

PROYECTO STORAGE CONTROLLER



Realizado por: Gonzalo Alcaide Barbero

ÍNDICE DE CONTENIDO

1.	INTRODUCCIÓN	1
2.	OBJETIVOS Y REQUISITOS	1
a)	OBJETIVOS	1
b)	REQUISITOS	1
3.	ESTUDIO PREVIO	2
a)	PRIMERA SOLUCIÓN	2
b)	SEGUNDA SOLUCIÓN	2
c)	TERCERA SOLUCIÓN	2
d)	SOLUCION ELEGIDA	2
4.	PLAN DE TRABAJO	3
5.	DISEÑO	4
a)	DISEÑO DE BASE DE DATOS	4
b)	TABLAS Y NORMALIZACIÓN	5
c)	DIAGRAMA RELACIONAL	6
d)	DISEÑO GENERAL	7
e)	DISEÑO DETALLADO	8
f)	BOCETO DE APLICACIÓN	9
6.	IMPLEMENTACIÓN	11
a)	HOSTING DB	11
b)	STORE CONTROLLER	14
1.	BACKEND	14
2.	FRONTEND	16
7.	SEGURIDAD	30
8.	PRUEBAS	31
9.	RECURSOS	41
a)	RECURSOS HARDWARE	41
b)	RECURSOS SOFTWARE	41
c)	PRESUPUESTO	42
10.	CONCLUSIÓN	42
a)	CONSECUCCIÓN DE LOS OBJETIVOS	42
b)	PROBLEMAS ENCONTRADOS	43
c)	MEJORAS	44
11.	BIBLIOGRAFÍA	45
12.	ANEXO	46
a)	MANUAL DE USUARIO	46

ÍNDICE DE FIGURAS

Ilustración 1 – Logo React JS	2
Ilustración 2 - Tabla plan de trabajo	3
Ilustración 3 - Entidad/Relación	4
Ilustración 4 - Diagrama relacional	6
Ilustración 5 - Diagrama diseño general	7
Ilustración 6 - Diseño diagrama específico	8
Ilustración 7 - Activity principal	9
Ilustración 8 - Activity Almacenes	9
Ilustración 9 - Activity Armarios	10
Ilustración 10 - Activity Objetos	10
Ilustración 11 - ProfesionalHosting	11
Ilustración 12 - Página Inicio Hosting	11
Ilustración 13 - Estructura DB	12
Ilustración 14 - Instalación de Cloudinary	12
Ilustración 15 - Código de inserción imagen	13
Ilustración 16 - Upload_preset Cloudinary	13
Ilustración 17 - Instalación Composer	14
Ilustración 18 - Instalación de laravel	14
Ilustración 19 - Configuración .env laravel	15
Ilustración 20 - Creación modelos laravel	15
Ilustración 21 - Creación controlador laravel	15
Ilustración 22 - Funcionalidad API	15
Ilustración 23 - Instalación React	16
Ilustración 24 - Instalación Tailwindcss	16
Ilustración 25 - Importación Tailwindcss	16
Ilustración 26 - Tailwindcss en config	17
Ilustración 27 - Rutas relativas	17
Ilustración 28 - Instalación React-router-dom	17
Ilustración 29 - Instalación librería formularios	18
Ilustración 30 - Input con librería	18
Ilustración 31 - Instalación iconos	18
Ilustración 32 - Código para icono	18
Ilustración 33 - Formulario con librerías	18
Ilustración 34 - Contenido pantalla Login	19
Ilustración 35 - Contenido pantalla Registro	19
Ilustración 36 - Contenido pantalla recuperar contraseña	20
Ilustración 37 - Contenido pantalla actualización contraseña	20
Ilustración 38 - Código await fetch4	21
Ilustración 39 - Navegación con navigate	21
Ilustración 40 - Navegación con Link	21
Ilustración 41 - Pantalla Menú Casas	21
Ilustración 42 - Pantalla Creación Casas	22
Ilustración 43 - Pantalla Actualización Casas	22
Ilustración 44 - Pantalla Eliminación Casas	23
Ilustración 45 - Código para buscador	23
Ilustración 46 - Pantalla Menú Armarios	24

Ilustración 47 - Instalación QR.....	24
Ilustración 48 - Mostrado QR	24
Ilustración 49 – URL del QR	24
Ilustración 50 - Pantalla Creación Armarios	25
Ilustración 51 - Pantalla Actualización Armarios	25
Ilustración 52 - Pantalla Eliminación Armarios	26
Ilustración 53 - Pantalla QR Armarios.....	26
Ilustración 54 - Pantalla Armario QR	27
Ilustración 55 - Pantalla Menú Objetos	27
Ilustración 56 - Pantalla Creación Objetos	28
Ilustración 57 - Pantalla Actualizar Productos	28
Ilustración 58 - Pantalla Eliminación Objeto.....	29
Ilustración 59 – GitHub.....	30
Ilustración 60 - Inicio de sesión con credenciales erróneas.....	31
Ilustración 61 - Registro con contraseña muy pequeña	31
Ilustración 62 - Registro con contraseñas distintas	31
Ilustración 63 - Registro con usuario existente	31
Ilustración 64 - Recuperación de cuenta no existen.....	32
Ilustración 65 - Recuperación de contraseña exitosa	32
Ilustración 66 - Pantalla de casas con datos	32
Ilustración 67 - Pantalla de casas sin datos	32
Ilustración 68 - Actualización de casas	33
Ilustración 69 - Actualización realizada correctamente.....	33
Ilustración 70 - Eliminación de casas	33
Ilustración 71 - Eliminación realizada	33
Ilustración 72 - Inserción casas.....	34
Ilustración 73 - Casa insertada	34
Ilustración 74 - Pantalla de armarios con datos	34
Ilustración 75 - Pantalla de armarios sin datos.....	34
Ilustración 76 - Inserción de armario.....	35
Ilustración 77 - Armario insertado.....	35
Ilustración 78 - Actualización de armario	35
Ilustración 79 - Armario actualizado.....	35
Ilustración 80 - Pantalla QR sin datos	36
Ilustración 81 - Pantalla QR con datos.....	36
Ilustración 82 - Eliminación de armario.....	37
Ilustración 83 - Armario eliminado	37
Ilustración 84 - Menú objetos con datos	37
Ilustración 85 - Menú objetos sin datos	37
Ilustración 86 - Inserción de objetos	38
Ilustración 87 - Objetos insertados.....	38
Ilustración 88 - Actualización producto	39
Ilustración 89 - Producto actualizado	39
Ilustración 90 - Eliminación de objeto	40
Ilustración 91 - Objeto eliminado	40
Ilustración 92 - Listado sin búsqueda	40
Ilustración 93 - Listado con búsqueda	40
Ilustración 94 - Tabla presupuesto	42

Ilustración 95 - Tabla objetivos	42
Ilustración 96 - Clave Foreign arreglada	43
Ilustración 97 - Enlace DB arreglado.....	43

1. INTRODUCCIÓN

Este proyecto consiste en la creación de una página web que se llamara “STORAGE CONTROLLER”, consiste en una página web en la que podremos crear diferentes espacios de trabajos a los que denominaremos casas de forma indefinida, dentro de estas casas tendrán una serie de almacenes que guardarán diversos elementos.

El usuario podrá crear las casas que desee al igual que podrá crear los almacenes e introducir los elementos sin límite.

La web está desarrollada con un formato responsibe que permita a todos su uso.

2. OBJETIVOS Y REQUISITOS

a) OBJETIVOS

- Control de almacenamiento.
- Creación de QR en cada armario.
- Botón que nos permita imprimir el QR o compartirlo con otro usuario.
- Buscador que nos permite buscar un elemento y nos muestra en el almacén que se encuentra.
- Poder borrar tanto las casas como los almacenes o los elementos insertados.
- Desplegar la web en un hosting.

b) REQUISITOS

- Usuario y clave para acceder a la aplicación.
- Menú para poder acceder tanto a las diferentes casas como a los diferentes almacenes.
- Escena que permite escanear el QR de un almacén y te muestra el contenido de dicho almacén.

3. ESTUDIO PREVIO

En este apartado voy a realizar una recopilación y análisis de los datos necesarios, para definir en líneas generales las posibles soluciones que podemos llevar a cabo para nuestro proyecto.

La idea principal del desarrollo de nuestra web es usar un software/framework que nos permita crear una aplicación de navegador.

a) PRIMERA SOLUCIÓN.

- La primera opción que he tenido es usar jQuery, ya que es un framework basado en Javascript diseñado para crear sitios web y aplicaciones accesibles en todos los teléfonos, tabletas y dispositivos de escritorio.

b) SEGUNDA SOLUCIÓN

- La segunda opción es trabajar con el framework de React.js, lo que permitirá desarrollar aplicaciones web dinámicas y escalables de manera eficiente, aprovechando su arquitectura basada en componentes reutilizables, su virtual DOM para mejorar el rendimiento y su amplia comunidad de soporte.

c) TERCERA SOLUCIÓN

- La tercera opción es trabajar con un framework llamado AngularJS, este nos permite desarrollar aplicaciones móviles híbridas.

d) SOLUCION ELEGIDA

- En conclusión, la opción elegida con la cual voy a llevar a cabo el trabajo es **REACT.JS**, ya que este me proporciona más eficiencia, ayudas y documentación.

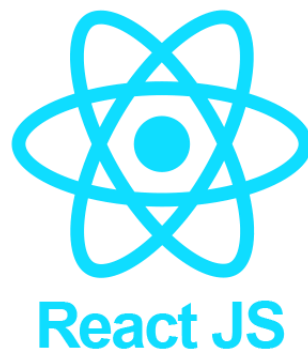


Ilustración 1 – Logo React JS

4. PLAN DE TRABAJO

PLAN DE TRABAJO	
SEMANAS	TAREAS
20 al 31 de marzo de 2025	Introducción, objetivos y requisitos y plan de trabajo.
1 al 5 de abril de 2025	Estudio previo y diseño del proyecto.
10 al 11 de abril de 2025	Búsqueda de recursos.
12 al 14 de abril de 2025	Implementación y QA.
17 al 21 de abril de 2025	Implementación y QA.
24 al 28 de abril de 2025	Implementación y QA.
1 al 5 de mayo 2025	Implementación y QA.
8 al 11 de mayo 2025	Implementación y QA.
15 al 19 de mayo de 2025	Implementación, QA y pruebas UAT.
22 al 26 de mayo de 2025	Terminar implementación y empezar con la seguridad.
29 al 31 de mayo de 2025	Pruebas de desarrollo y Recursos.
1 al 9 de junio de 2025	Conclusión y bibliografía y Anexo.
12 al 14 de junio de 2025	Realizar el manual de usuario.
15 al 16 de junio de 2025	Entrega del proyecto en el centro.

Ilustración 2 - Tabla plan de trabajo

5. DISEÑO

a) DISEÑO DE BASE DE DATOS

ENTIDADES

- **USUARIOS:** Esta entidad almacenara los datos de los usuarios. Esta tabla guardara el id del usuario y el correo con el que sea logueado.
- **CASAS / ALMACENES:** Esta entidad almacenara los almacenes / casas que tenga creado dicho usuario. Aquí se guardarán el nombre del almacén/casa, una imagen de este mismo y una descripción por si quieres añadir alguna nota o especificación. También nos guardara el usuario que ha creado dicho almacén.
- **CAJONES / ARMARIOS:** Esta entidad almacenara los armarios/cajones que tenga creado cada usuario. Aquí se guardará el nombre del armario/cajón y una descripción por si queremos añadir alguna nota o especificación.
- **OBJETOS:** Esta entidad almacenara los objetos que estarán dentro de los armarios/cajones que han sido introducidos por el usuario que controla dichos cajones. Aquí se guardarán el nombre del objeto, una imagen de dicho objeto, una descripción por si queremos poner alguna nota del objeto y se podrá la fecha de introducción del objeto por si queremos saber cuándo se introdujo.

RELACIONES

ROJO → Identificación | **NEGRO** → Normales | **AZULES** → Opcional

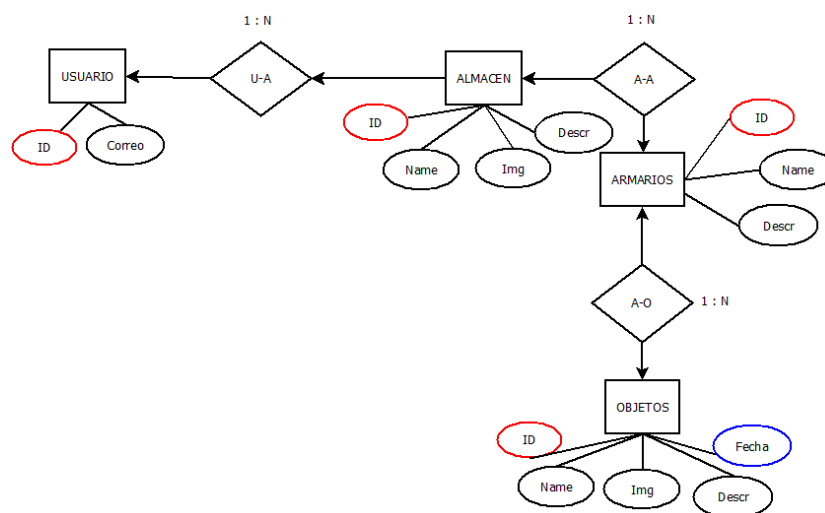


Ilustración 3 - Entidad/Relación

b) TABLAS Y NORMALIZACIÓN

USUARIOS (ID, Correo)

- Clave primaria (**ID**): único, no nulo.

