidor, esto con el fin de optimizar el código JS para ser usado en el ambiente de producción, para ello ejecutamos el comando:

```
npm run build
```

4.1.6. Configuración del docker y la base de datos

En el siguiente paso, explicaremos cómo crear la imagen de docker necesaria, donde montaremos nuestra base de datos PostgreSQL y posteriormente levantar el contenedor docker para que nuestra aplicación se pueda conectar a la base de datos.

La base de datos será creada desde un script, en el momento de crear el contenedor y lanzarlo por primera vez.

Así, Lo primero que hemos de hacer es localizar la carpeta *docker* en el proyecto. Desde la raíz del repositorio GIT, la ruta es [~/docker-config/](#). Dentro encontraremos el fichero: **compose.yml**.

Nos colocamos en línea de comandos en esa carpeta y montamos la imagen del dockerfile con el comando de docker: build:

```
cd docker  
docker compose build
```

Una vez creada la imagen tendremos que lanzar el comando up para lanzar los contenedores indicados en el archivo **compose.yml**:

```
docker compose up -d
```

De esta manera, ya tendremos el contenedor docker de nuestra base de datos postgres en funcionamiento.



Atención, el archivo **compose.yml** está configurado para levantar el contenedor de Postgres y conectarlo al puerto 5432 del host, si este ya está en uso se tendrá que cambiar el numero de la izquierda de la siguiente linea:

```
ports:  
- 5432:5432
```

Para terminar este paso, hemos de llenar la base de datos con algunos datos iniciales y generar las tablas que necesitará nuestra aplicación. Para ello nos colocamos, de nuevo, en la carpeta raíz del proyecto Laravel: [~/blanxart](#) y lanzamos los comandos:

```
php artisan migrate:fresh  
php artisan db:seed
```

Al finalizar la ejecución de los *seeders* y poblar las tablas de la base de datos, se generará un token, este token debe ingresarse en la variable de configuración "**API_KEY**" del fichero *.env* de la aplicación

Token: 1|VD8x2HAiBTUx7ltZRI9TYJW3S5D3LKHsbbAWvhDp1f112b49

```
VITE_PUSHER_APP_KEY="${PUSHER_APP_KEY}"  
VITE_PUSHER_HOST="${PUSHER_HOST}"  
VITE_PUSHER_PORT="${PUSHER_PORT}"  
VITE_PUSHER_SCHEME="${PUSHER_SCHEME}"  
VITE_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"
```

```
API_KEY = VD8x2HAiBTUx7ltZRI9TYJW3S5D3LKHsbbAWvhDp1f112b49
```

4.1.7. Puesta en marcha de la web

Para comprobar que todo funciona bien y empezar a usar la web en local puedes iniciar el servidor de desarrollo de Laravel utilizando el comando `php artisan serve`:

```
php artisan serve
```

Esto iniciará un servidor de desarrollo en <http://localhost:8000>, donde podrás acceder a la aplicación.



Atención, debemos asegurarnos de haber ejecutado el comando "npm run build" antes de lanzar el servidor de laravel con el comando anteriormente visto.

4.2. Instancia EC2 AWS (Amazon Web Services)

En esta sección explicaremos cómo instalar la aplicación en servidor, en este caso usaremos una instancia EC2 de AWS, generalmente estos servicios son de cobro, sin embargo se usa un laboratorio de pruebas, concedido por el centro, con un saldo para realizar pruebas de \$100 USD. Esta instancia contará con un sistema operativo **Ubuntu Server 22.04**, disponible para la capa gratuita del AWS. en cualquier caso el proceso será similar en cualquier máquina de la familia Linux.

Primero deberíamos de instalar Apache2 y PHP en el servidor Debian ejecutando los siguientes comandos:

```
sudo apt update
```

```
sudo apt install apache2 postgresql postgresql-contrib php php-curl php-bcmath php-json php-pgsql php-mbstring php-xml php-tokenizer php-zip composer git
```

Solicitará confirmación, escribimos "Y" y pulsamos la tecla "enter".

Una vez finalizada la instalación anterior, comprobamos la versión y el estado del servicio del servidor web (apache2):

```
sudo systemctl is-enabled apache2
sudo systemctl status apache2
```

Igualmente con el servicio de la base de datos (postgresql):

```
sudo systemctl is-enabled postgresql
sudo systemctl status postgresql
```

Tambien verificamos la versión de PHP y Composer:

```
php -v  
sudo -u www-data composer -v
```

4.2.1. Instalación de NodeJS y NPM

Ya que la aplicación cuenta con componentes de interfaces de usuario, desarrollados en con Vue, se requiere realizar su debida instalación. Dicha instalación se realizará con NVM (Node Version Manager), este software nos permite instalar cualquier versión de NodeJS que se necesite para el proyecto.

instalamos NVM en el sistema:

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.7/install.sh | bash
```

Debemos cerrar la terminal actual y volver a abrir una nueva para que los cambios sean tomados; Una vez realizado esto, descargamos e instalamos NodeJS:

```
nvm install 20
```

Verificamos las versiones de Node y de NPM instaladas en el sistema:

```
node -v  
npm -v
```

4.2.2. Configuración de PHP

Antes de ejecutar Laravel en el sistema, debemos habilitar algunas extensiones de PHP para que funcione correctamente.

Usando el editor nano, ejecutamos la siguiente orden:

```
sudo nano /etc/php/8.3/apache2/php.ini
```

Dentro del fichero *php.ini* descomentamos las siguientes extensiones:

```
extension= FileInfo  
extension= Mbstring  
extension= OpenSSL  
extension= Pdo_Pgsql  
extension= Pgsql
```

guardamos los cambios realizados en el fichero "Ctrl+o", presionamos "enter" para sobre escribir el nombre del fichero y salimos de este "Ctrl+x".

4.2.3. Configuración de postgresql

Crearemos el usuario y le asignaremos una contraseña con la cual la aplicación se conectará a la base de datos, para ello ejecutaremos el siguiente comando:

```
sudo -u postgres createuser --interactive
```

El sistema nos solicitará el nombre del nuevo usuario y una confirmación de si este usuario será superusuario a lo cual diremos que si.

```
Enter name of role to add: usuario  
Shall the new role be a superuser? (y/n): y
```

```
ALTER USER usuario PASSWORD '1234';
```

Por último, crearemos la base de datos sobre la cual trabajará la aplicación, primero debemos cambiar a la cuenta de postgres dentro del servidor.

```
sudo -i -u postgres
```

Estando ya en la cuenta de postgres creamos la base de datos, ejecuntando el siguiente comando:

```
postgres@server:~$ createdb blanxart
```

Por ultimo para verificar que la base de datos se haya creado correctamente, ingresamos al indicador de postgres y listamos las base de datos existentes.

