

# Infoargo

## Memoria Técnica

13/03/2023

Óscar, Adrián, Gil Pablo, Rubén

<b>1. Introducción</b>	<b>3</b>
<b>2. Diseño de la Base de datos</b>	<b>4</b>
<b>3. Canva / Diseño inicial</b>	<b>5</b>
<b>4. Visual Studio Code</b>	<b>5</b>
<b>5. Docker</b>	<b>6</b>

# 1. Introducción

Para la resolución de este reto, hemos dado uso a varias tecnologías para la facilitación de nuestro proyecto.

Empezamos por la realización de un diseño conceptual común junto a los demás grupos del reto para crear una base de datos completa y esencial para crear esta aplicación web.

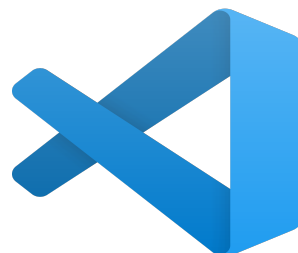
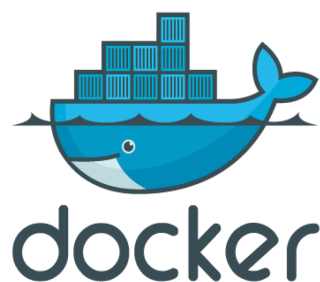
Una vez realizada la base de datos, se comenzó a crear la página a través de Canva con los mockups para tener un modelo gráfico a seguir.

A continuación, le damos forma a eso mockups gracias a Visual Studio Code con el uso de lenguajes como HTML, PHP o JAVASCRIPT y de un MVC (Modelo Vista Controlador), el cuál nos ha ayudado con la estructura del trabajo.

Para finalizar, se llevó a cabo la creación de un servidor.

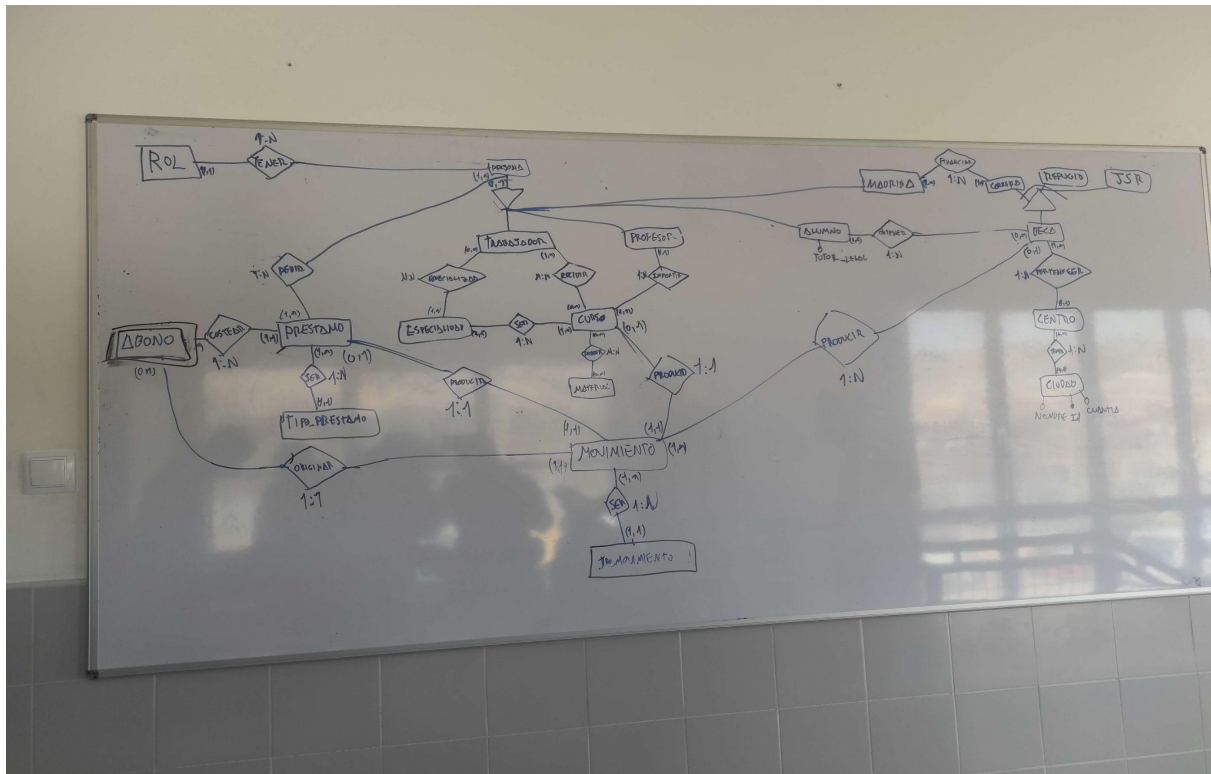
Se ha utilizado docker para el despliegue de la aplicación web, donde hemos añadido una imagen de Nginx para crear el servidor.