1FN: Se encuentra en primera forma normal porque cada atributo tiene un valor en cada fila.

2FN Se encuentra en segunda forma normal porque está en 1FN y la clave primaria está formada por un único campo.

3FN: Se encuentra en 3FN porque está en 2FN y además no hay ningún campo ajeno a la clave primaria por el cual podemos saber el valor de otro campo.

ALMACENES (ID, Name, Imagen, Descripción, id_usuario)

- Clave primaria (**ID**): único, no nulo.
- Clave externa (**id_usuario**) referenciado a **ID** de la tabla **USUARIOS**

1FN: Se encuentra en primera forma normal porque cada atributo tiene un valor en cada fila.

2FN Se encuentra en segunda forma normal porque está en 1FN y la clave primaria está formada por un único campo.

3FN: Se encuentra en 3FN porque está en 2FN y además no hay ningún campo ajeno a la clave primaria por el cual podemos saber el valor de otro campo.

ARMARIOS (ID, Name, Descripción, id_almacen)

- Clave primaria (**ID**): único, no nulo.
- Clave externa (**id_almacen**) referenciado a **ID** de la tabla **ALMACENES**.
 - **ACTUALIZACIÓN**: Cascada.
 - **EN BORRADO**: Cascada.

1FN: Se encuentra en primera forma normal porque cada atributo tiene un valor en cada fila.

2FN Se encuentra en segunda forma normal porque está en 1FN y la clave primaria está formada por un único campo.

3FN: Se encuentra en 3FN porque está en 2FN y además no hay ningún campo ajeno a la clave primaria por el cual podemos saber el valor de otro campo.

OBJETOS (ID, Name, Imagen, Descripción, Fecha, id_armario)

- Clave primaria (**ID**): único, no nulo.
- Clave externa (**id_armario**) referenciado a **ID** de la tabla **ARMARIOS**.
 - **ACTUALIZACION**: Cascada.
 - **EN BORRADO**: Cascada.

1FN: Se encuentra en primera forma normal porque cada atributo tiene un valor en cada fila.

2FN Se encuentra en segunda forma normal porque está en 1FN y la clave primaria está formada por un único campo.

3FN: Se encuentra en 3FN porque está en 2FN y además no hay ningún campo ajeno a la clave primaria por el cual podemos saber el valor de otro campo.

c) DIAGRAMA RELACIONAL

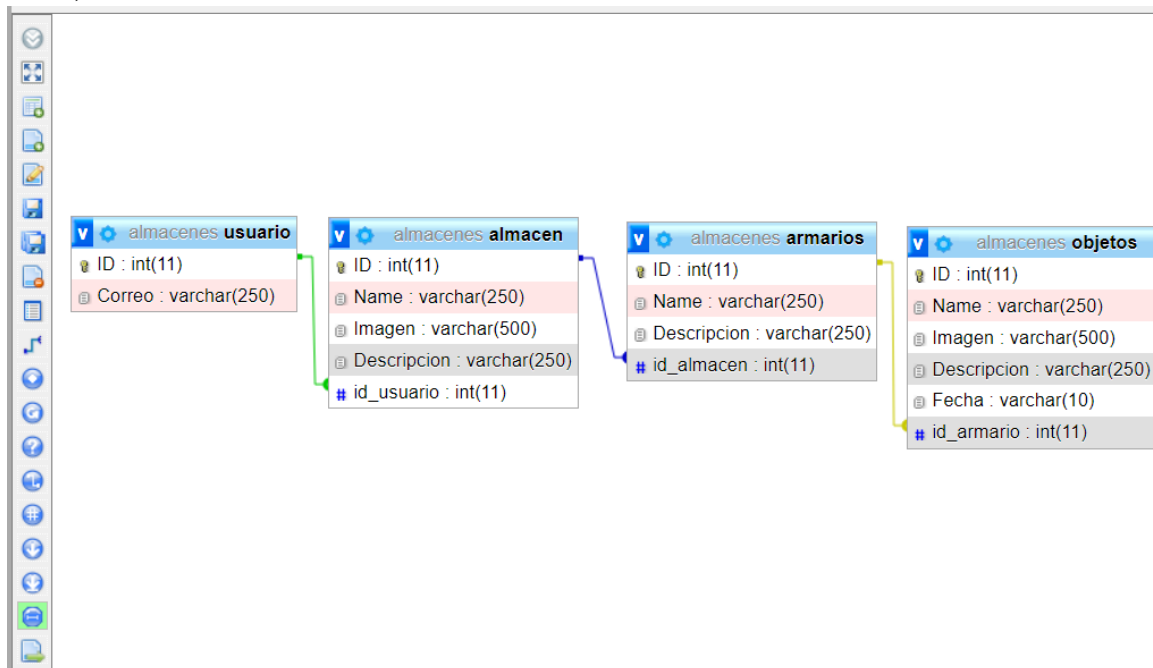


Ilustración 4 - Diagrama relacional

d) DISEÑO GENERAL

Cuando el usuario abra la web, lo primero que comprobaremos si está registrado:

- Si esta registrado, tendrá que iniciar sesión, nos logueamos, puede ocurrir que la clave sea correcta o incorrecta.
- Si la clave es incorrecta, preguntamos si ha olvidado la clave, si ha olvidado la clave, nos vamos a la ventana de recuperar clave. Introducimos la nueva clave y vamos al menú principal. Si no ha olvidado, vamos a la ventana de inicio de sesión e introducimos la clave.
- Si la clave es correcta, pasamos al menú de gestión de almacenamiento.
- Si no está registrado, iremos a la pestaña de registro, introducimos el correo y la contraseña y si todo está bien vamos al menú principal.

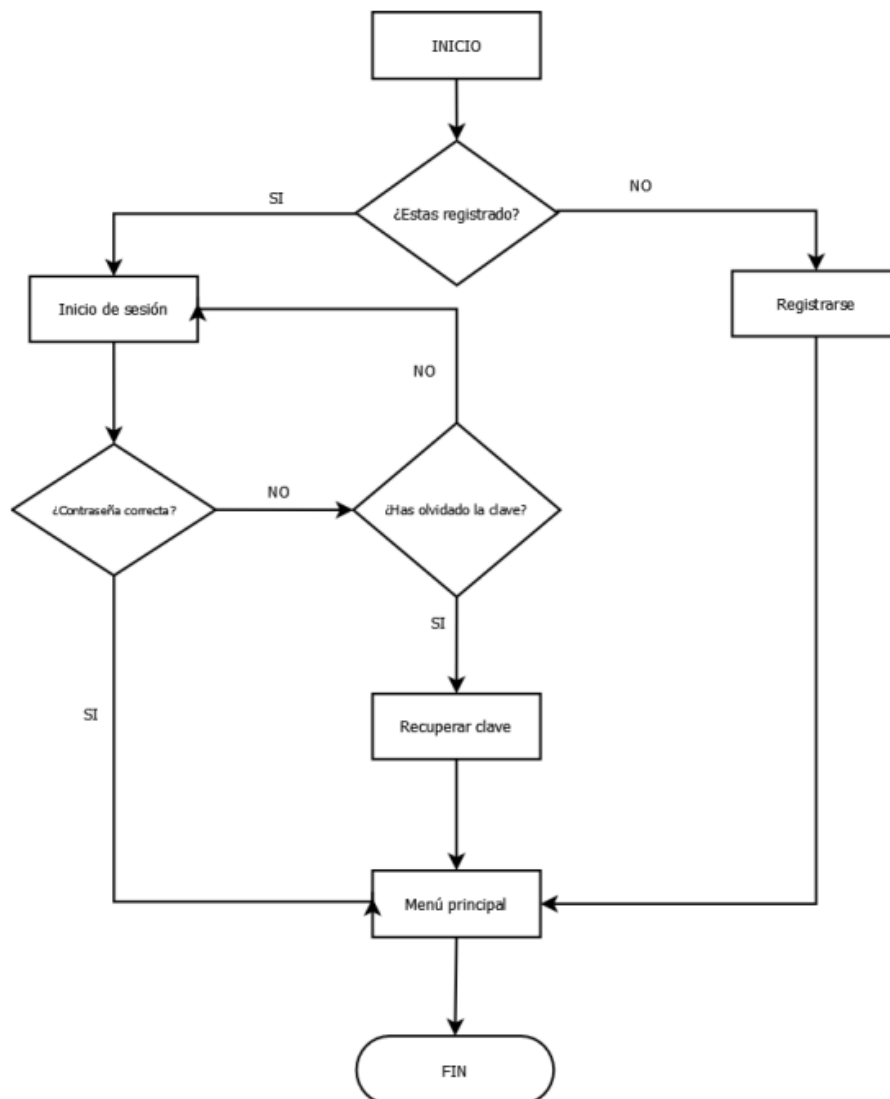


Ilustración 5 - Diagrama diseño general

e) DISEÑO DETALLADO

Cuando nos registramos en la web o iniciamos sesión en caso de ya estar registrados, vamos a acceder a un menú principal. En este nos dará la oportunidad de crear algún almacén o de ver el contenido de alguno ya creado.

Dentro de este almacén podremos crear cajones o en caso de tener alguno podremos ver el contenido de este.

En cada apartado tendremos la opción de modificar o eliminar los elementos que queramos.

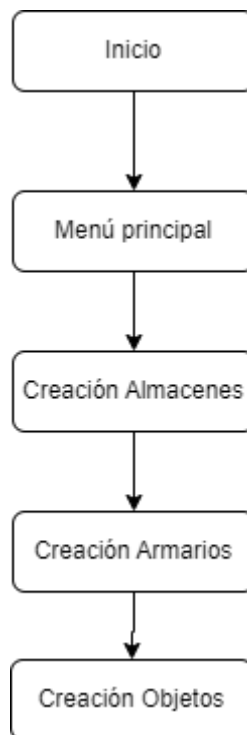


Ilustración 6 - Diseño diagrama específico

f) BOCETO DE APLICACIÓN

En este apartado tendremos un concepto inicial de la estructura de nuestra web, los bocetos mostrados se han realizado con figma.