```
$ psql  
postgres=# \l
```

para volver al usuario del servidor usamos el comando "exit".

4.2.4. Clonar repositorio GIT

Realizamos un cambio de directorio en donde generalmente se almacenan los proyectos en el servidor.

Ejecutamos el siguiente comando:

```
cd /var/www/
```

clonamos el proyecto desde el repositorio en donde se encuentra el proyecto guardado, por lo general se clona la rama *main*.

```
sudo git clone https://git.copernic.cat/garcia.dominguez.samuel/blanxart-m12-amador-garcia-rojas-mateos.git
```

a continuación GIT solicitará las credenciales para validar la acción sobre el repositorio, se deben ingresar para que se realice correctamente el proceso de clonado. Se creará una carpeta con el nombre del repositorio.

validamos que se haya creado con el siguiente comando:

```
ls -l
```

nos ubicamos dentro del directorio del proyecto, esta se encuentra dentro del directorio del repositorio clonado.

```
cd blanxart-m12-amador-garcia-rojas-mateos/blanxart/
```

Una vez dentro del directorio del proyecto, ejecutamos el siguiente comando:

```
sudo composer update
```

Despues de actualizar y descargar las dependencias necesarias para la ejecución del proyecto, creamos el fichero .env a partir del fichero .env.example e ingresamos a este con el editor nano.

```
sudo cp .env.example .env  
sudo nano .env
```

Dentro del fichero .env, verificamos que las variables de conexión a la base de datos sean las correctas.

```
DB_CONNECTION=pgsql  
DB_HOST=127.0.0.1  
DB_PORT=5432  
DB_DATABASE=blanxart  
DB_USERNAME=usuario  
DB_PASSWORD=1234
```

Despues generamos la clave de la aplicación.

```
sudo php artisan key:generate
```

En este punto, podemos probar la conexión entre la base de datos y la aplicación clonada siguiendo dentro del directorio del proyecto, ejecutamos el siguiente comando:

```
php artisan migrate:fresh
```

Sí todo se ha ejecutado con normalidad, se crearan las bases de datos según las migraciones existentes en la aplicación.

4.2.5. Configuración de nodeJS

Ubicados en el directorio de la aplicación, debemos instalar el gestor de paquetes de node (npm) dentro del proyecto, para ello ejecutamos el comando:

```
npm install
```

Por último debemos construir los componentes vue en el servidor, esto con el fin de optimizar el código JS para ser usado en el ambiente de producción, para ello ejecutamos el comando:

```
npm run build
```

4.2.6. Configuración de Apache2

En esta sección configuraremos el servicio del servidor web Apache2 y crearemos el virtual host de nuestra aplicación.

Habilitamos el modulo *rewrite* de apache.

```
sudo a2enmod rewrite
```

Creamos el nuevo virtual host para nuestra aplicación en la ruta '**/etc/apache2/sites-available/blanxart.conf**', con ayuda del editor nano.

```
sudo nano /etc/apache2/sites-available/blanxart.conf
```

dentro del nuevo fichero creado, agregamos la siguiente configuración, cambiando el campos **ServerName** con el dominio de la aplicación, en este caso con la ip publica proporcionada por AWS. Tambien cambiamos la ruta de la etiqueta "**Directory**", con la ruta donde se encuentra alojado la aplicación "**/var/www/blanxart-m12-amador-garcia-rojas-mateos**".

Tambien la etiqueta "**DocumentRoot**" con la ruta del directorio *public* de la aplicación "**/var/www/blanxart-m12-amador-garcia-rojas-mateos/blanxart/public**".

```

<VirtualHost *:80>

    ServerAdmin admin@hwdomain.io
    ServerName 52.23.235.5
    DocumentRoot /var/www/blanxart-m12-amador-garcia-rojas-mateos/blanxart/public

    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/blanxart-m12-amador-garcia-rojas-mateos>
        AllowOverride All
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>

```

Guardamos los cambios con "Ctrl+o", presionamos "enter" para sobrescribir el nombre del fichero y salimos del edito nano con "Ctrl+x".

Ahora activamos la configuración del virtual host creado y verificamos que la sintaxis de apache, si no existen errores la terminal nos muestra el mensaje "**Syntax OK**", ejecuntando los siguientes comandos:

```

sudo a2ensite blanxart.conf
sudo apachectl configtest

```

Por ultimo reiniciamos el servicio del servidor web Apache para aplicar los cambios y configuraciones realizadas.

```

sudo systemctl restart apache2

```

4.2.7. Configuración final

En este punto la instalación de la aplicación esta realizada y podremos acceder a la aplicación mediante el fichero "**hosts**", para ello modificaremos el fichero de la ruta "**/etc/hosts**" con privilegios root.

```

sudo nano /etc/hosts

```

agregamos la ip pública proporcionada por AWS seguido del dominio de nuestra aplicación, si no se cuenta con un dominio, ingresamos únicamente la ip.

```
{ip publica} dominio
```

Guardamos cambios con "Ctrl+o", presionamos "enter" para sobrescribir el nombre del fichero y salimos del editor nano con "Ctrl+x".

Tambien debemos cambiar el propietario del directorio de la aplicación, esto con el fin de que la aplicación sea accesible desde internet.

```
sudo chown -R www-data:www-data /var/www/blanxart-m12-amador-garcia-rojas-mateos/blanxart/
```

Por último, estando ubicados en el directorio de la aplicación ejecutamos los *seeders* y asi poder realizar un smoke test de nuestra aplicación y comprobar que funcione correctamente.

```
php artisan db:seed
```

Al finalizar la ejecución de los *seeders* y poblar las tablas de la base de datos, se generará un token, este token debe ingresarse en la variable de configuración "**API_KEY**" del fichero *.env* de la aplicación

```
Token: 1|VD8x2HAiBTUx71tZRI9TYJW3S5D3LKHsbbAWvhDp1f112b49
```

```
sudo nano .env
```

```
VITE_PUSHER_APP_KEY="${PUSHER_APP_KEY}"
VITE_PUSHER_HOST="${PUSHER_HOST}"
VITE_PUSHER_PORT="${PUSHER_PORT}"
VITE_PUSHER_SCHEME="${PUSHER_SCHEME}"
VITE_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"
```

