

MEMÒRIA

Nom del projecte: Phoenix

Autors: Joel García y Rubén Redondo

Data: 20-04-2023

Índice

1. Objetivos.....	2
2. Introducció.....	2
3. Descripción del proyecto.....	3
3.1 Front-End:.....	3
3.1.1 HTML.....	3
3.1.2 CSS.....	4
3.1.3 JavaScript.....	8
3.2 Back-End:.....	12
3.2.1. Javascript:.....	12
3.2.2. Base de datos.....	13
3.3 Funcionalidad:.....	18
3.3.1 Diagrama de casos de uso:.....	18
3.3.2 Restricciones de la página:.....	19
4. Conclusiones.....	22
4.1 Conclusión del proyecto:.....	22
4.2 Propuestas de mejora:.....	23
5. Fuentes.....	24
6. Anexos.....	28

1. Objectivos

Phoenix es una innovadora plataforma web que combina la pasión por la automoción y lo exclusivo. El objetivo principal al realizar este proyecto es utilizar todos los conocimientos aprendidos durante el curso y aplicarlos en un proyecto novedoso que brinda una experiencia única que permite descubrir el automóvil de tus sueños de una manera totalmente novedosa.

2. Introducción

Phoenix ha sido un proyecto desarrollado por dos estudiantes de Programación. Phoenix se basa en una página web desarrollada tanto por backend como por frontend y totalmente enlazada con una propia base de datos creada exclusivamente para la web. La bd ha sido creada y desarrollada en Supabase la cual tiene como servidor PostgreSQL.

El apartado de Front-End ha sido creado y maquetado en HTML5, y hemos conseguido una buena apariencia visual a través de CSS, JavaScript y la utilización de frameworks y bibliotecas como Bootstrap, SweetAlert, MDB bootstrap entre otras.

En cuanto al Back-End, ha sido totalmente desarrollado en un mismo lenguaje, JavaScript. En el apartado de backend podrás

encontrar todo el tema de funcionalidad, accesibilidad y desarrollo de la usabilidad de la web. También podrás encontrar el mapping de la base de datos, la cual es PostgreSQL.

3.Descripción del proyecto

3.1 Front-End:

3.1.1 HTML

Hemos utilizado el lenguaje HTML5 para poder maquetar y estructurar todas las páginas del sitio web. Además de esto, hemos utilizado html para el marcado de texto, la incorporación de elementos multimedia, la creación de enlaces y finalmente para la creación de formularios interactivos.

También hemos utilizado la biblioteca **Bootstrap**:

Bootstrap es una biblioteca/framework que sirve para el desarrollo front-end, y su funcionalidad es proporcionar estilos CSS y componentes JS que permite a los desarrolladores diseñar rápidamente interfaces de usuario atractivas y compatibles con diferentes dispositivos y tamaños de pantalla.

En cuanto a implementaciones de Bootstrap, he utilizado **MDBbootstrap**, que es una implementación de Bootstrap que amplía y mejora las funcionalidades y componentes ofrecidos por el framework Bootstrap.

Tanto Bootstrap y MDBbootstrap lo hemos utilizado con la importación CDN en la etiqueta `<head></head>` del HTML:

```
<!-- Bootstrap -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-KK94CHFLLe+nY2dmCWGMq91rCGa5gtU4mk92HdvYe+M/SXH381p5ILy+M" crossorigin="anonymous">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ENjd04Dr2bkBIFxQpoeTz1H1Cje39w44j0KdF19UigI4dd0Q3GNS7N7KfAdVQ5Ze" crossorigin="anonymous"></script>

<!-- MDB -->
<link href="https://cdnjs.cloudflare.com/ajax/libs/mdb-ui-kit/6.3.1/mdb.min.css" rel="stylesheet">
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-KK94CHFLLe+nY2dmCWGMq91rCGa5gtU4mk92HdvYe+M/SXH381p5ILy+M" crossorigin="anonymous">
```

3.1.2 CSS

Hemos utilizado el lenguaje CSS para definir la apariencia y el diseño visual del documento HTML. Ha sido utilizado para controlar el aspecto y el formato de los elementos en una página web, como el color, la tipografía, el espaciado, los tamaños, los bordes y otros aspectos visuales. Específicamente nosotros lo hemos utilizado para:

El control visual (colores, fondos...), para la separación de la estructura HTML y el diseño, para la reutilización de estilos (con clases), para la adaptabilidad y finalmente para la eficiencia de carga, ya que separando el diseño a archivos externos CSS se logra una mayor eficiencia de carga en las webs.

Principalmente el diseño nosotros lo hemos implementado a través de clases:

```
.inputConfig{
padding-right: 55%;
padding-top: 10%;
flex: 1;
margin-right: 20px;
}
.seleccionador{
width: 700px;
height: 550px;
padding-left: 10%;
padding-top: 0%;
}
select{
width: 150%;
}
.container-contact1 {
display: flex;
justify-content: space-between;
align-items: flex-start;
}
```

Pero también le hemos implementado diseño a id`s e incluso a etiquetas:

Etiquetas:

```
5 table td,  
7 ✓ table th {  
8   text-overflow: ellipsis;  
9   white-space: nowrap;  
9   overflow: hidden;  
1  }  
2  thead th,  
3  ✓ tbody th {  
4    color: ■ #fff;  
5  }  
6  ✓ tbody td {  
7    font-weight: 500;  
8    color: ■ rgba(255,255,255,.65);  
9  }  
9
```