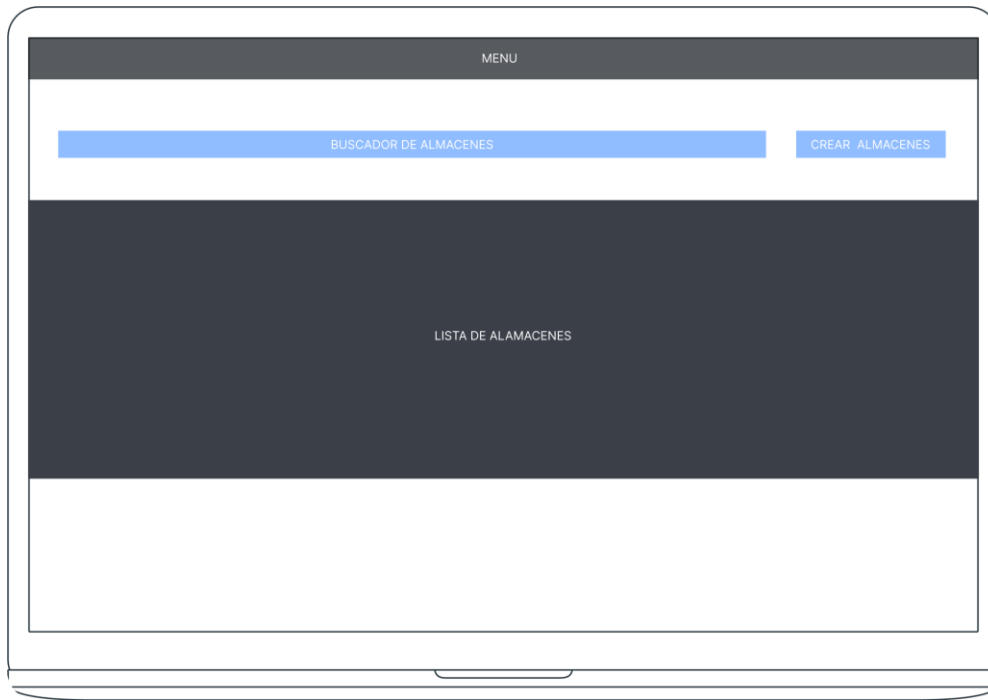


Ilustración 7 - Activity principal

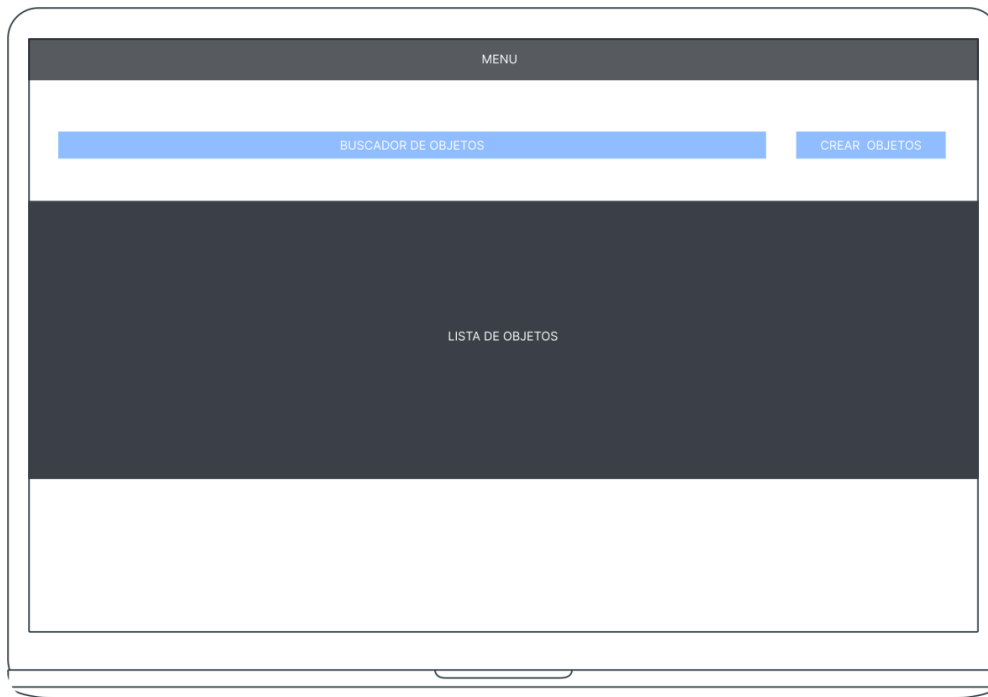


Ilustración 8 - Activity Almacenes

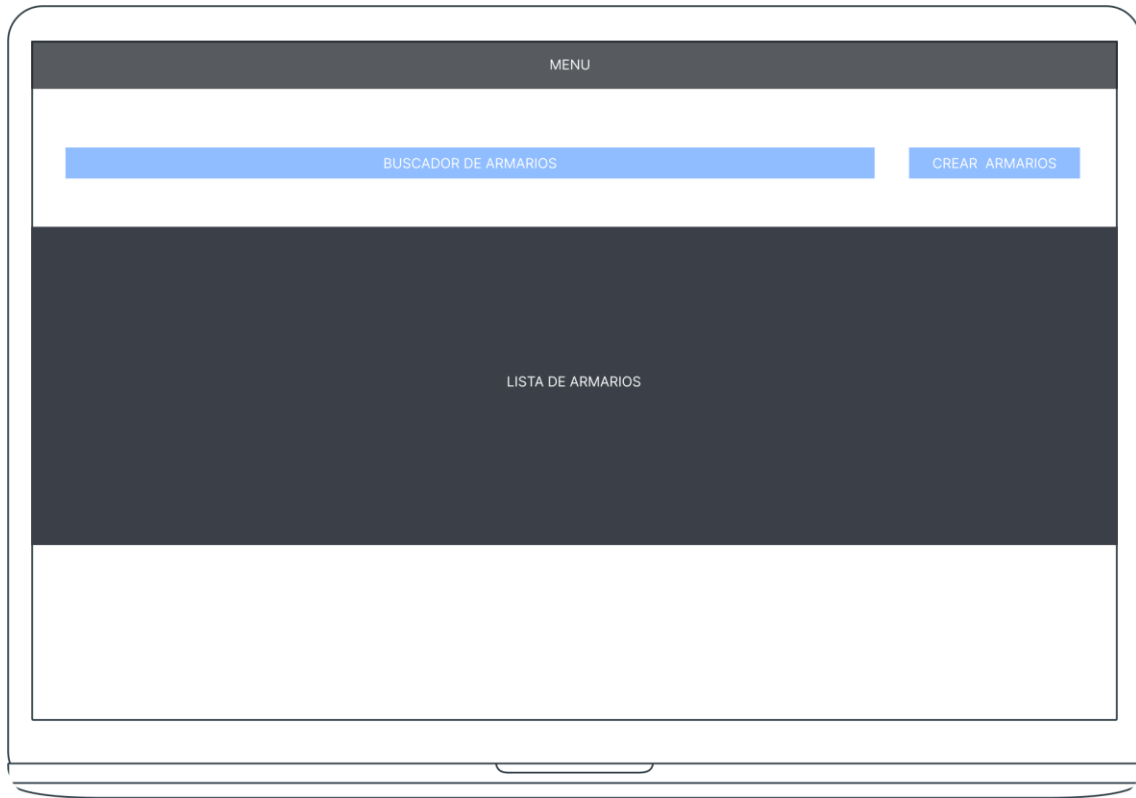


Ilustración 9 - Activity Armarios

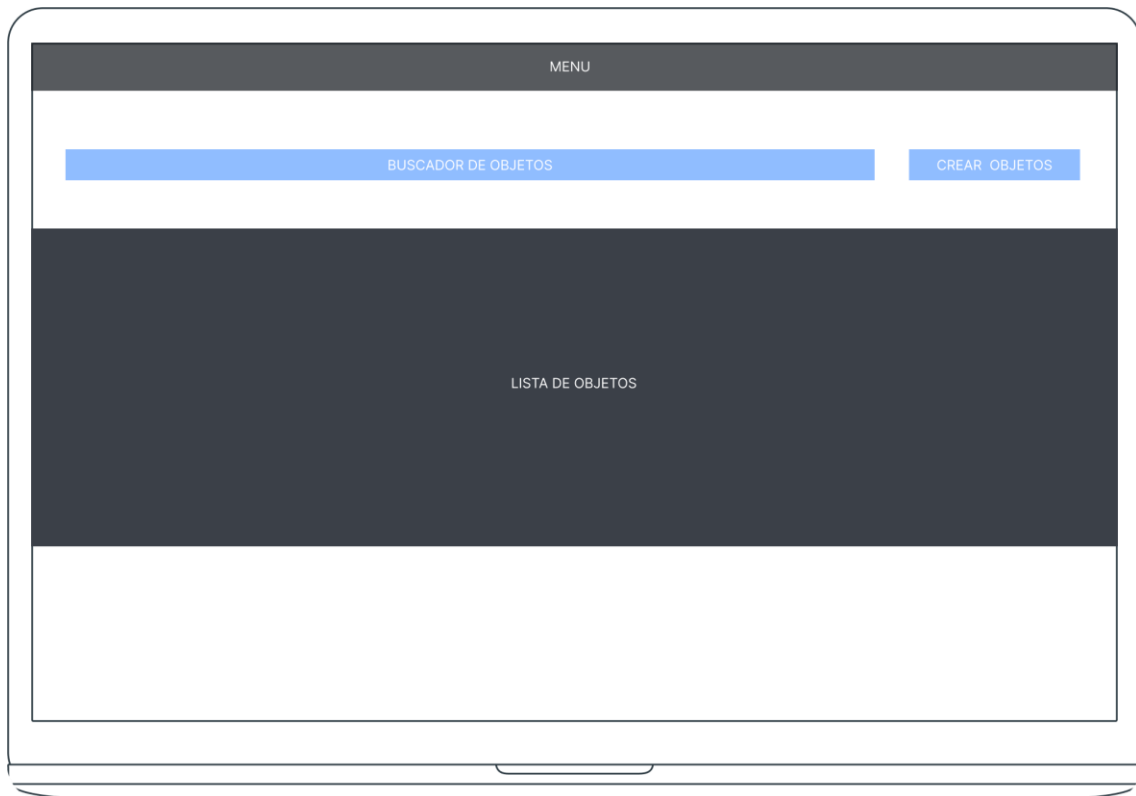


Ilustración 10 - Activity Objetos

6. IMPLEMENTACIÓN

a) HOSTING DB POST CREACIÓN DB.

- Para el hosting de nuestra página web y de nuestra base de datos el centro educativo nos ha contratado un hosting online llamado profesional hosting. Este nos permite tanto crear varias bases de datos como subir nuestro proyecto en su respectivo repositorio para posteriormente poder acceder a él desde cualquier dispositivo



Ilustración 11 - ProfesionalHosting

CREACIÓN DB EN PROFESIONALHOSTING

- Antes realizar la implementación de la web tenemos que crear nuestra base de datos donde almacenaremos los datos de los usuarios
- Nos iremos a la plataforma otorgada por el centro he iniciaremos sesión con las credenciales otorgadas por ellos.

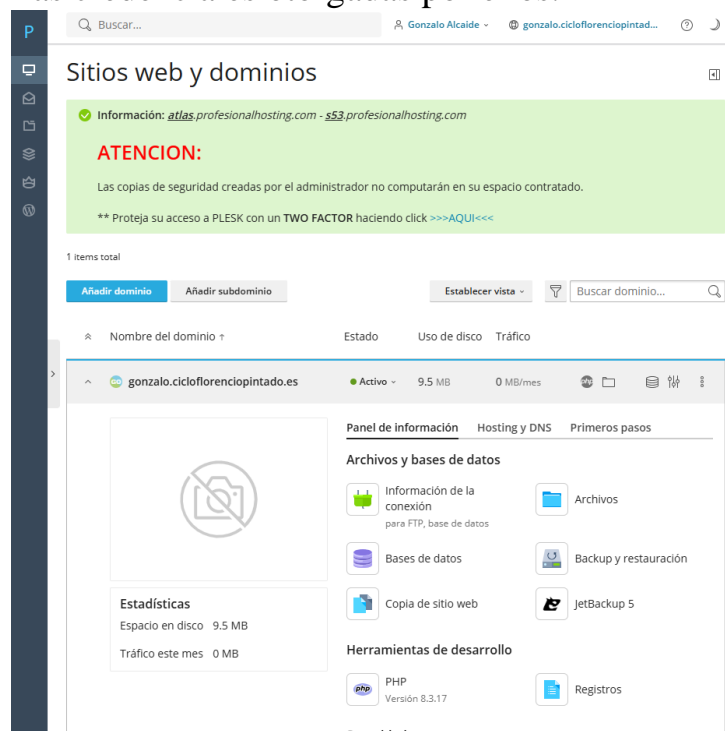


Ilustración 12 - Página Inicio Hosting

- Una vez dentro nos iremos al apartado de gestión de bases de datos y crearemos la nuestra para posteriormente importar la plantilla que hemos realizado anteriormente en el estudio.

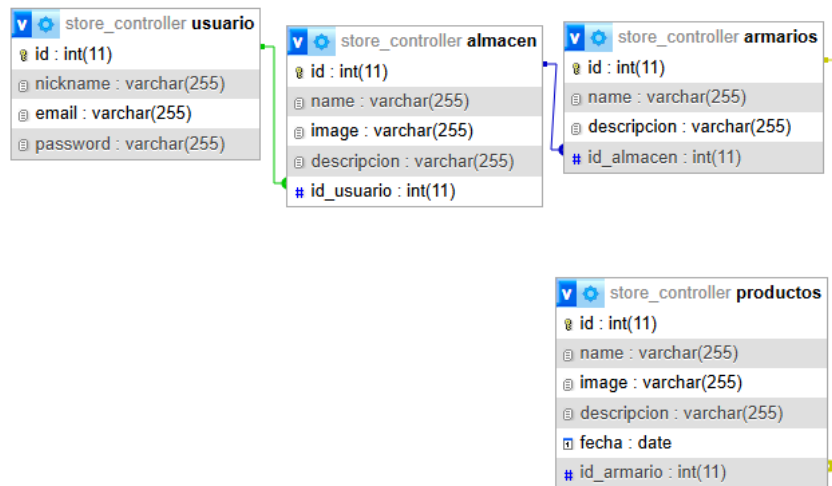


Ilustración 13 - Estructura DB

CREACIÓN DE CONEXIÓN CON CLOUDINARY

Mientras realizábamos la implementación de nuestro código nos hemos dado cuenta de un factor que no teníamos previsto, al estar utilizando React como tecnología para el funcionamiento de nuestra web tenemos el inconveniente de que esta no nos permite realizar subida de ficheros a nuestro proyecto. Necesitaríamos una petición backend la cual realice esto o la opción que hemos escogido en nuestro caso utilizar la plataforma Cloudinary para el almacenamiento de nuestras imágenes.