```
API_KEY = VD8x2HAiBTUx71tZRI9TYJW3S5D3LKHsbbAWvhDp1f112b49
```

Guardamos cambios con "Ctrl+o", presionamos "enter" para sobrescribir el nombre del fichero y salimos del editor nano con "Ctrl+x".

Chapter 5. Manual de usuario

En este apartado se detallará cada apartado de la página web y sus respectivas funcionalidades para dar a conocer de forma detallada y sencilla la estructura de la web Sanitech.

Sanitech es un espacio de salud digital para nuestros usuarios la cual tiene como finalidad cubrir las siguientes necesidades:

- Acceder a información de preparación de pruebas médicas.
- Gestionar citas complementarias y visitas.
- Ofrecer un sistema de recordatorios de las citas.

Cuando accedemos por primera vez a la página web, seremos redirigidos al login, donde podemos introducir nuestro DNI y contraseña para acceder a la web. Dependiendo del rol que tengamos, podemos registrarnos como paciente, médico o administrador.

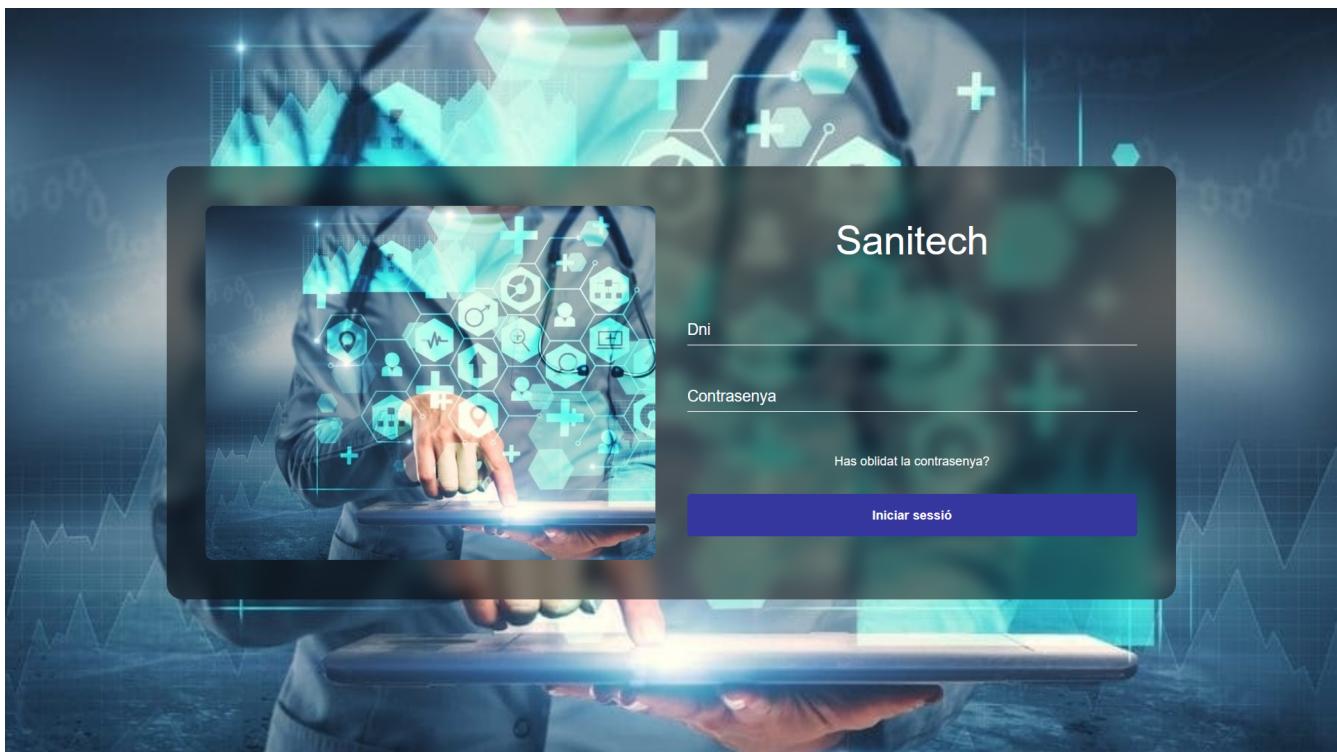


Imagen 2. Vista Login.

5.1. Usuario Paciente

Como usuario paciente, podemos hacer lo siguiente:

- Acceder a los videos y documentos explicativos de las pruebas.
- Consultar los resultados de las pruebas realizadas.
- Consultar pruebas y visitas pendientes.
- Pedir justificantes de una visita médica.
- Recibir notificaciones de recordatorio de las visitas asignadas con fecha y hora, con un tiempo de antelación.

- Rechazar citas asignadas.

5.1.1. Home

Una vez registrados como paciente, seremos redirigidos a la página principal del sitio web. Podemos ver un header de color azul oscuro con un menú donde están los 4 accesos, permitiéndonos ingresar de forma más rápida.

En el centro podemos ver una bienvenida con nuestro nombre y 4 bloques: (1) Agenda, (2) Informes y resultados, (3) Solicitudes y (4) Notificaciones. Podemos ver en el bloque de notificaciones que tiene un badge notification de color rojo con el número total de notificaciones nuevas no leidas por el usuario.

1. Vista home paciente. image::home_paciente.png[Home paciente]

Si cerramos sesión y volvemos a ingresar, podemos ver la fecha y hora de nuestra última conexión en la web en la parte superior, debajo de la bienvenida.

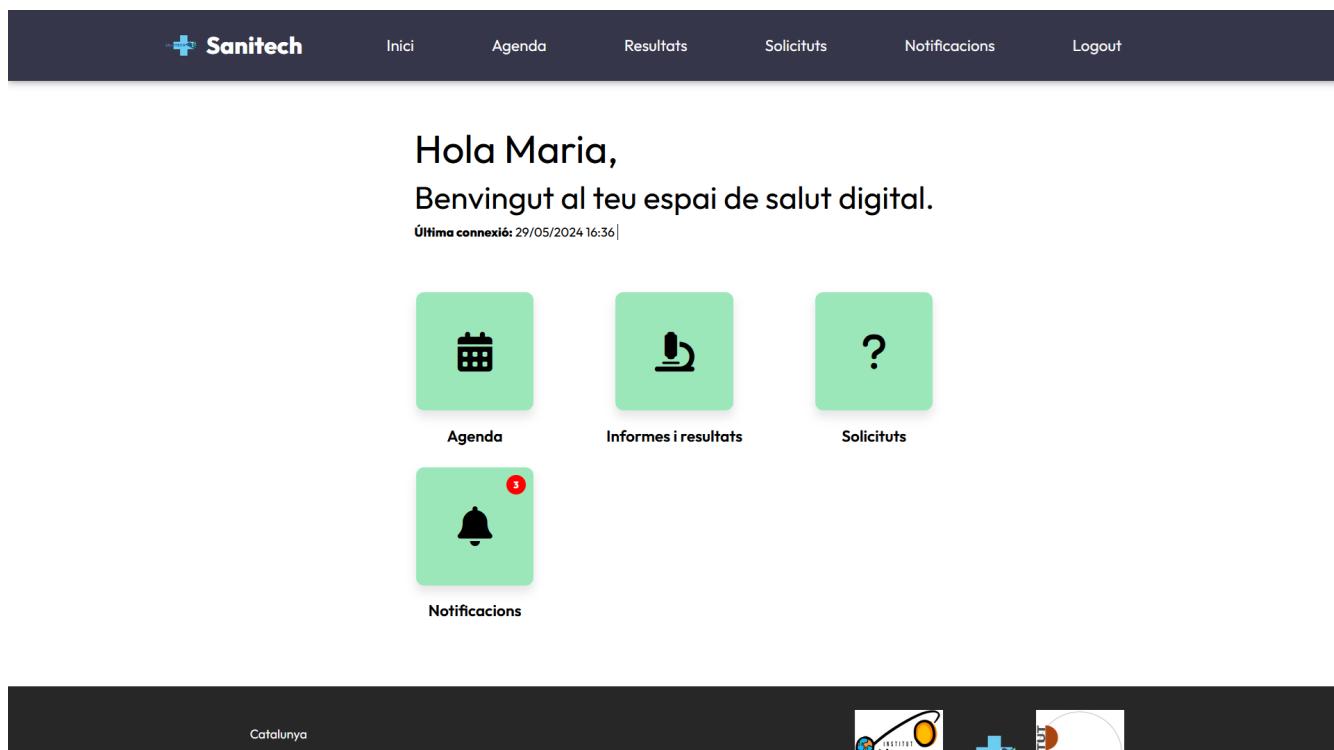


Imagen 3. Vista última conexión.

5.1.2. Agenda

En este apartado podemos ver todas nuestras citas complementarias y visitas, las cuales podemos filtrar por 3 tipos: (1) citas realizadas (aquellas que ya se han hecho), (2) citas no realizadas (aquellas que no hemos hecho pero tenemos una fecha programada) y (3) citas pendientes por asignar (aquellas que ni hemos hecho ni tienen una fecha programada).

Las citas complementarias tienen asignado un documento pdf y un video con la explicación de la prueba.

The screenshot shows the 'Agenda' (Appointment) section of the Sanitech application. At the top, there's a navigation bar with links for 'Inici' (Home), 'Agenda', 'Resultats', 'Solicitudes', 'Notificaciones', and 'Logout'. Below the navigation, a back button '← Tornar' (Back) is visible. The main title 'Agenda' is centered above a message: 'Aquí podrás visualizar todas las citas.' (Here you can view all your appointments). A dropdown menu labeled 'tipus de cita' (Appointment type) is open, showing 'Citas realizadas' (Completed appointments). Below this, a table titled 'Citas trobades: 1' (Found appointments: 1) lists one appointment: 'Visita' (Visit) on '2024-05-16'. The table has columns for 'Nom de la prova' (Test name), 'Adjunts' (Attachments), and 'Data' (Date). There are icons for PDF and video attachments. Navigation arrows < 1 > are at the bottom of the table. The footer features the 'Catalunya' logo and other institutional logos.

Imagen 4. Vista agenda paciente.

5.1.3. Informes

En el apartado de informes y resultados veremos un bloque con unas tarjetas desplegables las cuales contienen la información del resultado de la prueba realizada, como el nombre y la hora de la publicación, información del resultado, el servicio y el centro donde se ha realizado.

The screenshot shows the 'Informes clínicos' (Clinical Reports) section of the Sanitech application. The top navigation bar is identical to the previous screenshot. The main title 'Informes clínicos' is centered above a message: 'Aquí puedes visualizar todos los resultados.' (Here you can view all results). A table displays two report cards: 'Rectoscopia' (Gastroscopy) and 'Espirometria' (Pulmonary Function Test). The 'Espirometria' card is expanded, showing the following details:

- Resultat:** Els resultats mostren una disminució en el flux de l'aire durant l'expiració, el que suggereix una obstrucció en les vies respiratòries. Això pot indicar condicions com asma, bronquitis crònica o la malaltia pulmonar obstructiva crònica (EPOC).