ID #:

```
#fotoprim{  
width: 100px;  
}
```

También aclarar, que en nuestro proyecto hemos utilizado animaciones, transiciones y pseudoclasses, aquí tenéis unos ejemplos:

Pseudoclasses:

```
.nombres input::placeholder {  
  color: white;  
  font-size: 0.7em;  
  font-style: italic;  
}
```

```
.button:hover {  
  background: #34974d;  
}
```

Transiciones/keyframes:

```
.input1:focus + .shadow-input1 {  
  -webkit-animation: anim-shadow 0.5s ease-in-out forwards;  
  animation: anim-shadow 0.5s ease-in-out forwards;  
}  
@-webkit-keyframes anim-shadow {  
  to {  
    box-shadow: 0px 0px 80px 30px;  
    opacity: 0;  
  }  
}  
  
@keyframes anim-shadow {  
  to {  
    box-shadow: 0px 0px 80px 30px;  
    opacity: 0;  
  }  
}
```


3.1.3 JavaScript

En el desarrollo Front-End también hemos decidido implementarlo con JS ya que a diferencia de HTML y CSS, que se centran en la estructura y el estilo de una página web, JavaScript permite la creación de aplicaciones web dinámicas y en tiempo real.

Específicamente nosotros hemos utilizado JS en frontend por las siguientes razones:

Por la interactividad y por las animaciones y efectos visuales que podemos obtener.

Aquí os dejamos algunos ejemplos:

Este código decidimos implementarlo ya que queríamos que dependiendo de qué opción del option estuvieras, queríamos que saliera una imagen diferente juntamente con un card y una descripción totalmente diferente y adaptada al option del select correspondiente.

```
//neumaticos
const neumaticos = await Rueda.getAll()
let neumaticosOptions = `<option value="" selected disabled>-- Select Tires --</option>`
for(const rueda of neumaticos){
  neumaticosOptions += `<option value="${rueda.id}">${rueda.pulgadas} - ${rueda.fabricante}</option>`
}

document.querySelector('#tireSelect').innerHTML = neumaticosOptions

const selectNeumaticos = document.querySelector('#tireSelect');
selectNeumaticos.addEventListener("change", async(event)=>{
  let id = event.target.value;
  const neumaticoApintar = await Rueda.getById(id)
  valorRueda = id
  console.log(neumaticoApintar);
  selectedImage.src = `../img/${neumaticoApintar.img}.png`
  document.querySelector('#addCard').innerHTML=`
    <div class="cardSetting shadow">
      <div class="card2">
        <h5 class="text-white primerH1">By<p>Phoenix</p></h5>
        <h5 class="text-white">Inches<p>${neumaticoApintar.pulgadas}</p></h5>
        <h5 class="text-white">Measures<p>${neumaticoApintar.medidas}</p></h5>
        <h5 class="text-white">Noise<p>${neumaticoApintar.ruido} </p></h5>
        <h5 class="text-white">Manufacturer<p>${neumaticoApintar.fabricante}</p></h5>
      </div>
    </div>`;
})
```

También hemos utilizado JS para la implementación de clases:

En este código lo que conseguimos es que cuando hagamos click en send, coga el valor de la clase swal2-html-container y le añade la clase text-white, la cual es una clase de Bootstrap.

```
document.querySelector('#send').addEventListener("click",()=>{
  document.querySelector('.swal2-html-container').classList.add("text-white");
})
```

3.1.4 SweetAlert

SweetAlert es una biblioteca JavaScript que proporciona una manera elegante y personalizable de mostrar mensajes de alerta y diálogos modales en una página web. A diferencia de las alertas y cuadros de diálogo predeterminados del navegador, SweetAlert ofrece una interfaz más atractiva y flexible para presentar mensajes de confirmación, advertencia, éxito, error y otros tipos de notificaciones al usuario.

Esta biblioteca también ha sido importada a través de CDN:

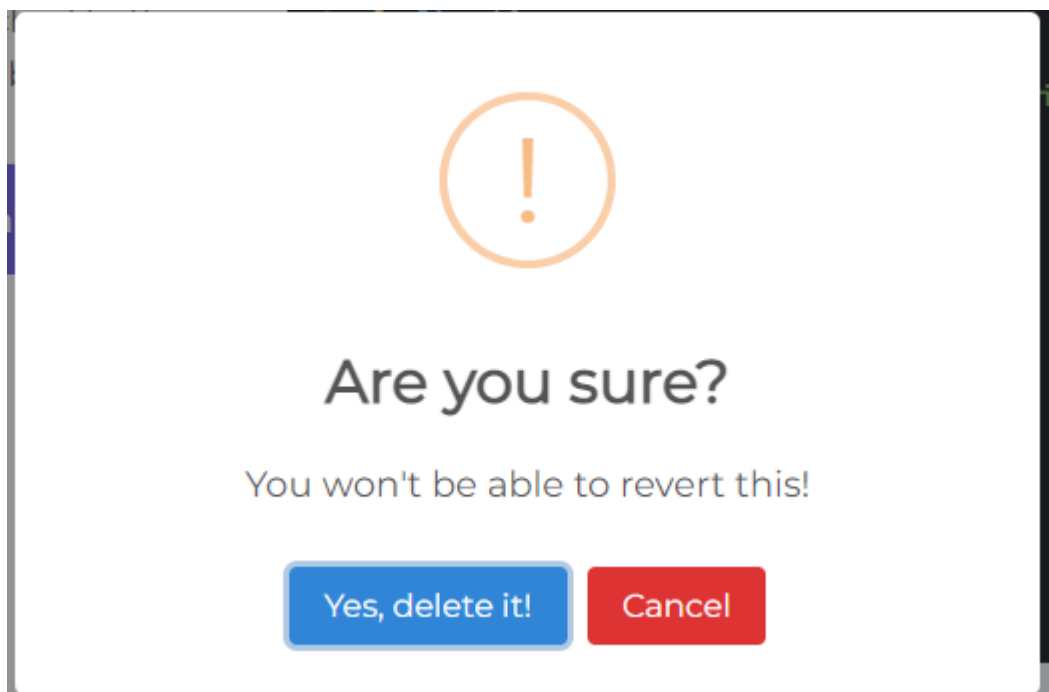
```
<!--SweetAlert-->  
<link href="https://cdn.jsdelivr.net/npm/@sweetalert2/theme-dark@4/dark.css" rel="stylesheet">  
<script src="https://cdn.jsdelivr.net/npm/sweetalert2@11/dist/sweetalert2.min.js"></script>
```