IMPLEMENTACIÓN CLOUDINARY

Tendremos que instalar el siguiente comando en nuestro proyecto para el funcionamiento de Cloudinary. Además de tener una cuenta creada.

```
npm install cloudinary
```

Ilustración 14 - Instalación de Cloudinary

Una vez instalado en nuestro proyecto usaremos el siguiente código para realizar las subidas de imágenes a nuestra plataforma. (Los datos tapados serán los datos que sacaremos de almacenamiento en Cloudinary para indicar donde realizaremos la subida de estas imágenes).

```
let imageUrl = null
if (selectedFile) {
  const formData = new FormData()
  formData.append("file", selectedFile)
  formData.append("upload_preset", [REDACTED])
  formData.append("cloud_name", [REDACTED])
  const cloudinaryRes = await fetch([REDACTED], {
    method: "POST",
    body: formData,
  })

  const result = await cloudinaryRes.json()

  if (!cloudinaryRes.ok) {
    throw new Error(result.error?.message || "Error al subir la imagen a Cloudinary")
  }

  imageUrl = result.secure_url
}
```

Ilustración 15 - Código de inserción imagen

Para esto tenemos que tener en cuenta que nuestro upload_preset tenga esta configuración en caso contrario nos dará un error y no nos subirá la imagen a nuestro almacenamiento.

Signing mode ⓘ

Unsigned ▼

Select 'Unsigned' for uploading directly from the browser or embedded Upload Widget, and 'Signed' for Media Library uploads.

Ilustración 16 - Upload_preset Cloudinary

b) STORE CONTROLLER

Una vez tengamos nuestro servidor funcionando podemos empezar el desarrollo e implementación de nuestro proyecto.

Para nuestro proyecto necesitaremos tener dos partes.

1. BACKEND

Nuestro backend se ocupará de conectar nuestro proyecto con nuestra base de datos almacenada en el hosting. Se ocupará de la inserción, eliminación y actualización de los datos. Además, será el encargado que nos permitirá mostrarle al usuario los datos que este mismo tiene.

INSTALACIONES NECESARIOS

Lo primero y más importante para instalar laravel y poder realizar nuestras peticiones con la base de datos será descargar e instalar composer. Ya que este se encarga de instalarla y actualizar las dependencias de laravel de forma correcta.

URL DESCARGA COMPOSER

<https://getcomposer.org/>



Ilustración 17 - Instalación Composer

Una vez instalado composer nos iremos a nuestro proyecto y usaremos el siguiente comando para la creación del proyecto, esto nos creara la estructura de nuestro laravel para realizar las peticiones.

```
composer create-project laravel/laravel backend-store-controller
```

Ilustración 18 - Instalación de laravel

CONFIGURACIÓN DE LARAVEL

Una vez tengamos creado nuestro proyecto tendremos que configurar el archivo llamado .env. Este se encargará de guardar los datos necesarios para la conexión de nuestra base de datos con el código que implementaremos.

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_DATABASE=store_controller
DB_PORT=3307
DB_USERNAME=root
DB_PASSWORD=
```

Ilustración 19 - Configuración .env laravel

CREACIÓN DE COMPENENTES Y FUNCIONALIDAD

Primeramente tendremos que crear dentro de nuestro proyecto unos modelos con los valores que tendrá nuestra tabla en la base de datos

```
php artisan make:model Usuario
```

Ilustración 20 - Creación modelos laravel

Una vez creado nuestro modelo crearemos la funcionalidad, para ello crearemos un controlador. Este tendrá toda la funcionalidad para realizar peticiones, actualizaciones, etc...

```
php artisan make:controller Api/usuarioController
```

Ilustración 21 - Creación controlador laravel

Una vez creado todo esto nos iremos a la carpeta routers y en nuestro archivo api.php crearemos las rutas para acceder a nuestras peticiones creadas en el controlador.

```
Route::post('/register', [usuarioController::class, 'createUser']);
Route::post('/login', [usuarioController::class, 'login']);
Route::post('/forgot-password', [usuarioController::class, 'forgotPassword']);
Route::post('/reset-password', [usuarioController::class, 'resetPassword']);
```

Ilustración 22 - Funcionalidad API

2. FRONTEND

Nuestro frontend estará realizado con React, este se encargará de mostrar los datos del backend para el usuario. Investigando hemos visto que para la navegación entre pantallas al estar utilizando el framework de React necesitamos una librería que nos permite hacer esto mismo.

INSTALACIONES Y CONFIGURACIÓN NECESARIOS

En nuestro caso vamos a usar vite para la creación del proyecto con React. Lo primero que tendremos que hacer es irnos a nuestro IDE correspondiente y usar el siguiente comando. Este nos permitirá elegir entre varios framework, pero nosotros elegiremos React + TypeScript.



```
npm create vite@latest
```

Ilustración 23 - Instalación React

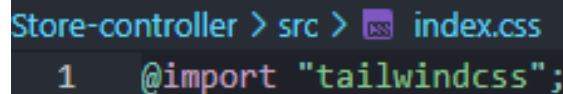
Una vez instalado nuestro proyecto y creado vamos a añadir Tailwindcss para la mejora de nuestras interfaces. Tailwindcss es un framework CSS.



```
npm install tailwindcss @tailwindcss/vite
```

Ilustración 24 - Instalación Tailwindcss

Para el funcionamiento correcto de este framework tendremos que importarlo en nuestro index.css.



```
Store-controller > src > index.css  
1 @import "tailwindcss";
```

Ilustración 25 - Importación Tailwindcss

Además, tendremos que añadirlo al archivo de vite.config para terminar de configurar el funcionamiento

```
import React from '@vitejs/plugin-react'
import tailwindcss from '@tailwindcss/vite'

// https://vite.dev/config/
export default defineConfig({
  plugins: [react(), tailwindcss()],
  resolve: {
    alias: {
      '@': fileURLToPath(new URL('./src', import.meta.url))
    }
  }
})
```

Ilustración 26 - Tailwindcss en config

Por otro lado, configuraremos las rutas relativas para no tener tanto problema a la hora de recoger cualquier componente. En nuestro archivo tsconfig.app.js añadiremos lo siguiente.

```
{
  "baseUrl": "src",
  "paths": {
    "@/*": ["./*"],
    "@/components/*": [".components/*"],
    "@/contexts/*": [".contexts/*"],
    "@/views/*": [".views/*"],
    "@/layouts/*": [".layouts/*"],
    "@/hooks/*": [".hooks/*"],
    "@/utils/*": [".utils/*"],
    "@/store/*": [".store/*"],
    "@/assets/*": [".assets/*"],
    "@/styles/*": [".styles/*"]
  }
},
```

Ilustración 27 - Rutas relativas

Investigando en el desarrollo hemos descubierto que en React no podemos hacer la navegación de manera normal por lo cual necesitaremos instalar el siguiente comando para poder realizar la navegación entre páginas.

```
npm i react-router-dom
```

Ilustración 28 - Instalación React-router-dom

DESARROLLO E INSTALACIONES ADICIONALES

Antes de explicar las pantallas vamos a instalar unas librerías para facilitar la creación de los formularios. Una se encarga de ayudarnos con el desarrollo del formulario y otra para el uso de iconos. Primero instalaremos la librería para los formularios.

```
npm i react-hook-form
```

Ilustración 29 - Instalación librería formularios

Así quedaría un input con nuestra librería instalada.

```
<input
  id="password"
  name="password"
  type={showPassword ? "text" : "password"}
  autoComplete="current-password"
  required
  value={password}
  onChange={(e) => setPassword(e.target.value)}
  className="block w-full pl-8 sm:pl-10 pr-8 sm:pr-10 py-2 sm:py-3 md:py-3 border border-gray-300 rounded-lg focus:outline-none focus:ring-2 focus:ring-blue-500 focus:border-transparent"
  placeholder="password"
/>
```

Ilustración 30 - Input con librería

Seguido instalaremos la librería de los iconos.

```
npm install lucide-react
```

Ilustración 31 - Instalación iconos

El código para el mostrado de nuestros iconos es el siguiente y así se vería en nuestra interfaz.

```
<Mail className="h-4 w-4 sm:h-5 sm:w-5 text-gray-400" />
```

Ilustración 32 - Código para icono

Correo Electrónico

 tu@email.com

Contraseña





Ilustración 33 - Formulario con librerías

A continuación, vamos a ir explicando las pantallas que aparecen en la web y algunas de sus características

La primera pantalla que aparecerá será nuestro Login, en esta tendremos la opción de iniciar sesión una vez tengamos un usuario registrado, la opción de ir a la pantalla de registro de usuario y a la de recuperación de contraseña. También tendremos la funcionalidad para guardar nuestras credenciales, esto habrá sido creado con nuestro localStorage.

The image shows a login form titled "Iniciar sesión" (Log in) with the subtitle "Accede a tu cuenta para continuar" (Access your account to continue). The form is part of a page titled "CONTROLADOR DE TIENDA". It contains the following elements:

- A label "Correo Electrónico" (Email) above a text input field containing "tu@email.com".
- A label "Contraseña" (Password) above a password input field containing "*****".
- A checkbox labeled "Grábame" (Remember me) and a link "¿Olvidaste tu contraseña?" (Forgot your password?).
- A blue button labeled "Iniciar sesión" (Log in).
- A link at the bottom: "¿No tienes una cuenta? Regístrate aquí" (Don't you have an account? Register here).

Ilustración 34 - Contenido pantalla Login

La siguiente pantalla será la de registro que se encargará de comprobar que nuestro usuario ya no está registrado y lo almacenará en su correspondiente DB para tener control de usuarios

The image shows a registration form titled "Crear cuenta" (Create account) with the subtitle "Únete a nuestra plataforma" (Join our platform). The form is part of a page titled "CONTROLADOR DE TIENDA". It contains the following elements:

- A label "Apellido" (Last name) above a text input field containing "Tu apodo".
- A label "Correo Electrónico" (Email) above a text input field containing "tu@email.com".
- A label "Contraseña" (Password) above a password input field containing "*****".
- A label "Confirmar Contraseña" (Confirm Password) above a password input field containing "*****".
- A blue button labeled "Crear cuenta" (Create account).
- A link at the bottom: "¿Ya tienes una cuenta? Inicia sesión aquí" (Do you already have an account? Log in here).

Ilustración 35 - Contenido pantalla Registro

Por último, tendremos la pantalla de recuperación, esta se encargará de que al usuario pasarle un correo electrónico esta comprobará que nuestro usuario existe en la base de datos y nos llevará a una pagina para cambiar la contraseña



Ilustración 36 - Contenido pantalla recuperar contraseña

Esta será nuestra pantalla para realizar el cambio de contraseña, tendremos que introducir la contraseña nueva y confirmarla para que se realice la actualización de esta.

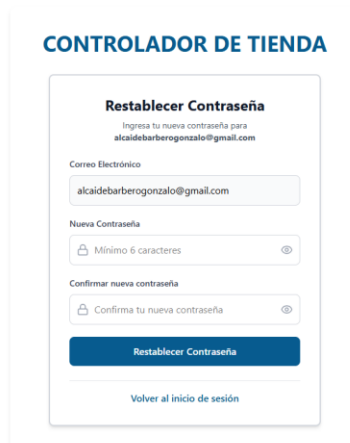


Ilustración 37 - Contenido pantalla actualización contraseña

Una vez explicadas estas pantallas tendremos que tener en cuenta que todas las peticiones que se van a realizar en nuestra web hacia la base de datos están realizadas con una llamada por un `await fetch`.

```
const response = await fetch("http://localhost:8000/api/login", {
```

Ilustración 38 - Código `await fetch`

También tendremos que tener en cuenta que la navegación completa de nuestra web se estará realizando con los componentes `link` y `navigate` proporcionados por la librería instalada anteriormente.