- Servei:** Pneumologia
- Centre:** CAP Oest Terrassa

Imagen 5. Vista informes y resultados.

5.1.4. Solicitudes

En solicitudes, tendremos acceso a 3 bloques, podremos cambiar de médico, pedir un justificante de una cita realizada o pedir una cita de tipo visita a nuestro médico asignado.

The screenshot shows the 'Solicitudes' (Requests) section of the Sanitech app. At the top, there is a navigation bar with the Sanitech logo and links for 'Inici' (Home), 'Agenda', 'Resultats', 'Solicitudes', 'Notificacions', and 'Logout'. Below the navigation bar, there is a back button labeled '← Tornar' (Back). The main title 'Solicitudes' is displayed in bold, followed by the subtitle 'Fes consultes, gestions o demana una cita.' (Make consultations, manage or request an appointment.). There are three main buttons: 'Canviar metge' (Change doctor) with a person icon, 'Justificant' (Justification) with a document icon, and 'Demana cita' (Request appointment) with a calendar icon. At the bottom of the screen, there is a footer bar with the text 'Catalunya' and '© Sanitech 2021. All rights reserved.', along with logos for 'INSTITUT AUTONÒMIC DE SALUT DE CATALUNYA', 'SANITECH', and 'HUT'.

Imagen 6. Vista solicitudes.

5.1.5. Cambio de médico

Si queremos cambiar de médico, únicamente tenemos que rellenar el campo "motiu del canvi de metge" especificando por qué queremos cambiar de médico, y darle al botón enviar.

← Tornar

Canvi de metge

Redacti el motiu pel qual vol canviar de metge

Les meves dades	
Nom	Maria López Garcia
CIP	LOGA 1940315 00 3
Codi	08192

Motiu del canvi de metge

Motiu del canvi de metge...

Imagen 7. Vista cambio de medico.

5.1.6. Solicitar justificante

En este apartado podemos ver una tabla donde se listan todas las citas realizadas. Si queremos generar un justificante de una de estas citas, solo tenemos que dar click al icono del pdf y automáticamente se descargará nuestro justificante.

← Tornar

Demantar justificant

Genera un justificant de les teves cites realitzades.

Cites trobades: 1

Nom	Data	Generar justificant
Visita	2024-05-16	

< 1 >

Catalunya

Imagen 8. Vista solicitud de justificante.

5.1.7. Pedir cita al médico

Al pedir una cita con nuestro médico, podemos ver en la parte superior una previsualización de información nuestra general. Para pedir cita, tenemos que escoger una fecha disponible y una hora disponible para ese dia. En caso que no exista una fecha disponible aparecerá una notificación de alerta diciendo que el dia seleccionado no tiene horas disponibles para escoger.

← Tornar

Cita amb el metge

Soliciti una cita amb el metge

Les meves dades	
Nom	Maria López Garcia
CIP	LOGA 1940315 00 3
Codi	08192

1. Seleccioni la data de cita:

dd / mm /aaaa

2. Seleccioni la hora de cita:

Selecciona una hora

3. Motiu de la visita:

Imagen 9. Vista solicitud de cita como paciente.

Finalmente tenemos que llenar el campo del motivo por el cual queremos hacer la visita.

3. Motiu de la visita:

Enviar



Imagen 10. Vista formulario de solicitud de cita.

5.1.8. Notificaciones

En el apartado de notificaciones se pueden ver unas tarjetas con información. Existen distintos tipos de notificaciones: (1) notificaciones de recordatorio, (2) de cambio de médico y (3) para aceptar o rechazar una cita.

The screenshot shows the 'Notificaciones' (Notifications) screen of the Sanitech app. At the top, there is a header with the Sanitech logo and links for 'Inici' (Home), 'Agenda', 'Resultats', 'Solicitudes', 'Notificaciones', and 'Logout'. Below the header, there is a back button labeled 'Tornar' (Return). The main area is titled 'Notificaciones' and contains the following notifications:

- Cambio medico**: 'medico cambiado correctamente' (Physician changed correctly) at 12:54.
- Recordatori cita Medica**: 'Recorda que tens una cita programada per el dia 16/5 a les 10:00' (Remember you have an appointment scheduled for May 16 at 10:00) at 12:54.
- Cita Espirometria**: 'Té un cita programada per el dia 30/5 a les 12:00, podrà assistir?' (You have an appointment scheduled for May 30 at 12:00, will you be able to attend?) at 12:54. This notification has a blue button labeled 'Resposta' (Response).



Imagen 11. Vista notificaciones de recordatorio.

Para aceptar o rechazar una cita, solo bastará con dar click al botón respuesta y aparecerá una ventana emergente para aceptar, rechazar la cita o cancelar la acción.

This screenshot shows a dark-themed version of the 'Notificaciones' (Notifications) screen with an additional modal dialog for accepting an appointment:

Recordatori cita Medica: 'Recorda que tens una cita programada per el dia 16/5 a les 10:00' (Remember you have an appointment scheduled for May 16 at 10:00) at 12:54. A modal dialog box is overlaid on the screen with the question 'Podrà assistir a aquesta cita?' (Will you be able to attend this appointment?). It contains two buttons: a green 'Sí' (Yes) button and a red 'No' (No) button. There is also a 'Tancar' (Close) button in the bottom right corner of the modal.

Imagen 12. Vista notificación de aceptación de citas agendadas.

5.2. Usuario Médico

Como usuario médico, podemos hacer lo siguiente:

- Crear una cita.
- Buscar a nuestros pacientes asignados por nombre.
- Consultar la agenda de los pacientes que tengamos asignados.
- Consultar los resultados de las pruebas de nuestros pacientes.

5.2.1. Home

Al iniciar sesión como médico, tendremos acceso a 3 opciones: (1) crear una cita, (2) ver los resultados y pruebas de los pacientes y (3) consultar sus agendas.

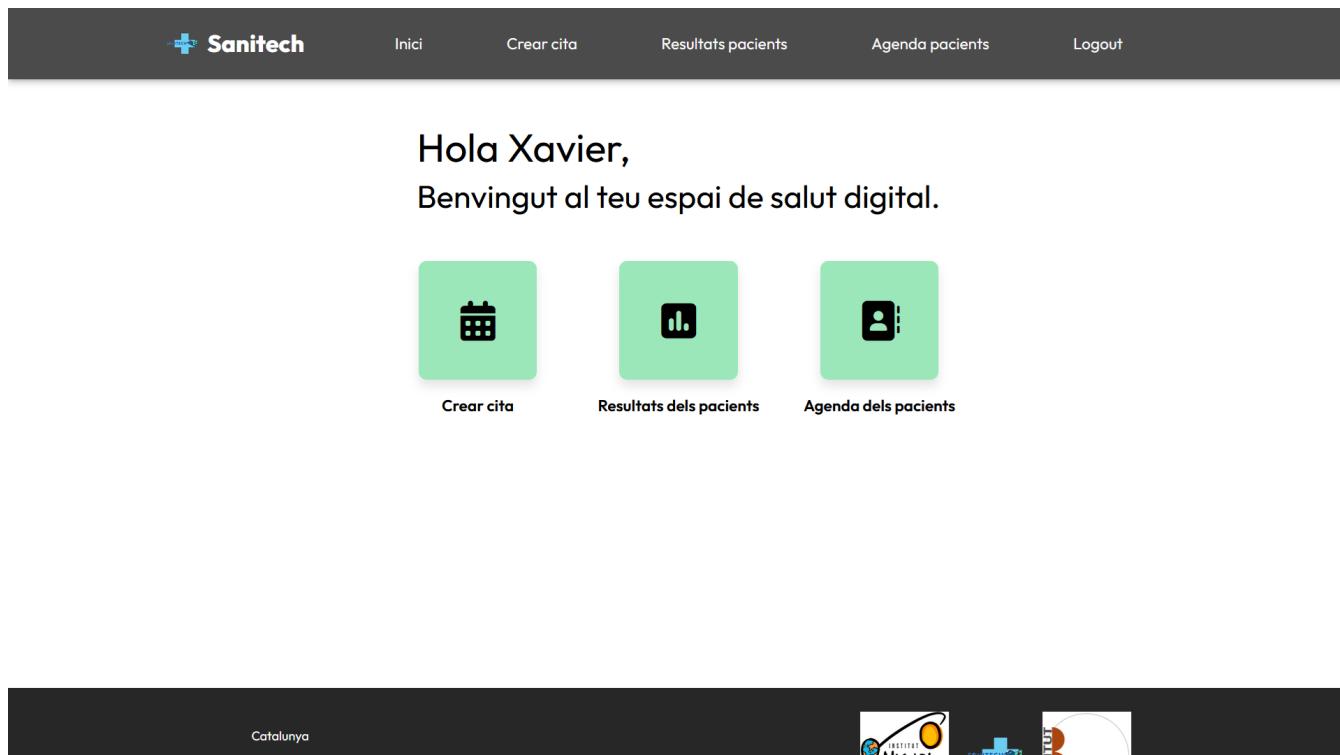


Imagen 13. Vista home medico.

5.2.2. Buscador de pacientes

El buscador de pacientes es una funcionalidad que sirve para facilitar la búsqueda de los resultados de las pruebas o las citas de nuestros pacientes. Es una pantalla intermedia para cada apartado mencionado anteriormente.

[← Tornar](#)

Cercador pacients

Selecciona el pacient al qual li assignarà la cita

Juan Martínez Pérez



CIF: MAPE 0 790822 01 6

DNI: 71985632T

Telef: 622 33 44 55

Carlos Carreón



CIF: A123456789123

DNI: 28510831P

Telef: 905061964

Maria López Garcia

CIF: LOPA1234567890007

Imagen 14. Vista buscador de pacientes.

5.2.3. Crear cita

Como médico, podremos crear una cita a nuestro paciente una vez seleccionado en el buscador de pacientes. En esta pantalla podemos ver en la parte superior un bloque con la información relevante de nuestro paciente, como el DNI, el nombre y su número de teléfono.


 Sanitech
 [Inici](#)
[Crear cita](#)
[Resultats pacients](#)
[Agenda pacients](#)
[Logout](#)

Crear cita

Solicita una cita amb el metge

Dades del pacient	
Nom	Juan Martínez Pérez
DNI	71985632T
Telèfon	622 33 44 55

La cita tindrà una prova?

Seleccioneu el nivell d'emergència:

Nivell 1 (2-3 setmanas) Nivell 2 (3-4 setmanas) Nivell 3 (2-3 mesos) Nivell 4 (3 a 6 meses) Nivell 5 (més de 6 meses)

Crear Cita

Imagen 15. Vista crear cita como médico.

Primero que nada, tendremos que especificar si la cita tendrá una prueba. De ser así, aparecerá un selector donde podemos seleccionar el tipo de prueba. Después tenemos que seleccionar el nivel de

emergència de la prueba, la cual determinará el tiempo máximo que se pueda realizar la cita : nivel 1 (2 - 3 semanas), nivel 2 (3 - 4 semanas), nivel 3 (2 - 3 meses), nivel 4 (3 a 6 meses) y nivel 5 (mas de 6 meses).

The screenshot shows a web-based application with a dark header bar containing the logo 'Sanitech' and navigation links: 'Inici', 'Crear cita', 'Resultats pacients', 'Agenda pacients', and 'Logout'. Below the header is a large input field. Underneath it, there is a question 'La cita tindrà una prova?' followed by a blue toggle switch. A section titled 'Selecciona la prova per a la cita:' contains a dropdown menu with the placeholder 'Escull una prova...' and a list of options: 'Escull una prova...', 'Espirometria', 'Rectoscopia', 'Tac abdominal', and 'Broncoscopia'. At the bottom right of the page is a blue button labeled 'Crear Cita'.

Imagen 16. Vista formulario crear cita como médico.

5.2.4. Informes clínicos del médico

En el apartado de informes clínicos, al igual que el paciente, el médico también puede consultar los resultados de las pruebas del paciente que haya seleccionado. La información que aparece es la misma: el nombre, la fecha de publicación y, al desplegar la tarjeta, el resultado, servicio y centro donde se ha realizado la prueba.

The screenshot shows a medical reporting interface. At the top, there's a header with the Sanitech logo and links for 'Inici', 'Crear cita', 'Resultats pacients', 'Agenda pacients', and 'Logout'. Below this, a 'Tornar' (Return) button is visible. The main section is titled 'Informes clínics' (Clinical Reports) with the sub-instruction 'Aquí pots visualitzar tots els resultats.' (Here you can view all results.). A list of reports is displayed:

- Tac abdominal** (12:54): Includes a detailed description of abdominal findings.
- Rectoscopia** (12:54): An arrow icon indicates more details are available.
- Espirometria** (12:54): Another arrow icon indicates more details are available.

Imagen 17. Vista informes clínicos como médico.

5.2.5. Agenda del médico

Además de los resultados, el médico también puede consultar la agenda de sus pacientes, cuya funcionalidad es exactamente la misma que la agenda del mismo paciente. También podrá consultar los documentos y videos adjuntos.