Y aquí os dejamos algunos ejemplos de lo que realmente es SweetAlert:

Esto sería el código a implementar entre una etiqueta `<script></script>`:

```
Swal.fire({
  title: 'Are you sure?',
  text: "You won't be able to revert this!",
  icon: 'warning',
  showCancelButton: true,
  confirmButtonColor: '#3085d6',
  cancelButtonColor: '#d33',
  confirmButtonText: 'Yes, delete it!'
}).then((result) => {
  if (result.isConfirmed) {
    Swal.fire(
      'Deleted!',
      'Your file has been deleted.',
      'success'
    )
  }
})
```

Y este sería el resultado:



Cabe destacar que tanto los mensajes como los botones y los símbolos pueden ser personalizados al gusto del usuario.

3.2 Back-End:

3.2.1. Javascript:

- El código javascript utilizado para el backend consiste en utilizar unas vistas y unos componentes que se irán inyectando según la ventana a la cual quieras ir.

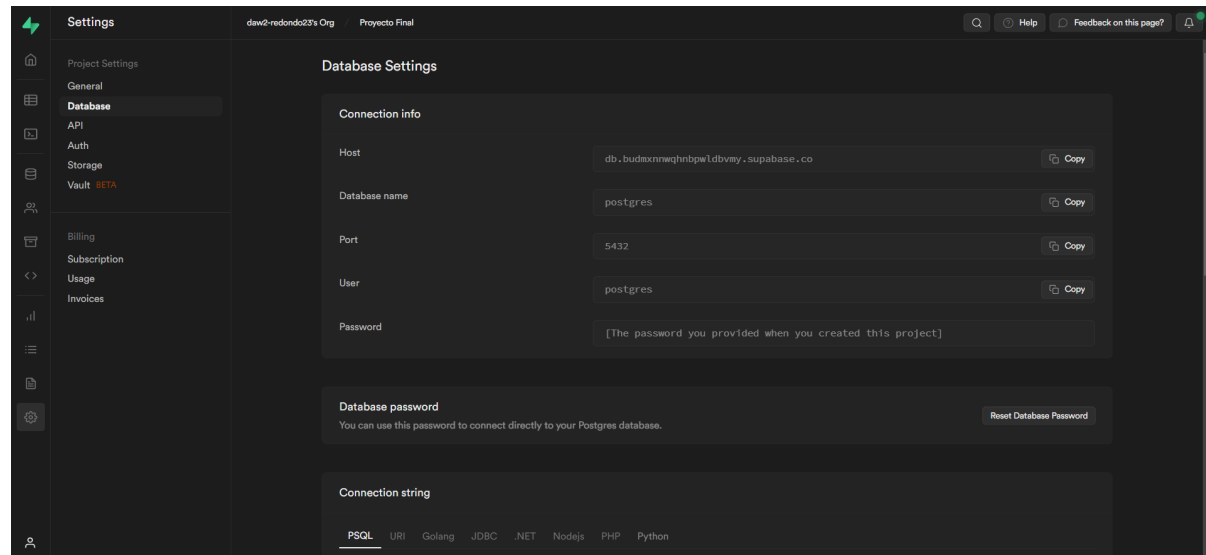
Foto de las vistas y componentes

- También hay un sistema de routing el cual sirve para moverte entre las diferentes páginas a través de la url, este sistema carga la ventana y su funcionalidad según la vista que le indiquemos, en caso de haber un error debido a que no existe una ruta con el nombre que se escribe en la url pintará por pantalla una plantilla error404.

Foto del router

3.2.2. Base de datos

- Para realizar este proyecto hemos usado Supabase que es una plataforma de desarrollo de aplicaciones de código abierto que proporciona una infraestructura completa para crear y escalar aplicaciones web y móviles con una base de datos PostgreSQL en tiempo real. Además Supabase ofrece una pila completa de herramientas y servicios, incluyendo autenticación de usuarios, almacenamiento de datos en tiempo real, API RESTful, eventos en tiempo real, consultas SQL entre otras funcionalidades. Otra gran ventaja es que utiliza la tecnología de replicación lógica de PostgreSQL para mantener los datos en tiempo real, lo que significa que los cambios realizados en la base de datos se reflejan instantáneamente en la aplicación sin necesidad de hacer solicitudes adicionales al servidor. Esto permite construir aplicaciones rápidas y reactivas. Uno de los principales motivos por los cuáles hemos utilizado Supabase es que es de código abierto y se puede implementar tanto en la nube como en entornos locales, brindando flexibilidad y control a los desarrolladores.