```
navigate(from, { replace: true })
```

Ilustración 39 - Navegación con `navigate`

```
<Link
  to="/auth/forgot-password"
  className="text-xs sm:text-sm text-[#075b8f] hover:text-[#012e4a] font-medium transition-colors"
>
  ¿Olvidaste tu contraseña?
</Link>
```

Ilustración 40 - Navegación con `Link`

Una vez estemos registrados e iniciemos sesión pasaremos a una pantalla que lleva el control de nuestras casas, en estas podremos ver si tenemos casa, añadirlas en caso de no tener o incluso actualizarlas y eliminarlas. También contará con un botón que llevara a los armarios de la casa.

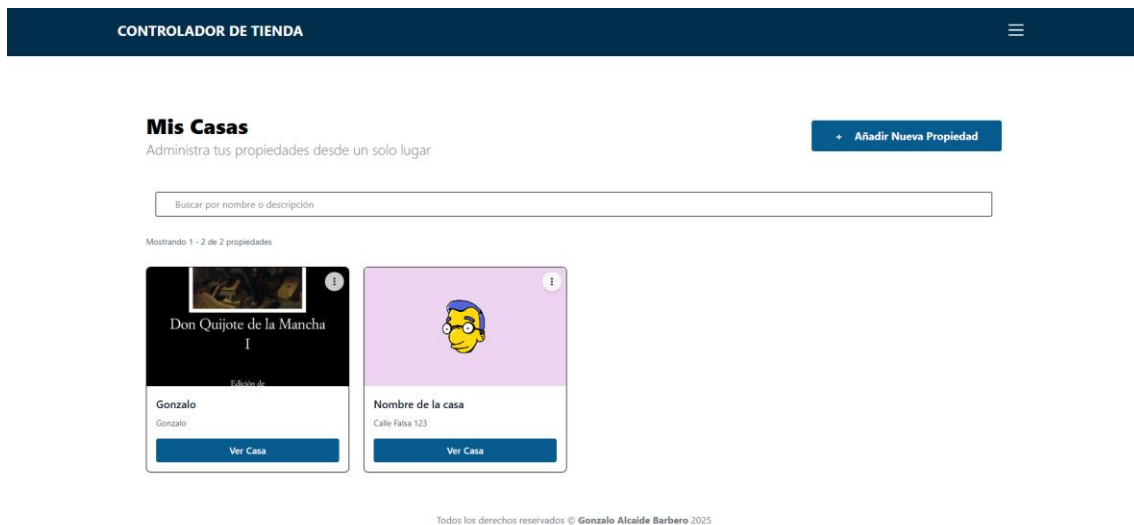


Ilustración 41 - Pantalla Menú Casas

La siguiente pantalla será la de creación de casas, esta se encargará de recoger todos los datos que necesitaremos para la inserción en la base de datos. Las respuestas se harán con un `await fetch` y usaremos la configuración de Cloudinary para conseguir la url de la imagen subida a esta plataforma y añadirla a nuestra petición. También contaremos con un botón para volver al menú de las casas.

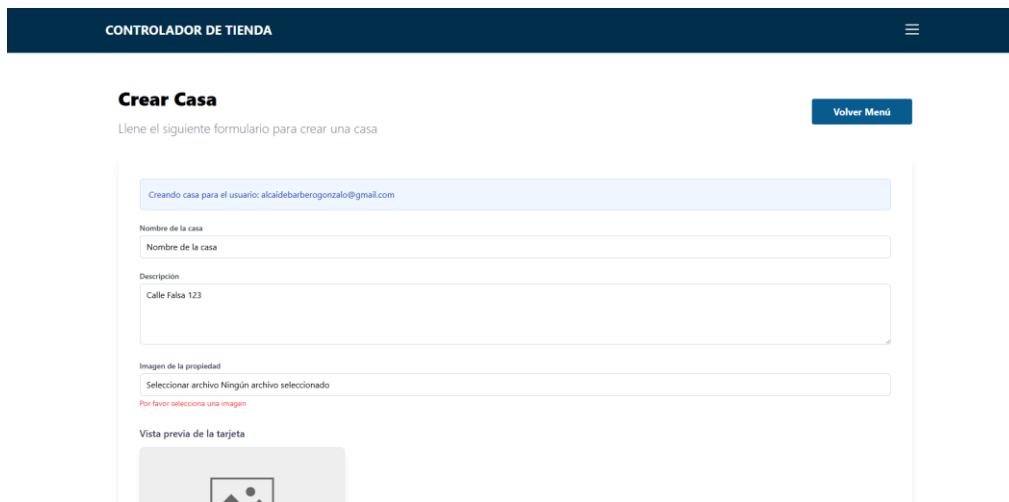


Ilustración 42 - Pantalla Creación Casas

Seguido de esta estará la pantalla de actualización de casas, esta hará la misma función que la de insertar cambiando que esta recibirá los datos de la casa para que modifiquemos solamente lo que queramos o modifiquemos todo. De la misma manera hemos usado `await fetch` para la petición de la base de datos y Cloudinary para el cambio de foto en caso de realizar un cambio de esta. Además, también tendremos el botón para volver para el menú

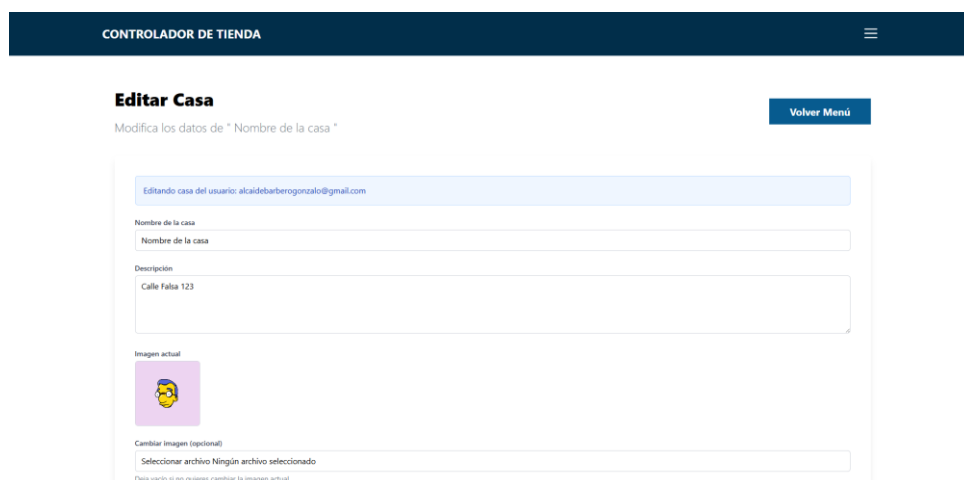


Ilustración 43 - Pantalla Actualización Casas

Dentro del círculo que tienen nuestras casas tenemos la opción de actualizar y eliminar. En caso de elegir la eliminación nos saltará un modal en la pantalla el cual nos indicará un mensaje para asegurar la eliminación de la casa. Una vez que se elimina el listado de las casas se actualiza de forma automática.

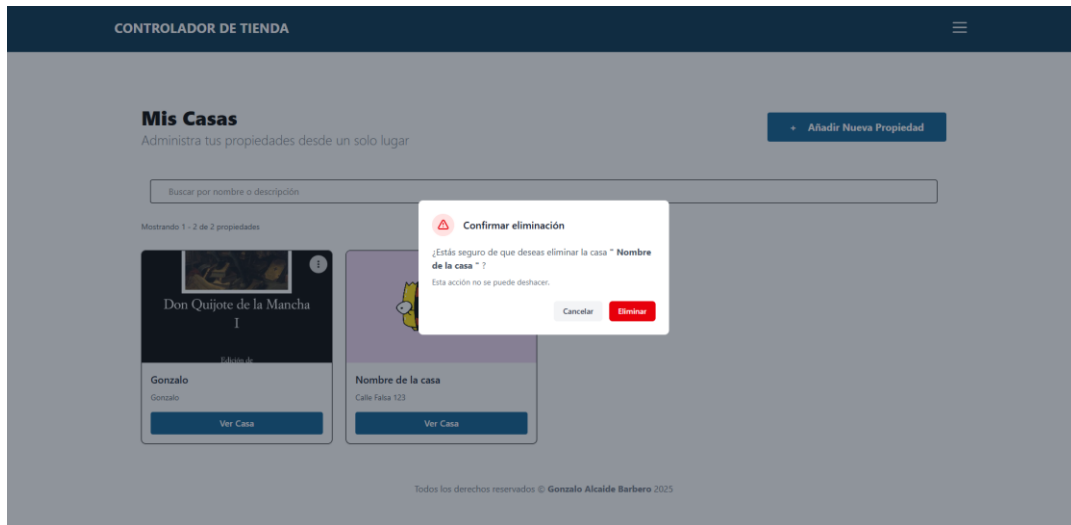


Ilustración 44 - Pantalla Eliminación Casas

Por otro lado todos los menús ya sean casas, armarios y objetos/productos tienen añadido una paginación en caso de tener muchos objetos y un buscador. El buscador se encarga de filtrar la lista recibida por nuestra petición y mostrar solo los datos que lo contengan.

```
const filteredHouses = useMemo(() => {
  if (!Array.isArray(allHouses)) {
    return []
  }

  return allHouses.filter((house) => {
    const matchesSearch =
      house.name?.toLowerCase().includes(searchTerm.toLowerCase()) ||
      house.descripcion?.toLowerCase().includes(searchTerm.toLowerCase())

    return matchesSearch
  })
}, [searchTerm, allHouses])
```

Ilustración 45 - Código para buscador

Por otro lado, para la paginación se encargará de mostrar un límite de 8 elementos antes de mostrar la paginación.

Cuando tengamos una casa creada podremos acceder al contenido de esta pulsando el botón “Ver Casa”, esta nos llevara a la pantalla de menú de armarios, donde podremos ver los armarios que tenga la casa, crearlos o actualizarlos. Todo esta esta recibido los datos con await fetch y las peticiones realizadas en el backend. Tambien tendremos una opción que nos generar un QR, este al escanearlo o pulsar en el nos llevara a una página donde nos mostrara los productos que hay dentro de dicho armario.

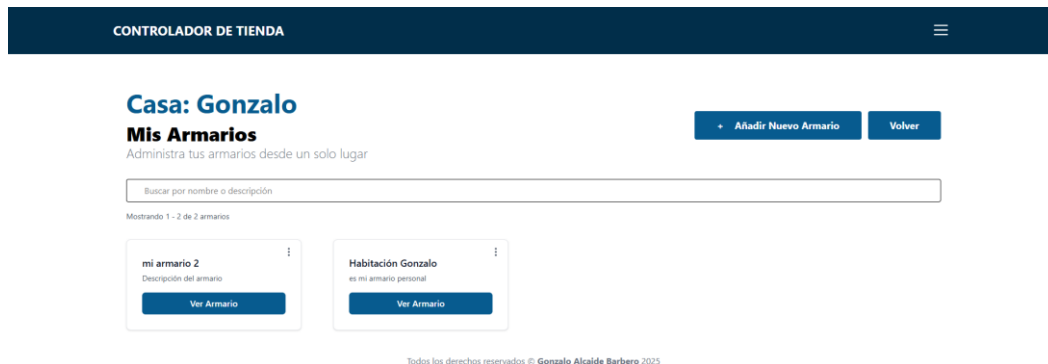


Ilustración 46 - Pantalla Menú Armarios

Para la implementación del QR hemos tenido que realizar la instalación de una librería, esta nos permitirá mostrar un QR y crear el contenido que nos devolverá al escanear

```
npm install qrcode.react
```

Ilustración 47 - Instalación QR

Este es el código para el mostrado del QR y para la creación de la url que devuelve al ser escaneado

```
<QRCodeSVG value={getQRContent()} size={200} level="M" includeMargin={true} />
```

Ilustración 48 - Mostrado QR