The screenshot shows a patient agenda interface. The top navigation bar includes the Sanitech logo and links for 'Inici', 'Crear cita', 'Resultats pacients', 'Agenda pacients', and 'Logout'. A 'Tornar' (Return) button is present. The main area is titled 'Agenda' (Agenda) with the instruction 'Aquí podrás visualitzar totes les cites.' (Here you can view all appointments.). A dropdown menu for 'tipus de cita' (Appointment type) is open, showing 'Cites realitzades'. Below this, a table lists a single appointment:

Nom de la prova	Adjunts	Data
Visita	PDF	2024-05-15

Pagination controls (1/2) are shown at the bottom of the list.

Imagen 18. Vista agenda del paciente como médico.

5.3. Usuario Administrador

Como usuario administrador, podemos hacer lo siguiente:

- Reprogramar las citas de todos los pacientes registrados en la web.
- Asignar fecha a aquellas citas que no las tienen.
- Efectuar o rechazar el cambio de médico que un paciente haya solicitado.
- Consultar la agenda de todos los médicos registrados de la web.

5.3.1. Home

Como usuario administrador, tenemos 2 accesos en la página de inicio: tareas y agenda de los médicos.

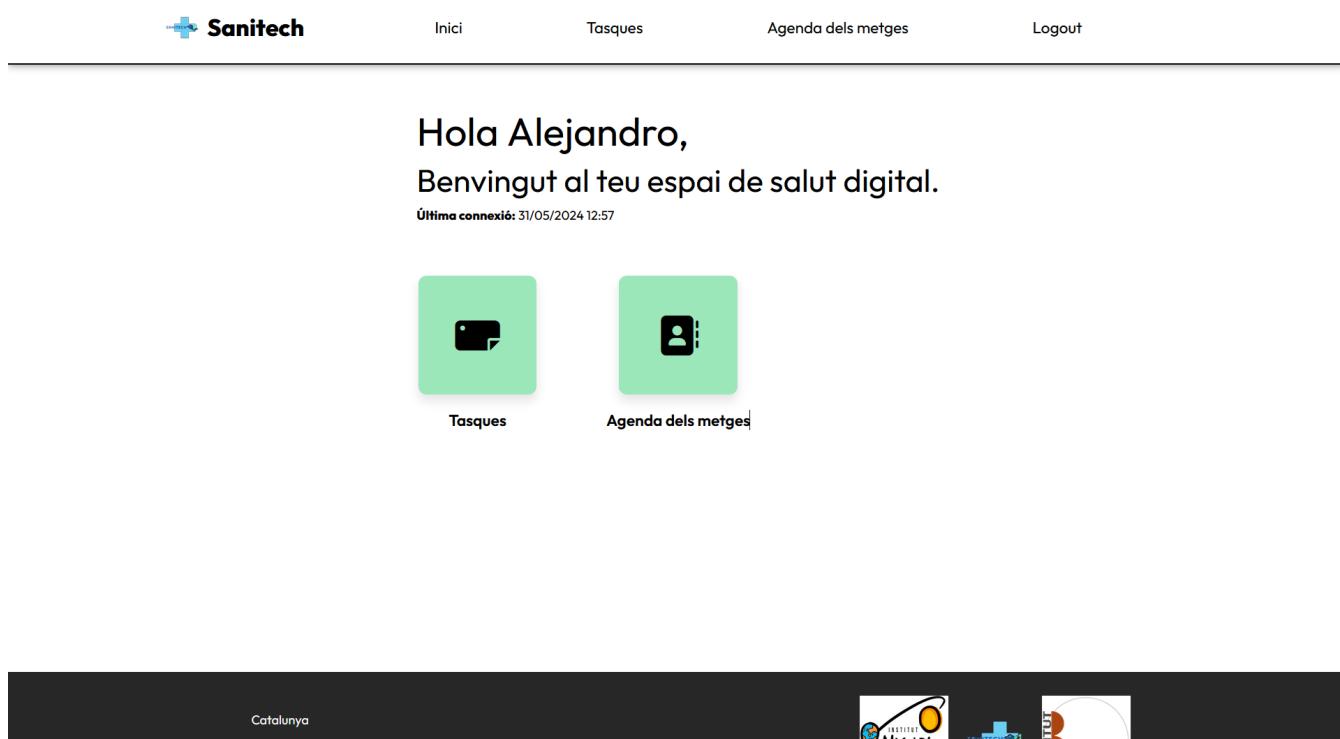


Imagen 19. Vista home administrador.

5.3.2. Tareas

Si accedemos a tareas, podemos ver que como administrador podemos gestionar las citas de todos los médicos y asignar nuevos médicos a los pacientes.

Imagen 20. Vista tareas.

5.3.3. Listado de citas

Al momento de seleccionar *reprogramar cita* o *asignar cita* podremos ver una tabla que contiene todas las citas de todos los pacientes registrados en la aplicación. Cada cita contiene información relevante, como el nivel de emergencia, la fecha y hora, nombre del paciente, DNI, CIP y el nombre de la prueba en cuestión.

Imagen 21. Vista listado de citas.

5.3.4. Reprogramar cita

Cuando reprogramemos una cita, deberemos de seleccionar una en la tabla mencionada anteriormente. Una vez dentro del formulario, veremos en la parte superior información relevante del paciente de la cita seleccionada, y podemos editar cualquier campo de la cita como la especialidad, el médico, la fecha y hora de la cita. El campo número 5 será un mensaje que le llegará al paciente en forma de notificación.

← Tornar

Reprogramar cita

Aquí pots reprogramar la cita d'un pacient

Dades de la cita	
Nom	Sergio
Nivell d'emergència	5
Nom de la prova	Rectoscopia

1. Seleccioni la especialitat:

Selecciona una especialitat

2. Seleccioni el metge:

Selecciona un metge

3. Seleccioni la data de cita:

dd/mm/aaaa

4. Seleccioni la hora de cita:

Selecciona una hora

5. Missatge per al pacient:

Enviar

Imagen 22. Vista reprogramar cita.

← Tornar

Reprogramar cita

Aquí pots reprogramar la cita d'un pacient

1. Seleccioni la especialitat:

Selecciona una especialitat

2. Seleccioni el metge:

Selecciona un metge

3. Seleccioni la data de cita:

dd/mm/aaaa

4. Seleccioni la hora de cita:

Selecciona una hora

5. Missatge per al pacient:

Enviar

Imagen 23. Vista formulario reprogramar cita.

5.3.5. Asignar cita

A la hora de asignar una cita, tendremos que seleccionar una de la tabla mencionada anteriormente (en esta solo se mostrarán las citas que no tengan una fecha asignada) y deberemos de llenar todos los campos del formulario.

Dades de la cita	
Nom	Maria
Nivell d'emergència	2
Nom de la prova	Espirometria

1. Selecció la especialitat:
Selecciona una especialitat

2. Selecció el metge:
Selecciona un metge

Imagen 24. Vista asignar cita como administrativo.

3. Selecció la data de cita:
dd / mm /aaaa

4. Selecció la hora de cita:
Selecciona una hora

5. Missatge per al pacient:

Enviar

Imagen 25. Vista formulario asignar cita.

5.3.6. Cambio de médico por el administrador

Cuando un paciente solicita un cambio de médico, el administrador debe de aceptar o rechazar esta petición. El administrador puede ver la información de quien ha solicitado el cambio y el motivo por el cual quiere cambiar de médico.

The screenshot shows a web-based application for managing medical records. At the top, there is a navigation bar with the Sanitech logo, links for 'Inici' (Home), 'Tasques' (Tasks), 'Agenda dels metges' (Doctors' agenda), and 'Logout'. The main content area has a title 'Solicitud de canvi de metge' (Change of doctor request). Below this, there are two sections: 'Dades del pacient' (Patient data) and 'Motiu del pacient' (Reason of the patient). The 'Dades del pacient' section contains the following information:

Dades del pacient	
Nom	Maria López Garcia
DNI	50321478X
Telèfon	611 22 33 44
Metge actual	Xavier Pelayo López

The 'Motiu del pacient' section contains the reason: 'Prueba'.