- En nuestra base de datos tenemos diferentes tablas las cuales nos sirven para recoger todos los datos necesarios para nuestro proyecto, estas tablas son:

1. **Users:** Esta tabla viene por defecto en Supabase y recoge la información del usuario cuando se registra, a parte se necesita una validación en formato de correo electrónico y si el usuario no confirma se quedará esperando y no el usuario se queda incompleto.

Manage

Users

Configuration

Policies

Providers

Email Templates

URL Configuration

Q Search by email or phone number

All Users

Reload

Add user

Email	Phone	Provider	Created	Last Sign In	User UID	
dreghztr@gmail.com	-	Email	27 May, 2023 16:03	27 May, 2023 18:52	9a945d88...	...
dreghz@gmail.com	-	Email	27 May, 2023 16:01	27 May, 2023 19:08	4ec343b9...	...
agieksm@gmail.com	-	Email	21 May, 2023 14:44	Waiting for verification...	183de395...	...
enzoruben89@gmail.com	-	Email	18 May, 2023 20:14	18 May, 2023 20:15	f3f4ae34...	...
garciamanjonjoel@fpllefia.com	-	Email	16 May, 2023 20:30	Waiting for verification...	c42f98fe...	...
rubenredondo8@gmail.com	-	Email	16 May, 2023 19:47	27 May, 2023 20:20	47a3296d...	...
jegae@fpllefia.com	-	Email	10 May, 2023 16:53	10 May, 2023 16:55	c749536d...	...
redondobarrosoruben@fpllefia.com	-	Email	26 Apr, 2023 16:07	26 May, 2023 21:20	8f9d46f2...	...

Showing 1 to 8 of 8 results

- Perfiles:** En esta tabla guardaremos la información personal de cada usuario como el nombre o su apellido y le asociaremos a través de la columna user_id el id que se creó al crear su usuario en la tabla users.

	id	created_at	timestampz	nombre	apellidos	email	avatar	user_id	telefono	rol	direccion	codPostal	pais	ciudad
8	8	2023-05-16 17:47:41.867407+00		Administrador	admin	rubenredondo8@gmail.com	avatar.png	47a3296d-891b-482d-a8de-cf4184...	68754812	admin	C/Admin 13	8918	España	Badajoz
1	1	2023-05-03 14:18:08.540638+00		Ruben	Redondo	redondobarrosoruben@fpllefià.com	avatar.png	8f0d46f2-f3b2-484c-ab81-37f028...	628765629	registrado	C/ San Marcos nº13	8918	España	Badajoz
12	12	2023-05-27 14:01:28.804832+00		Joel	Garcia	dreghz@gmail.com	avatar.png	4ec343b9-c1d4-4ecb-9b43-58f3c...	65645468	registrado	Piça Engineer Oculofe	8914	España	Badajoz
16	16	2023-05-27 14:03:16.708948+00		JoelPrueba	Garcia	dreghztr@gmail.com	avatar.png	9a945d08-d21a-4d9c-a4a8-d8de8...	686454684	registrado	C/ Gran Via 32	28013	España	Madrid

- Asientos:** En esta tabla está la información de los asientos disponibles para el usuario cuando va a usar el configurador.

id	created_at	timestampz	fabricante	material	color	modelo	img
4	2023-04-19 16:06:09.559125+00		Recaro	fibra de vidrio	negro	Pole Position	seat4
2	2023-04-19 15:59:34.473069+00		Sparco	carbono	negro	Legend	seat2
3	2023-04-19 16:03:47.681429+00		Sparco	fibra de vidrio	negro	Qrt-r sky	seat3
1	2023-04-19 15:59:06.693019+00		Sparco	carbono	rojo	Legend	seat1

4. **Motor:** En esta tabla está la información de los motores disponibles para el usuario cuando va a usar el configurador.

id	created_at	fabricante	potencia	per	cilindrada	consumo	velocidadMax	aceleracion	numCilindros	img
1	2023-04-19 15:21:39.272915+00	McLaren	825	800	3994	12.4	335	2.8	8	v8cad
2	2023-04-19 15:41:05.531424+00	Audi	620	580	5204	12.9	331	3.1	10	vt0cad
3	2023-04-19 15:49:05.14781+00	Ferrari	799	718	6496	16.4	340	3	12	vt2cad

5. **Ruedas:** En esta tabla está la información de las ruedas disponibles para el usuario cuando va a usar el configurador.

id	created_at	pulgadas	medidas	ruido	fabricante	img
3	2023-04-19 15:55:52.380967+00	17	245/40 R17 91 Y	71	HRE wheel - Bridgestone	tire3
4	2023-04-19 15:56:44.902615+00	20	305/30 R20 103 Y	71	HRE wheel - Bridgestone	tire4
2	2023-04-19 15:53:27.848561+00	19	245/35 R19 93 Y	72	HRE wheel - Continental	tire2
1	2023-04-19 15:52:16.574623+00	18	225/45 R18 95 Y	72	HRE wheel - Continental	tire1