```
const getQRContent = () => {
  return `${window.location.origin}/qr/${idArmario}`
}
```

Ilustración 49 – URL del QR

Tendremos también al igual que en la pantalla de menú de casas nuestra pantalla para insertar los armarios, esta se encargará de coger los datos de los armarios y hacer una petición por await fetch para de esta manera insertar los armarios que queramos crear.

The screenshot shows a web application header with 'CONTROLADOR DE TIENDA' and a hamburger menu icon. The main heading is 'Crear un armario' with a 'Volver Menú' button. Below the heading is a subtext: 'Llena el siguiente formulario para crear un armario'. The form contains two input fields: 'Nombre del Armario' and 'Descripción'. Below these is a 'Vista previa del armario' section showing a preview of the cabinet with the name 'Nombre del Armario', description 'Descripción del armario', and a 'Ver Armario' button. At the bottom right of the form are 'Limpiar' and 'Crear un armario' buttons.

Ilustración 50 - Pantalla Creación Armarios

En la pantalla de actualización de armarios recibiremos los datos del menú para de esta manera modificar lo que queramos o modificar todos los valores. Una vez tenga todos los datos enviara una petición a nuestro backend por await fetch en la cual enviara los datos para que se actualicen.

The screenshot shows a web application header with 'CONTROLADOR DE TIENDA' and a hamburger menu icon. The main heading is 'Editar Armario' with a 'Volver Menú' button. Below the heading is a subtext: 'Modifica los datos de "mi armario 2"'. The form contains two input fields: 'Nombre del Armario' (pre-filled with 'mi armario 2') and 'Descripción'. Below these is a 'Vista previa del armario' section showing a preview of the cabinet with the name 'mi armario 2', description 'Descripción del armario', and a 'Ver Armario' button. At the bottom right of the form are 'Cancelar' and 'Actualizar Armario' buttons.

Ilustración 51 - Pantalla Actualización Armarios

En los tres círculos que tienen nuestros armarios tenemos la opción de actualizar he eliminar. En caso de elegir la eliminación nos saltara un modal en la pantalla el cual nos indicara un mensaje para asegurar la eliminación del armario. Una vez que se elimina el listado de los armarios se actualiza de forma automática.

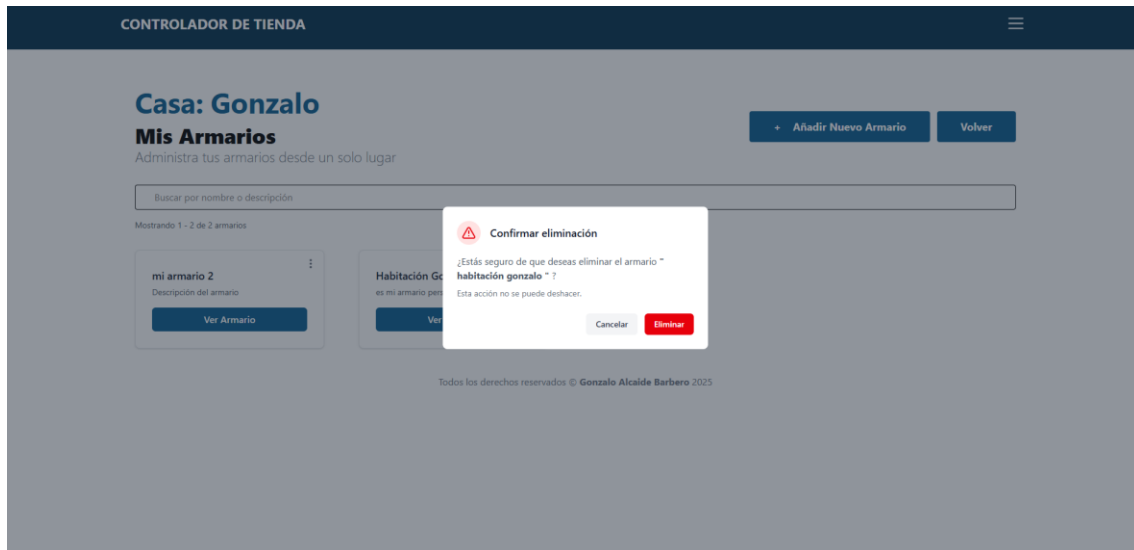


Ilustración 52 - Pantalla Eliminación Armarios

En caso contrario si pulsamos en código QR se nos abrirá un modal como este, en el tendremos la url para ver los objetos del armario seleccionado. Este modal nos permitirá llevarnos al escanear e incluso pulsando en él. Así damos opción a otros dispositivos a su fácil acceso.



Ilustración 53 - Pantalla QR Armarios

Esta será la vista que veremos al realizar el escaneo del QR o pulsar en el. Se nos mostrara los datos principales de nuestro armario accedido y posteriormente tendremos un listado de todos los productos que tiene dicho armario en su interior

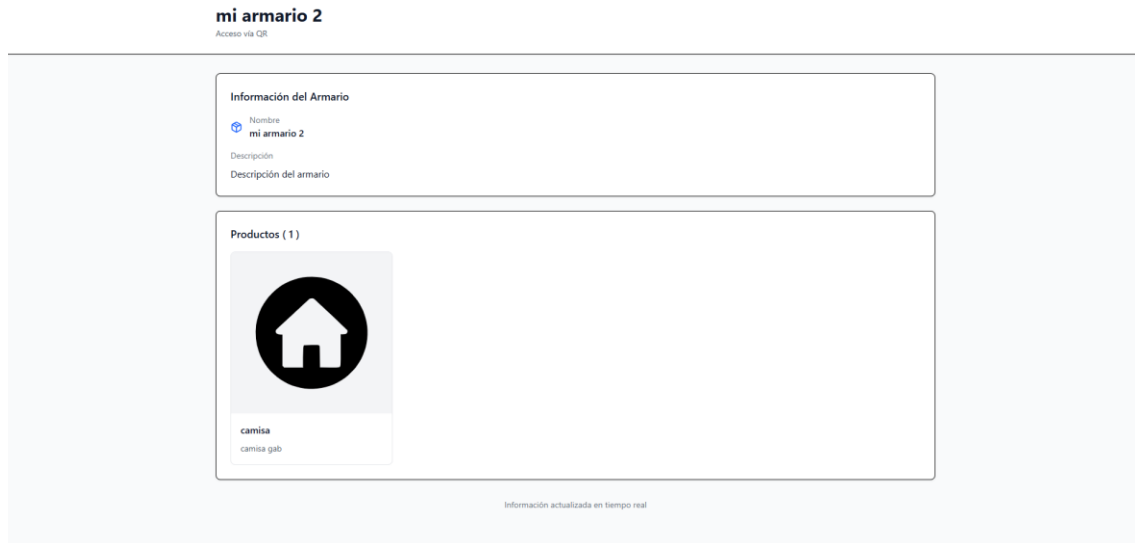


Ilustración 54 - Pantalla Armario QR

Una vez tengamos nuestro armario creado podremos ver lo que tiene dentro al “Ver Armario”. Esto nos llevara a la pantalla del menú de los objetos. En esta pantalla cargaran todos los datos que haya dentro de ese armario. Esto se realizará al igual que el resto de menús por una petición await fetch a una petición creada posteriormente en nuestro backend.

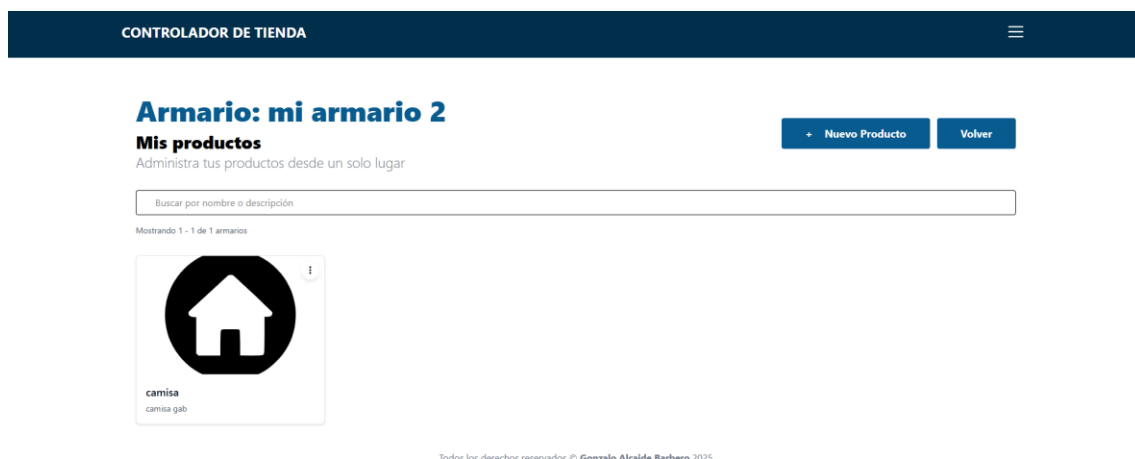


Ilustración 55 - Pantalla Menú Objetos

La siguiente será nuestra pantalla de creación de productos, esta será igual que la creación de casas ya que necesita los mismos tipos de datos. Por otro lado seguiremos usando la petición de Cloudinary para la subida de fotos y realizaremos la subida de los datos con nuestra funcionalidad await fetch

CONTROLADOR DE TIENDA☰

Crear productoVolver Menú

Llene el siguiente formulario para crear un producto

Creando producto para el usuario: alcaidebarberogonzalo@gmail.com

Nombre del producto

Nombre del producto

Descripción

Descripción del producto

Imagen del producto

Seleccionar archivo Ningún archivo seleccionado

Por favor selecciona una imagen

Vista previa de la tarjeta




Ilustración 56 - Pantalla Creación Objetos

Por último, estaría la pantalla de actualización de objetos, este recibiría los datos desde el menú por si el usuario quisiera cambiarlos todos por completo o simplemente cambiar algún detalle. Las peticiones serán realizadas llamando una función del backend con await fetch y en caso de cambiar la imagen usaremos la funcionalidad de Cloudinary.

CONTROLADOR DE TIENDA☰

Editar productoVolver al Armario

Modifica los datos de "camisa"

Editando producto del usuario: alcaidebarberogonzalo@gmail.com


Nombre del producto

camisa

Descripción

camisa gab

Imagen actual



Cambiar imagen (opcional)

Seleccionar archivo Ningún archivo seleccionado

Deja vacío si no quieres cambiar la imagen actual

Ilustración 57 - Pantalla Actualizar Productos

Nuestra última pantalla será la de eliminación, al pulsar en los 3 puntitos de la imagen de nuestros productos se nos abrirá un modal. Al pulsar en nuestro modal la opción que nos falta que será eliminación se nos mostrará un modal el cual nos avisará que la eliminación del producto se quiere hacer y que en caso de hacerlo no hay vuelta atrás.

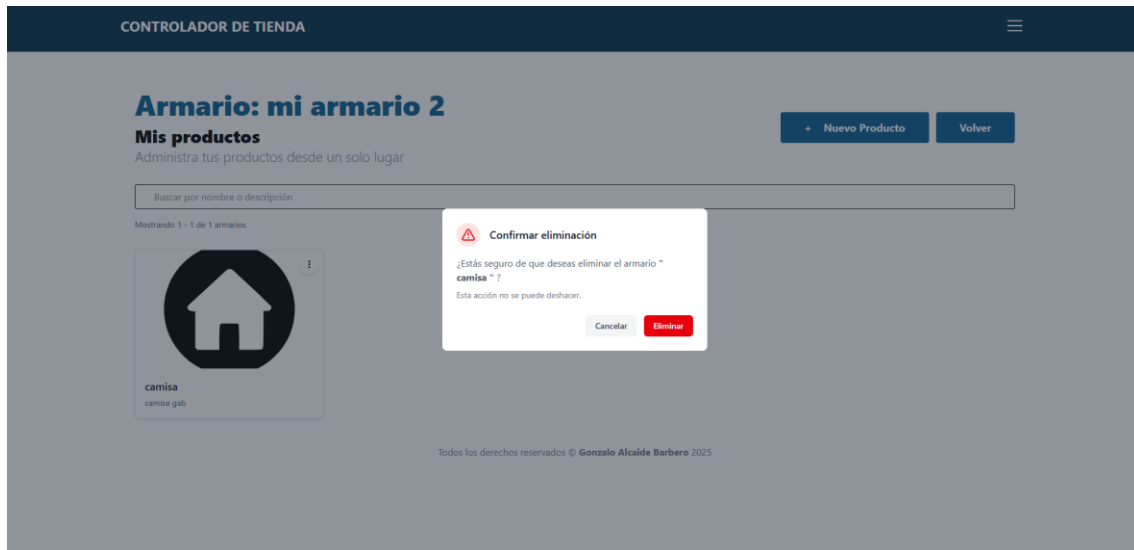


Ilustración 58 - Pantalla Eliminación Objeto

7. SEGURIDAD

- La seguridad es uno de los temas más importantes en este para para el usuario ya que este puede subir datos personales de ellos. La seguridad que se lleva a cabo es:
 - Autenticación: Esta se realiza con un correo electrónico que será único y con una contraseña que solo conocerá el usuario.
- Por otro lado, para la gestión de control de versiones utilizaremos Git. De esta manera tendremos una copia de seguridad a cada cambio que vayamos haciendo en nuestra aplicación. Y nos facilitaremos mucho el trabajo y nos aseguraremos contra posibles problemas o fallos.