At the bottom, there are two buttons: a green 'Aceptar' (Accept) button and a red 'Rebutjar' (Reject) button.

Imagen 26. Vista cambio de médico.

Si aceptamos la solicitud, tendremos que elegir un nuevo médico para nuestro paciente y enviar el formulario. Una vez enviado, el paciente recibirá una notificación diciendo que el cambio de médico fué un éxito.

Sanitech

Inici Tasques Agenda dels metges Logout

Nom	Maria López Garcia
DNI	50321478X
Telèfon	611 22 33 44
Metge actual	Xavier Pelayo López

Motiu del pacient

Prueba

Aceptar **Rebutjar**

Tria el nou metge

Selecciona un metge

Enviar

Imagen 27. Vista aceptación cambio de médico.

Si rechazamos la solicitud, tendremos que especificar el motivo por el rechazo, el cual verá el paciente en forma de notificación.

Sanitech

Inici Tasques Agenda dels metges Logout

Telèfon	611 22 33 44
Metge actual	Xavier Pelayo López

Motiu del pacient

Prueba

Aceptar **Rebutjar**

Especifiqui el motiu de rebuig de la petició

Motiu de rebuig de la petició...

Enviar

Imagen 28. Vista rechazo cambio de médico.

5.3.7. Buscador de médicos

Si decidimos consultar la agenda de los médicos, primero debemos pasar por esta pantalla. En ella podemos seleccionar el médico el cual queremos consultar sus citas programadas con sus pacientes. Si queremos buscar un médico en concreto, solo basta con escribir su nombre en el

buscador.

← Tornar

Cercador metges

Selecciona el metge per veure les seves cites assignades

Buscar metges...

Metges trobats: 10

Xavier Pelayo López
Especialitat : Medicina de família
Nro. Col·legiat: 15662

Encarnación Pinto Alarcón
Especialitat : Pediatría
Nro. Col·legiat: 31761

Olga De Anda
Especialitat : Pediatría
Nro. Col·legiat: 8589

Imagen 29. Vista buscador de médicos.

5.3.8. Agenda de médicos del administrador

Al seleccionar el médico, seremos redirigidos a su agenda, donde podemos ver en forma de tarjetas desplegables todas las citas que tiene asignadas, en las cuales se puede ver la siguiente información: (1) nombre del paciente, (2) fecha de cuando se realiza la cita, (3) hora de cuando se realiza la cita y (4) el motivo de la cita.

← Tornar

Agenda

Aquí pots visualitzar tots els resultats.

José Antonio Curiel

10/06/2024

Hora	Motiu
13:00	Voluptatem atque quia consequatur alias inventore quia pariatur. Et impedit quis repudiandae modi. Id consequatur voluptatum est recusandae.

África Loya

15/06/2024

Rosa María Rubio

17/06/2024

Imagen 30. Vista agenda de los médicos.

Chapter 6. Datos de prueba para testear la aplicación

Para poder hacer pruebas dentro de la aplicación y simular los flujos de funcionamiento desarrollados, dejamos a disposición las credenciales de acceso de un usuario por cada rol contemplado durante la planificación con el cliente.

Tabla 1. Tabla de pruebas

Rol	Nombre	DNI	Contraseña
Administrativo	Alejandro Soto Quientero	12345678D	sanitech
Médico	Xavier Pelayo López	48523671K	sanitech
Paciente 1	Maria López Garcia	50321478X	sanitech
Paciente 2	Juan Martínez Pérez	71985632T	sanitech

Sí se necesita probar con otros datos, o surge alguna duda con el funcionamiento, por favor comuníquese con el equipo de desarrollo.

Chapter 7. Lineas futuras.

Después de la entrega del proyecto, somos conscientes de que no hemos cumplido todos los requerimientos que se plantearon en un principio, estos serían las primeras líneas en desarrollar.

7.1. Requerimientos pendientes

- Funcionalidad para realizar reclamaciones por parte de los pacientes sobre sus citas realizadas.
- Funcionalidad para realizar valoraciones por parte de los pacientes sobre sus citas realizadas.
- Notificaciones push para los pacientes, para un eventual cambio de aplicación web a una aplicación móvil.
- Seleccionar fecha de las citas aplicando el rango de fechas según su nivel de emergencia.
- Historial de acciones realizadas por los usuarios dentro de la aplicación.

7.2. Posibles mejoras

Después de terminar todo lo pendiente del proyecto, hemos pensado alguna posible funcionalidad que se podría añadir al proyecto.

- Una funcionalidad que permita cargar dentro del sistema, el resultado de las pruebas realizadas a los pacientes, ya sea por parte del mismo médico de cabecera o por el administrativo, se ha pensado en un formulario para dicho fin.
- Mostrar únicamente los días hábiles (lunes a viernes) y los días que cuenten con alguna hora disponibles para realizar la solicitud de citas por parte de los pacientes o la asignación de citas por parte de los administrativos de la aplicación.
- Conectividad entre la aplicación desarrollada y el correo electrónico de los usuarios de la aplicación, para tener un respaldo para recibir y contestar notificaciones sobre eventos y acciones realizadas dentro de la aplicación.