6. **Coche:** Esta tabla recoge los id 's de los valores seleccionados y nos creará el coche con las características elegidas, teniendo así una relación con las tablas anteriores.

id	created_at	asiento	motor	neumatico	aleron
13	2023-05-27 14:24:33.302487+00	3	1	1	low

7. **Pedidos:** Esta tabla asocia el coche creado con el perfil del usuario que lo ha creado y a parte de tener un id crea aleatoriamente un numero de pedido.

	id int8	created_at date	id_coche int8	id_perfil int8	numeroPedido uuid
	4	2023-03-09	1	1	bed57704-8269-42fd-9ade-56973d028060
	9	2023-05-26	11	1	f1446f33-cb3c-44ee-a702-2c42b7ea22aa
	10	2023-05-26	12	1	bb1b85c8-546d-42a6-a23e-6fb0a8acb045
	11	2023-05-27	13	16	6eecd2d2-9195-45af-b672-9a5d999b6450
	15	2023-05-27	17	16	39353c60-ca61-47dc-82a3-a2eabb4b295b

Este es el ejemplo del mapping de la tabla users

```

JS userjs X
src > bd > JS userjs > ...
1 // Importamos la conexión a la base de datos
2 import { supabase } from './supabase.js'
3
4 export class User {
5   // Mapping de propiedades de la tabla perfiles
6   constructor (id = null, email = null, password = null) {
7     this.id = id
8     this.email = email
9     this.password = password
10  }
11
12  // crear registro (método static que se puede leer desde la clase sin necesidad de crear una instancia)
13  static async create (userData) {
14    const { data, error } = await supabase.auth.signUp(userData)
15
16    if (error) {
17      throw new Error(error.message)
18    }
19    console.log('usuario creado correctamente ', data)
20    return new User(data.user.id, data.user.email)
21  }
22
23  // login
24  static async login (userData) {
25    // USER LOGIN
26    const { data, error } = await supabase.auth.signInWithPassword(userData)
27    if (error) {
28      throw new Error(error.message)
29    }
30    console.log('usuario logeado', data.user)
31    return new User(data.user.id, data.user.email)
32  }
33
34  // logout
35  static async logout () {
36    // USER LOGOUT
37    const { error } = await supabase.auth.signOut()
38    if (error) {
39      throw new Error(error.message)
40    }
41    return true
42  }
43
44  // leer user logeado
45  static async getUser () {
46    // GET USER
47    const { data: { user }, error } = await supabase.auth.getUser()
48    console.log('Usuario logeado desde getuser', user)
49    if (error) {
50      throw new Error(error.message)
51    }
52    return new User(user.id, user.email)
53  }
54 }
55
56

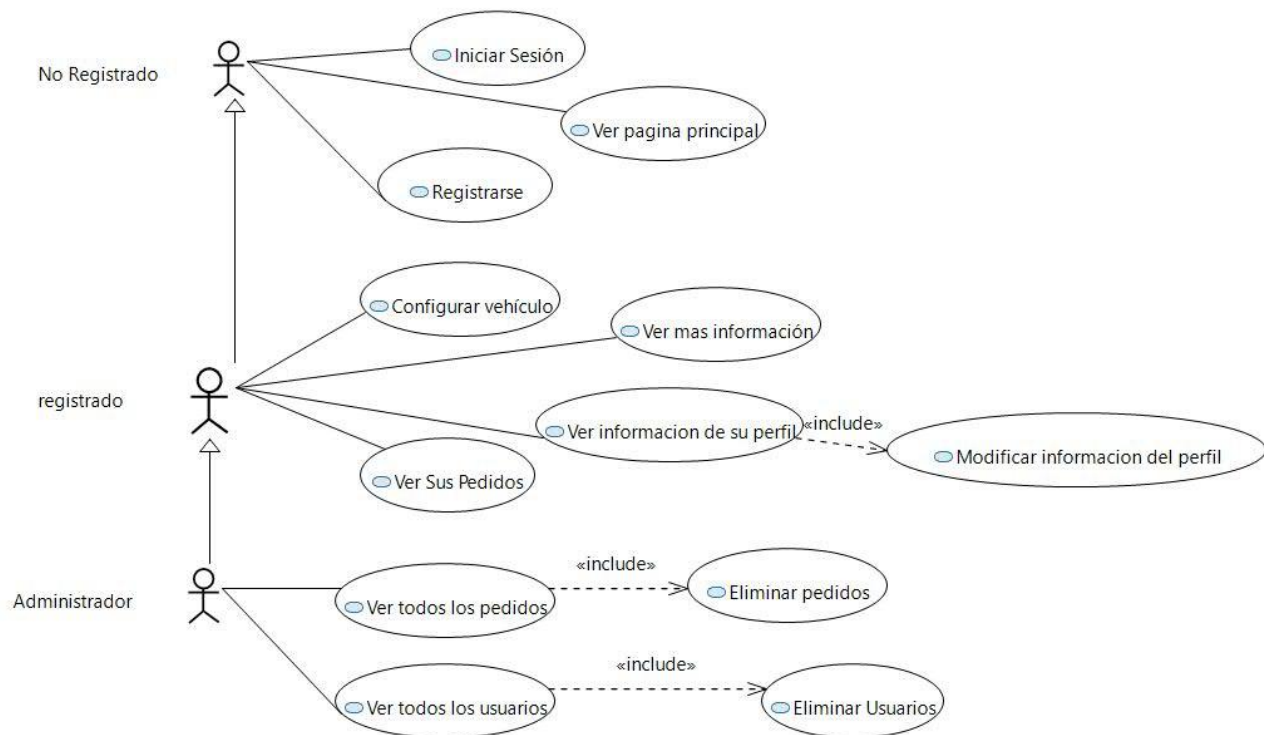
```