Ilustración 59 – GitHub

8. PRUEBAS

PRUEBAS DE INICIO DE SESIÓN

CONTROLADOR DE TIENDA

Iniciar sesión

Accede a tu cuenta para continuar

Credenciales incorrectas

Correo Electrónico

alcaidebarberogonzalo@gmail.com

Contraseña

☐ Grabarme

¿Olvidaste tu contraseña?

Iniciar sesión

¿No tienes una cuenta? [Regístrate aquí](#)

Ilustración 60 - Inicio de sesión con credenciales erróneas

PRUEBAS REGISTRO USUARIO

CONTROLADOR DE TIENDA

Crear cuenta

Únete a nuestra plataforma

La contraseña debe tener al menos 6 caracteres

Apellido

Gonzalo

Correo Electrónico

alcaidebarberogonzalo@gmail.com

Contraseña

Confirmar Contraseña

Crear cuenta

¿Ya tienes una cuenta? [Inicia sesión aquí](#)

Ilustración 61 - Registro con contraseña muy pequeña

CONTROLADOR DE TIENDA

Crear cuenta

Únete a nuestra plataforma

Las contraseñas no coinciden

Apellido

GONZALO ALCAIDE BARBERO

Correo Electrónico

alcaidebarberogonzalo@gmail.com

Contraseña

Confirmar Contraseña

Crear cuenta

¿Ya tienes una cuenta? [Inicia sesión aquí](#)

Ilustración 62 - Registro con contraseñas distintas

CONTROLADOR DE TIENDA

Crear cuenta

Únete a nuestra plataforma

Error en la validación de los datos

Apellido

GONZALO ALCAIDE BARBERO

Correo Electrónico

alcaidebarberogonzalo@gmail.com

Contraseña

Confirmar Contraseña

Crear cuenta

¿Ya tienes una cuenta? [Inicia sesión aquí](#)

Ilustración 63 - Registro con usuario existente

PRUEBAS RECUPERAR CONTRASEÑA



Ilustración 64 - Recuperación de cuenta no existen

Ilustración 65 - Recuperación de contraseña exitosa

PRUEBAS PANTALLA MENU CASAS

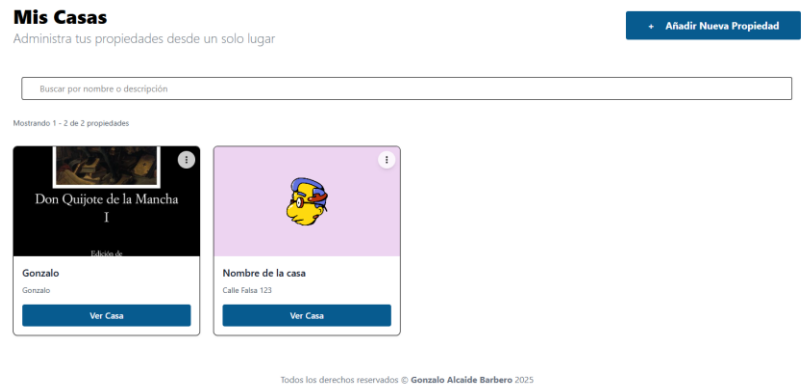


Ilustración 66 - Pantalla de casas con datos

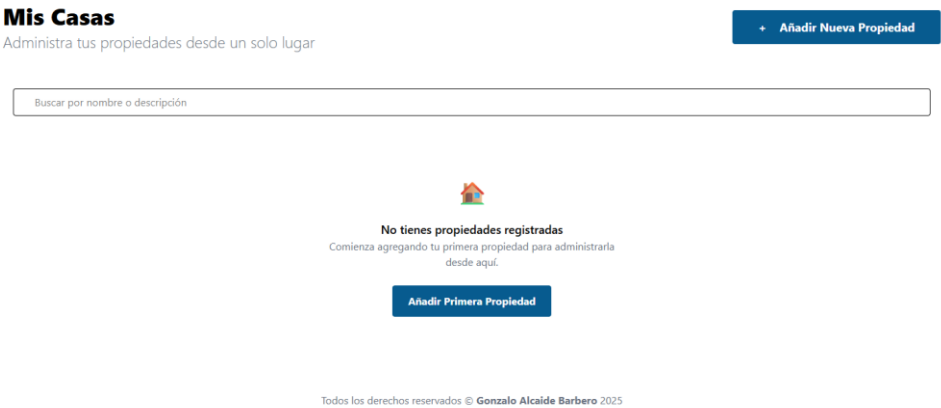


Ilustración 67 - Pantalla de casas sin datos

PRUEBAS ACTUALIZACIÓN CASA

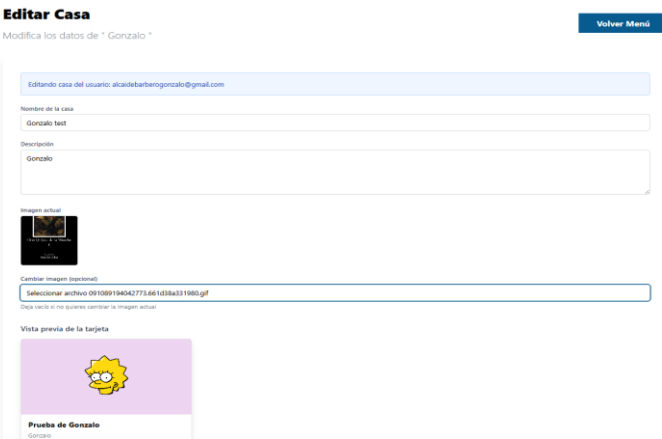


Ilustración 68 - Actualización de casas



Ilustración 69 - Actualización realizada correctamente

PRUEBAS ELIMINACIÓN CASA

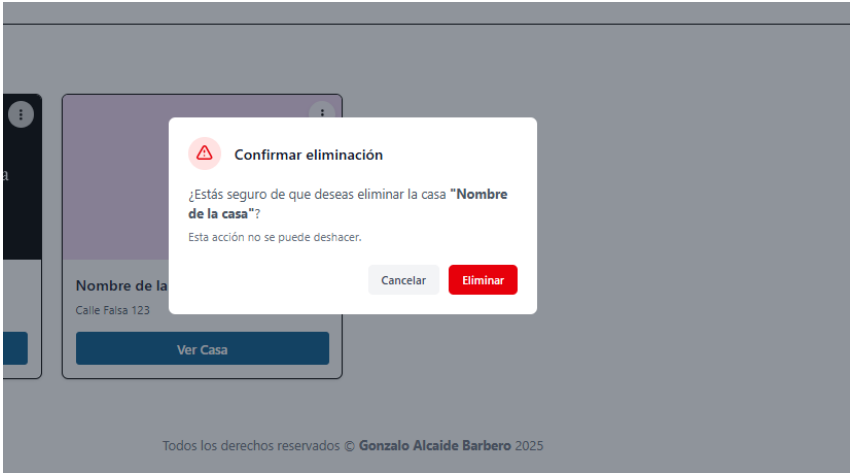


Ilustración 70 - Eliminación de casas



Ilustración 71 - Eliminación realizada

PRUEBAS INSERCCIÓN CASA

Crear Casa

Llena el siguiente formulario para crear una casa

[Volver Menu](#)

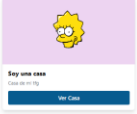
Creando casa para el usuario: alcaidebarbero@gnail.com

Nombre de la Casa
Soy una casa

Descripción
Casa de mi tfg

Imagen de la propiedad
Seleccionar archivo (910381184042773.661d3ba337900.gif)

Vista previa de la tarjeta



[Ver Casa](#)

[Guardar Casa](#)

Ilustración 72 - Inserción casas

Mis Casas

Administra tus propiedades desde un solo lugar

Buscar por nombre o descripción

Mostrando 1-2 de 2 propiedades



Ilustración 73 - Casa insertada

PRUEBAS PANTALLA MENU ARMARIOS

Casa: Gonzalo

Mis Armarios

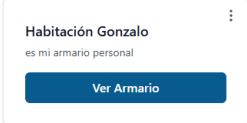
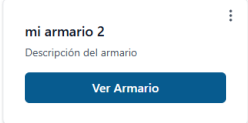
Administra tus armarios desde un solo lugar

[+ Añadir Nuevo Armario](#)

[Volver](#)

Buscar por nombre o descripción

Mostrando 1 - 2 de 2 armarios



Todos los derechos reservados © Gonzalo Alcaide Barbero 2025

Ilustración 74 - Pantalla de armarios con datos

Casa: Nombre de la casa


Mis Armarios

Administra tus armarios desde un solo lugar

[+ Añadir Nuevo Armario](#)

[Volver](#)

Buscar por nombre o descripción



No tienes armarios registrados

Comienza agregando tu primer armario para administrarlo desde aquí.

[Añadir Primer Armario](#)

Todos los derechos reservados © Gonzalo Alcaide Barbero 2025

Ilustración 75 - Pantalla de armarios sin datos

PRUEBAS INSERCIÓN ARMARIOS

Crear Armario

Llena el siguiente formulario para crear un armario

Volver Menú

Nombre del Armario

Nuevo armario

Descripción

es mi nuevo armario

Vista previa del armario

Nuevo armario

es mi nuevo armario

Ver Armario

Limpiar

Crear Armario

Todos los derechos reservados © Gonzalo Alcáide Barbero 2025

Ilustración 76 - Inserción de armario

Casa: Soy una casa

Mis Armarios

Administra tus armarios desde un solo lugar

+ Añadir Nuevo Armario

Volver

Buscar por nombre o descripción

Mostrando 1-1 de 1 armarios

Nuevo armario

es mi nuevo armario

Ver Armario

Ilustración 77 - Armario insertado

PRUEBAS ACTUALIZACIÓN ARMARIOS

Editar Armario

Modifica los datos de "Nuevo armario"

Volver Menú

Nombre del Armario

Nuevo armario

Descripción

es mi nuevo armario

Vista previa del armario

Nuevo armario

es mi nuevo armario

Ver Armario

Cancelar

Actualizar Armario

Ilustración 78 - Actualización de armario

Casa: Soy una casa

Mis Armarios

Administra tus armarios desde un solo lugar

+ Añadir Nuevo Armario

Volver

Buscar por nombre o descripción

Mostrando 1 - 1 de 1 armarios

Nuevo armario 2

es mi nuevo armario

Ver Armario

Todos los derechos reservados © Gonzalo Alcáide Barbero 2025

Ilustración 79 - Armario actualizado

35

PRUEBAS PANTALLAS QR

Nuevo armario 2

Acceso vía QR

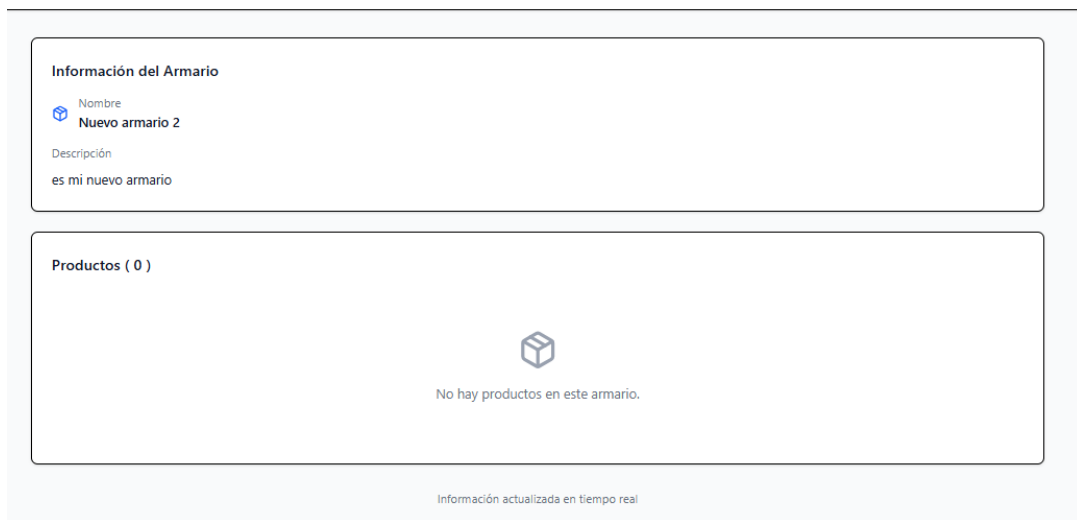


Ilustración 80 - Pantalla QR sin datos

mi armario 2

Acceso vía QR

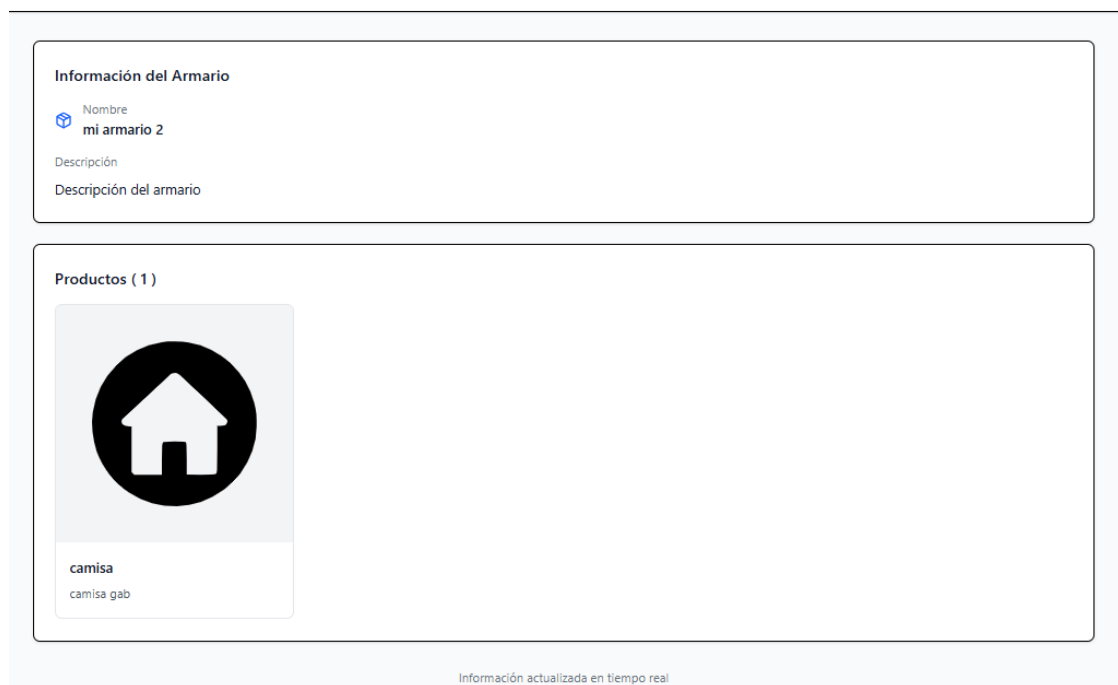


Ilustración 81 - Pantalla QR con datos

PRUEBAS ELIMINAR ARMARIO

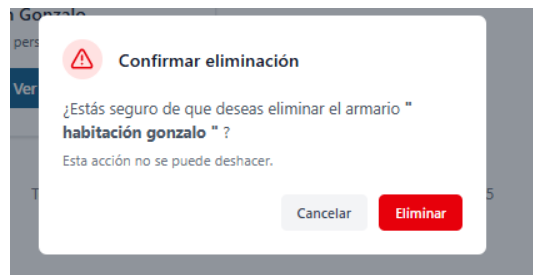


Ilustración 82 - Eliminación de armario

Casa: Prueba de Gonzalo

Mis Armarios

Administra tus armarios desde un solo lugar



Ilustración 83 - Armario eliminado

PRUEBAS MENU OBJETOS

Armario: mi armario 2

Mis productos

Administra tus productos desde un solo lugar



Todos los derechos reservados © Gonzalo Alcalde Barbero 2025

Ilustración 84 - Menú objetos con datos

Armario: Habitación Gonzalo

Mis productos

Administra tus productos desde un solo lugar



Todos los derechos reservados © Gonzalo Alcalde Barbero 2025

Ilustración 85 - Menú objetos sin datos

PRUEBAS INSERCIÓN OBJETOS

Crear producto

Llene el siguiente formulario para crear un producto

[Volver Menú](#)

Creando producto para el usuario: alcaidebarberogonzalo@gmail.com

Nombre del producto