- Envío de notificaciones vía SMS a los pacientes, inicialmente fue un requerimiento solicitado por el cliente, sin embargo se descartó ya que los servicios de SMS son de cobro.

Chapter 8. Conclusiones

Después de estos meses de trabajo, hemos de reconocer que estamos contentos con el resultado, tambien somos concientes que el estilo es un aspecto demasiado dinamico y contraversial, que siempre puede ser mejor, nos hubiera gustado tener más tiempo para analizar con mas detenimiento, sin embargo tenemos un resultado del que podemos sentirnos orgullosos. En cualquier caso, pensamos que lo importante realmente es el aprendizaje que nos llevamos, más que el producto final.

Desde luego ahora que estamos más familiarizados con todas estas tecnologías, las cuales hemos ido reforzando desde el proyecto anterior, se ha mejorado la logica y la estructura de un proyecto *laravel* junto con un framework para el front-end como lo es *Vue*.

Durante los desarrollos realizados, nos enfrentamos a una serie de errores tanto del lado del servidor y del lado del cliente, los cuales fueron solucionados y corregidos y que tambien hacen parte del aprendizaje continuo, ya que refuerzan nuestros conocimientos y quedan como experiencia para nuestro futuro como desarrolladores web.

8.1. Desviaciones en la Planificación

En este apartado expondremos los puntos que inicialmente se plantearon en la reunión inicial y que no fueron entregados en la entrega formal al cliente y los que no fueron entregados en la entrega final al equipo docente del instituto.

8.1.1. Entrega al cliente

- Funcionalidad para solicitar el cambio de médico de cabecera por parte del paciente.
- Funcionalidad para realizar reclamaciones por parte de los pacientes sobre sus citas realizadas.
- Funcionalidad para realizar valoraciones por parte de los pacientes sobre sus citas realizadas.
- Historial de acciones realizadas por los usuarios dentro de la aplicación.

8.1.2. Entrega al equipo docente

- Funcionalidad para realizar reclamaciones por parte de los pacientes sobre sus citas realizadas.
- Funcionalidad para realizar valoraciones por parte de los pacientes sobre sus citas realizadas.
- Historial de acciones realizadas por los usuarios dentro de la aplicación.

8.2. Aportaciones del Proyecto a los Conocimientos del Alumno

Al final, después de todo, creemos que a parte del conocimiento específico de cada nueva tecnología con la que hemos tratado en este proyecto, lo más importante que nos llevamos es el darnos cuenta de lo importante que es saber organizarse bien el tiempo.

Tratar con un cliente real, fue un nuevo reto que afrontar, donde estos nos expresaban sus

necesidades principales, requerimientos y lo que esperaban de la aplicación a desarrollar y nosotros como equipo organizábamos las posibles soluciones tanto de logica y de diseño.

Saber tomar las decisiones adecuadas, qué sacar adelante y qué dejar atrás, ir a lo más práctico siempre y de ahí construir si se puede permitir el tiempo. En un proyecto el tiempo es el recurso más valioso, por fortuna supimos priorizar los desarrollos y funcionalidades de la aplicación y así cumplir los requerimientos de nuestros clientes.

En cuanto a conocimientos específicos, ahora nos vemos capaces de enfrentarnos a proyectos de este calibre, con frameworks que no hemos tocado e incluso mediante arquitecturas MVC. Hemos aprendido lo necesario para servir una página web o una API en un servidor. Cómo funciona una API, a desarrollar una y cómo conectar con ella y lanzarle *requests*.

Tambien dimos grandes pasos en un nuevo paradigma de programación, la programación reactiva, mediante el framework Vue y toda su versatilidad, componentes, variables reactivas, *props* y *emits* entre componentes padre e hijos, observadores, etc.

Chapter 9. Webgrafía

En esta sección os presentaremos aquellos recursos web a los que hemos accedido para documentarnos en la programación en varios lenguajes o el uso de algunas APIs.

[Mozilla Develop Network](#): Css, JavaScript.

[Laracasts](#), y [Página oficial de Laravel](#): Laravel y extensiones como gd.

[w3schools](#): Css, JavaScript.

[Repositorio GIT sass-boilerplate, de KittyGiraude](#)l: sass usando paradigma de estructuración 7:1.

[StackOverflow](#): todo tipo de consultas generales sobre todas las tecnologías usadas.

[ChatGPT](#): consultas generales y ayuda de corrección sintáctica.

[Font Awesome](#): fuente de iconografía.

[Google Fonts](#): fuente de tipografía.

[Color](#)s: pruebas de paletas de colores.

[Colormind](#).io: pruebas de paletas de colores.

[Colorable.jxnblk](#): pruebas de contraste de colores entre texto y fondo.