3.3 Funcionalidad:

3.3.1 Diagrama de casos de uso:

- La idea de nuestro proyecto es ir dando pistas al usuario según vaya avanzando en la página, por ello un usuario anónimo solo puede ver la página principal. Si quiere obtener más información o configurar su vehículo necesitará registrarse o en caso de que ya tenga una cuenta deberá iniciar sesión
- Una vez que el usuario ya está registrado y logueado podrá obtener más información del vehículo e ir ya al configurador. También tendrá un menú en el cual podrá ver la información de su perfil y otro para ver sus pedidos.

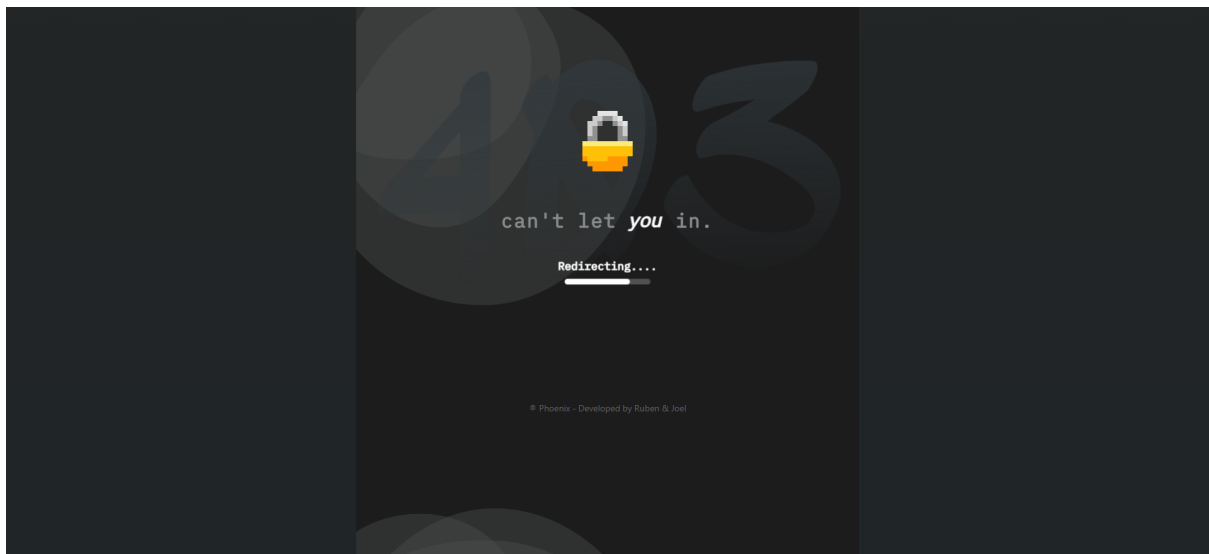
- El ultimo tipo de usuario es el administrador el cual puede realizar todas las acciones anteriores y podrá exclusivamente, ver todos los usuarios y eliminarlos y ver todos los pedidos y eliminarlos si es necesario.



3.3.2 Restricciones de la página:

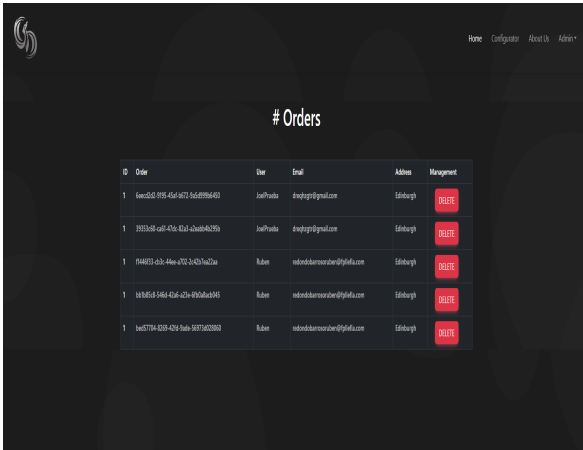
- Para garantizar que la página cumple con nuestros casos de uso a la hora de cargar las diferentes vistas se comprueba si hay un usuario logueado en todas las ventanas menos en la página principal ya que es la única en la cual puede acceder cualquiera. En caso de no tener acceso se muestra una pantalla de acceso denegado y se redirige a la página principal.

Usuario anónimo intentando entrar al configurador



- A parte de la verificación anterior también se comprueba el rol que tiene el perfil de cada usuario en algunos casos como en el desplegable del header para ver qué opciones tiene disponible cada usuario o para cargar las tablas con el botón de eliminar.