Ana Maria


Descripción

profesora de ~~css~~

Imagen del producto

Seleccionar archivo Harry_Potter_V_La_Piedra_Filosofal.jpg

Vista previa de la tarjeta



Ana Mari
profesora de informática

Crear producto

Ilustración 86 - Inserción de objetos

Armario: mi armario 2

Mis productos

Administra tus productos desde un solo lugar

[+ Nuevo Producto](#)[Volver](#)

Mostrando 1 - 2 de 2 armarios



Todos los derechos reservados © Gonzalo Alcaide Barbero 2025

Ilustración 87 - Objetos insertados

PRUEBAS ACTUALIZACIÓN OBJETOS

Editar producto

Modifica los datos de " Ana María "

[Volver al Armario](#)

Editar producto del usuario: alcaidebarberogonzalo@gmail.com

Nombre del producto

Ana María InsidePc

Descripción

profesora de css

Imagen actual

Cambiar imagen (opcional)

Seleccionar archivo Ningún archivo seleccionado

Deja vacío si no quieres cambiar la imagen actual

Vista previa de la tarjeta

Ilustración 88 - Actualización producto

Armario: mi armario 2

Mis productos

Administra tus productos desde un solo lugar

[+ Nuevo Producto](#)[Volver](#)

Buscar por nombre o descripción

Mostrando 1 - 2 de 2 armarios

Todos los derechos reservados © Gonzalo Alcaide Barbero 2025

Ilustración 89 - Producto actualizado

PRUEBAS ELIMINACIÓN OBJETOS

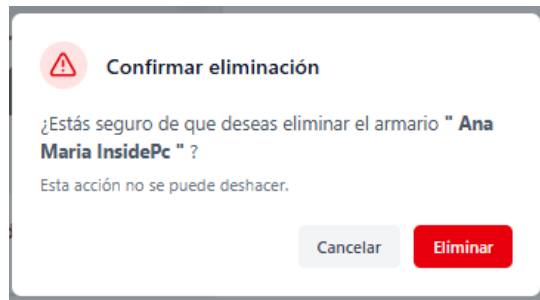


Ilustración 90 - Eliminación de objeto

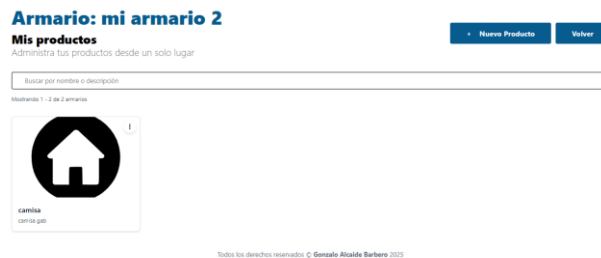


Ilustración 91 - Objeto eliminado

PRUEBAS BUSQUEDA EN MENÚS

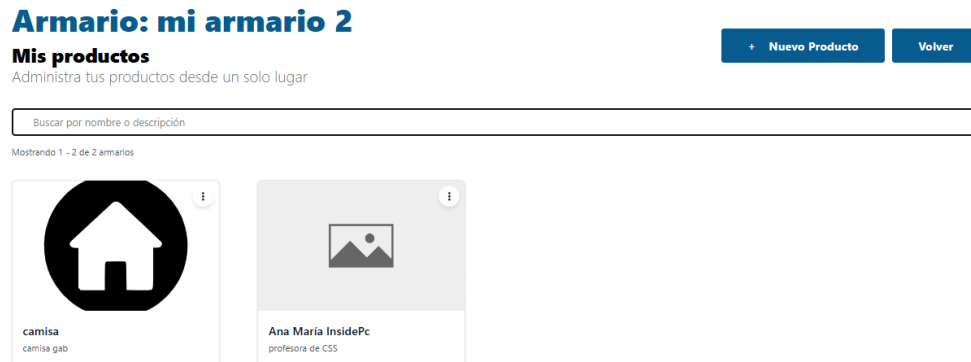


Ilustración 92 - Listado sin búsqueda

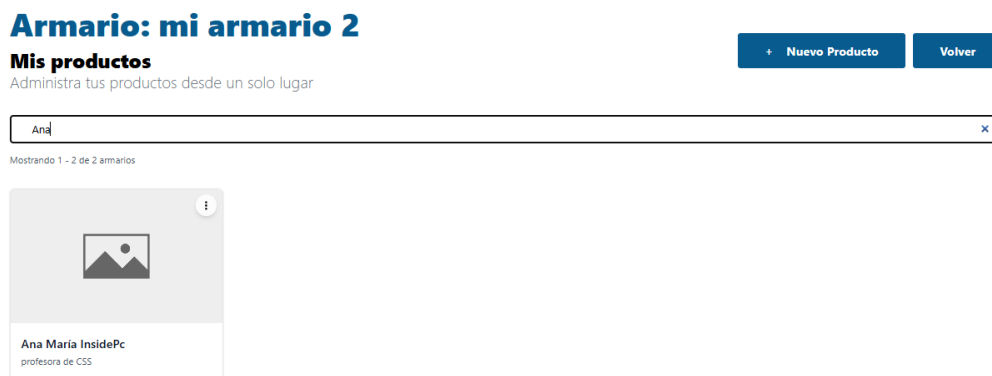


Ilustración 93 - Listado con búsqueda

9. RECURSOS

Los recursos necesarios para la utilización de la web son los siguientes:

a) RECURSOS HARDWARE

- Para la realización del proyecto hemos necesitado el siguiente hardware
 - Portátil
 - Pantalla
 - Conexión a internet
 - Disposiciones de móviles (revisión del responsable)
 - Servidor de hosting.

b) RECURSOS SOFTWARE

- Para el desarrollo de la web hemos necesitado.
 - Sistema operativo.
 - Windows 11.
 - Navegador.
 - Para poder ver nuestro desarrollo en tiempo real.
 - IDE.
 - Visual Studio Code.
 - Composer.
 - Para la instalación de laravel.
 - XAMPP.
 - Para montar nuestro proyecto antes de subirlo al hosting.
 - Postman.
 - Testing de las peticiones realizadas por laravel..
 - Figma.
 - Diseño de interfaz y planteamientos.

c) PRESUPUESTO

RECURSOS	TIPO	HORAS	€/HORAS	TOTAL
Desarrollador	Personal	300	15€	4500€
Hosting	Software	10	7€	70€
Autenticación	Software	5	5€	25€
Clouinary	Software	0€	0€	0€
Total				4595€

Ilustración 94 - Tabla presupuesto

10. CONCLUSIÓN

a) CONSECUCIÓN DE LOS OBJETIVOS

Los objetivos principales del proyecto se han cumplido por completo, incluyendo alguna mejora a estos mismos.

En la siguiente tabla se mostrarán uno a uno los objetivos y como se han cumplido.

OBJETIVOS	
Control de almacenamiento	Se ha cumplido y se ha conseguido un control total de la gestión.
Creación de QR en cada armario	Se ha cumplido y además se ha añadido un lector QR exclusivo para la aplicación
Buscador de elementos	Se ha cumplido y se ha añadido que el usuario pueda acceder a ese armario en concreto
Borrar cualquier dato	Se ha cumplido y se ha añadido la posibilidad de poder borrar cualquier dato a pesar de tener datos dentro y su respectivo aviso de perdida de datos

Ilustración 95 - Tabla objetivos

b) PROBLEMAS ENCONTRADOS

Mientras que implementaba las pruebas de QA a la vez que realizaba el código me he encontrado con un error que no habíamos tenido en cuenta en la base de datos. Este constaba en que si el usuario quería borrar una Casa o un Armario y esta tenía datos dentro no le permitía borrarlo debido a los datos que tenía dentro. En un principio no me pareció mala idea el que el usuario no pudiera borrar dichos elementos si estos tenían datos dentro, pero me di cuenta que si el usuario tiene muchos datos y tiene que borrarlos todos para poder borrar el elemento que quiere se le pudo acceder tedioso por ello he modificado la base de datos realizando un enlace en cascada a la hora de eliminar los datos.

- Esta modificación la hemos realizado por la interfaz de MySQL. Tendremos que hacer que también se actualice por cascada para corrección de fallos en la DB (el fallo consistía en que si el usuario borraba una casa sin borrar antes los objetos que tenía dentro de los armarios estos se quedaban almacenados y por tanto al buscar objetos podías encontrarlos cuando no deberían existir)



Ilustración 96 - Clave Foreign arreglada

- DB con las relaciones cambiada por completo:

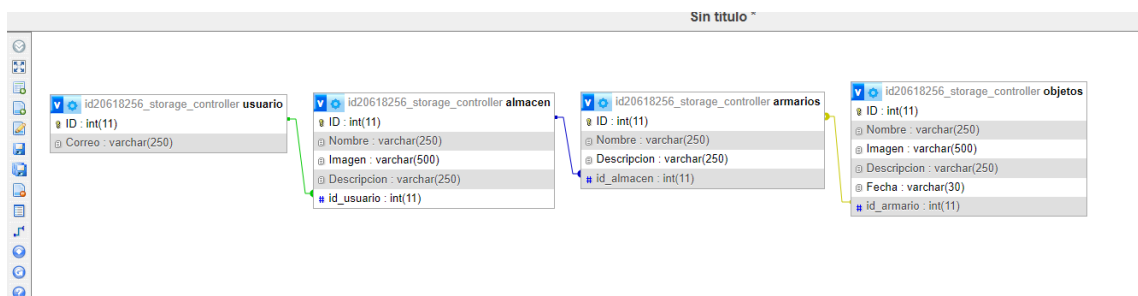


Ilustración 97 - Enlace DB arreglado

c) MEJORAS

Las mejoras a implementar serian crear un apartado dentro de la aplicación que nos permita compartir nuestras casas o almacenes con otros usuarios que estén registrados. La manera de hacerlo sería creando dentro de nuestra DB otra tabla en la cual asignaremos que el usuario pueda tener varias casas o almacén asignado y de esta manera podamos hacer que otro usuario tenga acceso a nuestros datos.

(Esta funcionalidad no se ha podido realizar por falta de tiempo y mal planteamiento de la DB)

11. BIBLIOGRAFÍA

- **DATOS INTEGRACIÓN DB**

- <https://cloudinary.com/documentation>
- https://www.youtube.com/watch?v=PzsXTs0uPu8&ab_channel=JLuisDev
- https://www.youtube.com/watch?v=xx8mLgWkCIY&ab_channel=CodersFree
- <https://www.geeksforgeeks.org/php/laravel/>
- <https://laravel.com/docs/5.1/>

- **DATOS INVESTIGACIÓN**

- <https://laravel.com/docs/11.x>
- https://www.youtube.com/watch?v=eLI8c_NtkBk
- <https://es.react.dev/>
- <https://tailwindcss.com/>
- https://www.youtube.com/results?search_query=pagination+react
- https://www.youtube.com/watch?v=Rd0dshSLCFw&ab_channel=Tutored
- https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Frameworks_libraries/React_getting_started
- <https://react-hook-form.com/>

12. ANEXO

a) MANUAL DE USUARIO