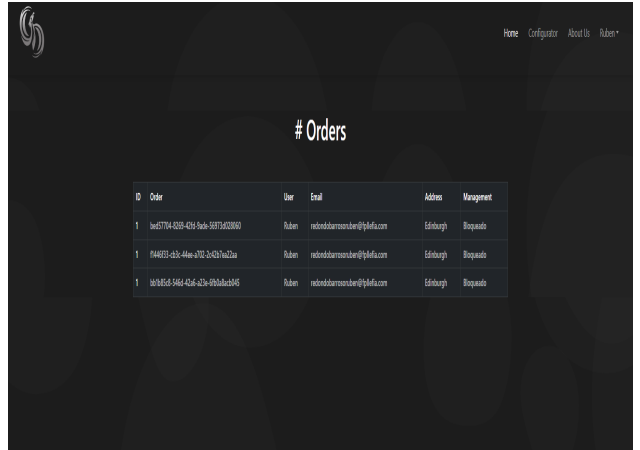
Vista del administrador



The screenshot shows the administrator interface with a dark theme. At the top, there is a navigation bar with links: Home, Configurador, About Us, and Admin*. The main content area is titled "# Orders" and contains a table with the following data:

ID	Order	User	Email	Address	Management
1	6aee05d-3195-426f-b271-b4d0f96a4d3	JoaPeueto	joapeueto@gmail.com	Edinburgh	EDITAR
1	33333d8-c9d4-4336-b3d3-a3a0b6b293a	JoaPeueto	joapeueto@gmail.com	Edinburgh	EDITAR
1	f1448333-c33c-44ae-a702-34c317a212aa	Ruben	rubendurroncoran@fpifeia.com	Edinburgh	EDITAR
1	1d3d45d-5464-42d6-a23a-993d8a3045	Ruben	rubendurroncoran@fpifeia.com	Edinburgh	EDITAR
1	1a657794-8209-4209-b9da-9d73a020002	Ruben	rubendurroncoran@fpifeia.com	Edinburgh	EDITAR

Vista de un usuario registrado



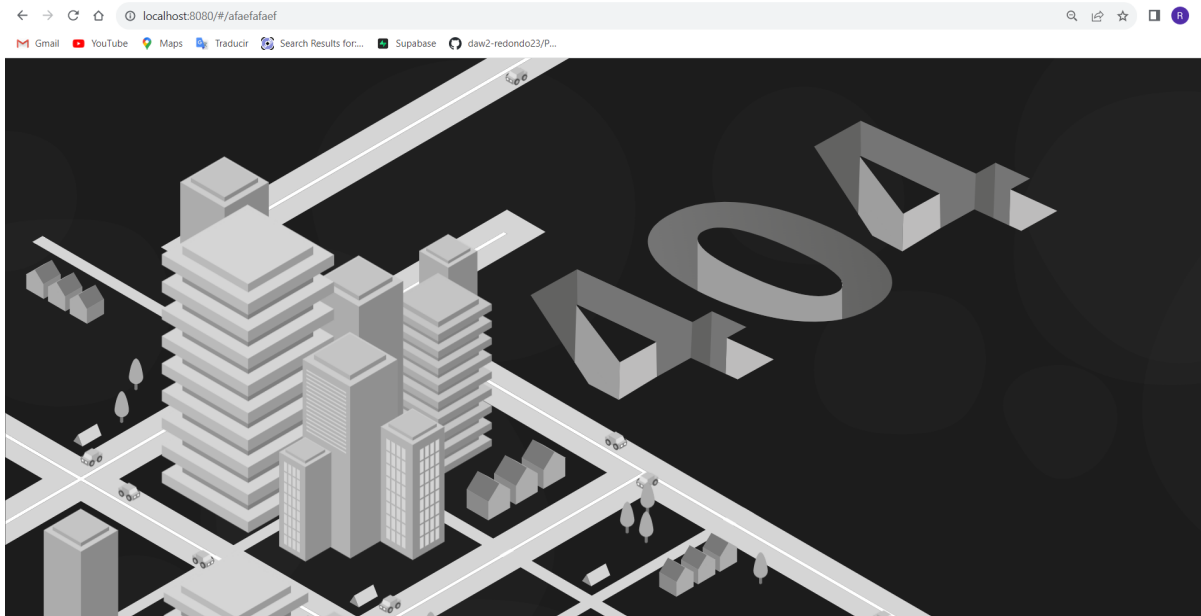
The screenshot shows the user interface with a dark theme. At the top, there is a navigation bar with links: Home, Configurador, About Us, and Ruben*. The main content area is titled "# Orders" and contains a table with the following data:

ID	Order	User	Email	Address	Management
1	1a657794-8209-4209-b9da-9d73a020002	Ruben	rubendurroncoran@fpifeia.com	Edinburgh	Boqueado
1	f1448333-c33c-44ae-a702-34c317a212aa	Ruben	rubendurroncoran@fpifeia.com	Edinburgh	Boqueado
1	1d3d45d-5464-42d6-a23a-993d8a3045	Ruben	rubendurroncoran@fpifeia.com	Edinburgh	Boqueado

- Como nuestra página es una SPA(Single Page Application) es muy importante controlar bien las rutas con el router, entonces para controlarlas llamamos a unas funciones que tiene definidas el la propiedad script del router, estas sirven para leer la url, se separa para ver cual es el nombre de la vista a cargar, se pone ese

nombre en nuestro array de rutas y si existe se carga esa plantilla en el main junto a su funcionalidad, en caso de no existir esa ruta se carga una plantilla de error 404 para indicar al usuario que esa ruta no existe.

Vista error404



4.Conclusiones

4.1 Conclusión del proyecto:

- En conclusión, este proyecto ha sido una oportunidad para explorar y aplicar los conocimientos de desarrollo web utilizando tecnologías como html, css, javascript y supabase. A través de la creación de una spa centrada en una marca de coches, se ha

logrado el objetivo de ofrecer una experiencia interactiva y atractiva para los usuarios. Además, la utilización de Supabase como infraestructura backend ha brindado una base sólida y escalable para gestionar la base de datos y proporcionar funcionalidades en tiempo real. En resumen, este proyecto ha demostrado la capacidad de utilizar tecnologías avanzadas de desarrollo web para crear una experiencia envolvente y efectiva, al tiempo que se refuerza la imagen de la marca y se atrae a potenciales clientes.

4.2 Propuestas de mejora:

- Algunas propuestas de mejora són:

1. Mayor número de opciones para personalizar los vehículos.
2. Mejora en la sincronía entre vistas.
3. Implementación de los precios de cada pieza e inserción de un contador en el configurador.
4. Implementar más animaciones css para algunos componentes.

5. Fuentes

Front-End:

Para el desarrollo del apartado Front-End hemos utilizado algunas ayudas, aquí os dejamos las fuentes utilizadas:

Bootstrap

<https://getbootstrap.com/>

Hemos utilizado el framework para poder desarrollar mejor el apartado de diseño y de visibilidad de toda la web y todas sus páginas.

SweetAlert

<https://sweetalert2.github.io/>

Sabíamos que utilizando la biblioteca de/sweetalert conseguimos una mejor apariencia en las alertas que queríamos añadir, por eso mismo creíamos que la mejor opción para esta funcionalidad era SweetAlert.

Blender

<https://www.blender.org/>

Decidimos utilizar Blender para poder crear nuestros propios diseños de maquetas de vehículos en 3D, para de esta manera no tener que utilizar maquetas o imágenes sin permiso, si no que decidimos crearlos nosotros mismos.

CSS Portal

<https://www.cssportal.com/css-resources.php>

En varias ocasiones hemos visitado esta web, ya que para diseños es muy útil y puedes coger muchas ideas para luego adaptarlas a tu proyecto, es cierto que no hemos cogido código de esta web, pero sí que nos ha inspirado y nos ha facilitado el desarrollo del diseño dándonos ideas, por estas razones hemos decidido incluirla en las fuentes.

BGJar

<https://bgjar.com/animated-shape>

Esta web la utilizamos únicamente para la creación de los background, ya que nos encantaba el diseño que nos proporcionaba, la fácil personalización de los mismos y sobre todo

que es de libre utilización ya que realmente los fondos los creas tú mismo. La web te proporciona las herramientas necesarias para poder hacerlo de manera muy sencilla.

MDBbootstrap

<https://mdbbootstrap.com/>

Biblioteca utilizada para poder expresar al máximo Bootstrap.

Shadows Brumm

<https://shadows.brumm.af/>

Esta web realmente la hemos utilizado muy poquito, pero nos ha sido muy útil para poder crear las sombras de los cards y de los box`s.

HTML Code Generator

<https://www.html-code-generator.com/css/rgba-color-generator>

Decidimos utilizar esta ayuda ya que veíamos que era la mejor opción para la implementación de colores y sobre todo de las transparencias y opacidades.

Universe

<https://uiverse.io/all>

Creo que esta ayuda ha sido de las más utilizadas, es una web que te proporciona plantillas de botones, cards, forms... y con la gran opción de poder personalizarlos a tu manera partiendo del código que te proporcionan. Es cierto que no puedes utilizar el código que ellos te proporcionan ya que no tendrá la misma apariencia que la que está en la web, pero con un par de retoques de CSS es sencillo conseguir el resultado deseado.

Freefrontend

<https://freefrontend.com/css-hover-effects/>

De esta web tampoco he utilizado ningún código, pero me ha sido muy útil para coger ideas y sobre todo para entender las pseudoclasses de CSS.

Dev.to

<https://dev.to/stackfindover/35-html-404-page-templates-5bge>

Web utilizada para las plantillas de error 404 y Access Denied.

Back-End:

Para el desarrollo del Back-End hemos utilizado alguna ayuda externa

Vanilla Pills - Carlos Arrebola

<https://carrebola.github.io/vanillaPill/docs/intro/>

Hemos usado esta web de ejemplo para poder hacer nuestra SPA, parte del código lo hemos cogido de aquí para tener una base sobre la cual trabajar.

6.Anexos

Blender

<https://www.blender.org/>

Hemos utilizado el programa Blender para poder crear nuestros propios diseños de vehículos 3D, decidimos ponerlo en anexos ya que realmente no tiene nada que ver con el desarrollo ni con código.

Nos ha sido muy útil ya que a través de nuestros propios diseños 3D hemos podido hacer diferentes variantes de imágenes como por ejemplo la de poder poner la manta encima del vehículo.

BeFunky

<https://www.befunky.com/es/opciones/editor-de-fotos/>

Para poder editar las imágenes ya sea en eliminar fondos, editar los colores etc... decidimos utilizar un editor de imágenes online y BeFunky fue la mejor opción bajo nuestro punto de vista.

Decidimos escogerlo por la calidad de las imágenes que nos proporcionaba y por las amplias opciones que nos proporcionaba.