Además, para mayor coordinación y organización de grupo se usa un cartel con las cosas que hacer y así repartir la faena entre los integrantes del grupo. Así como plataformas como Github, Google Drive o Discord para subir los trabajos o actualizaciones realizadas y tener una plataforma de comunicación siempre que no estemos en clase.



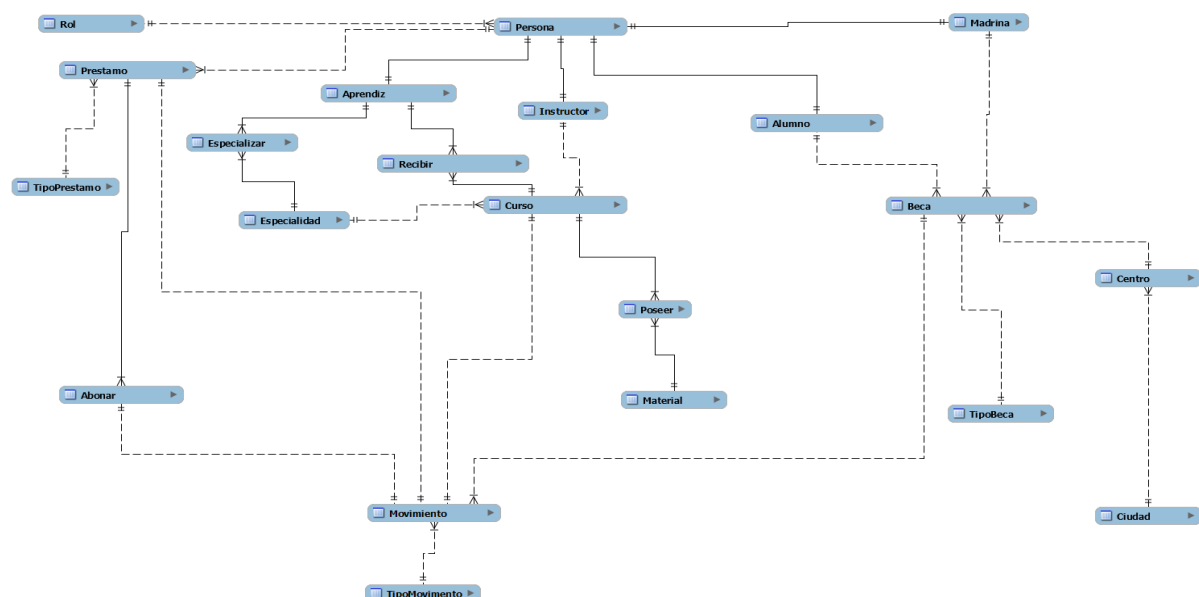
## 2. Diseño de la Base de datos

Como he dicho antes, para el diseño de la base de datos nos juntamos todos los equipos de reto para hacer un diseño conceptual común y eficiente, donde utilizamos una pizarra grande para dibujarlo.



Una vez finalizada, la pasamos a limpio, y para ello hemos utilizado la aplicación Dia.

A continuación, seguimos creando mediante MySQL Workbench las tablas y atributos de la base de datos para finalizar con modelo lógico.



### 3. Canva / Diseño inicial

Se diseñó un boceto inicial donde poder fijarse.

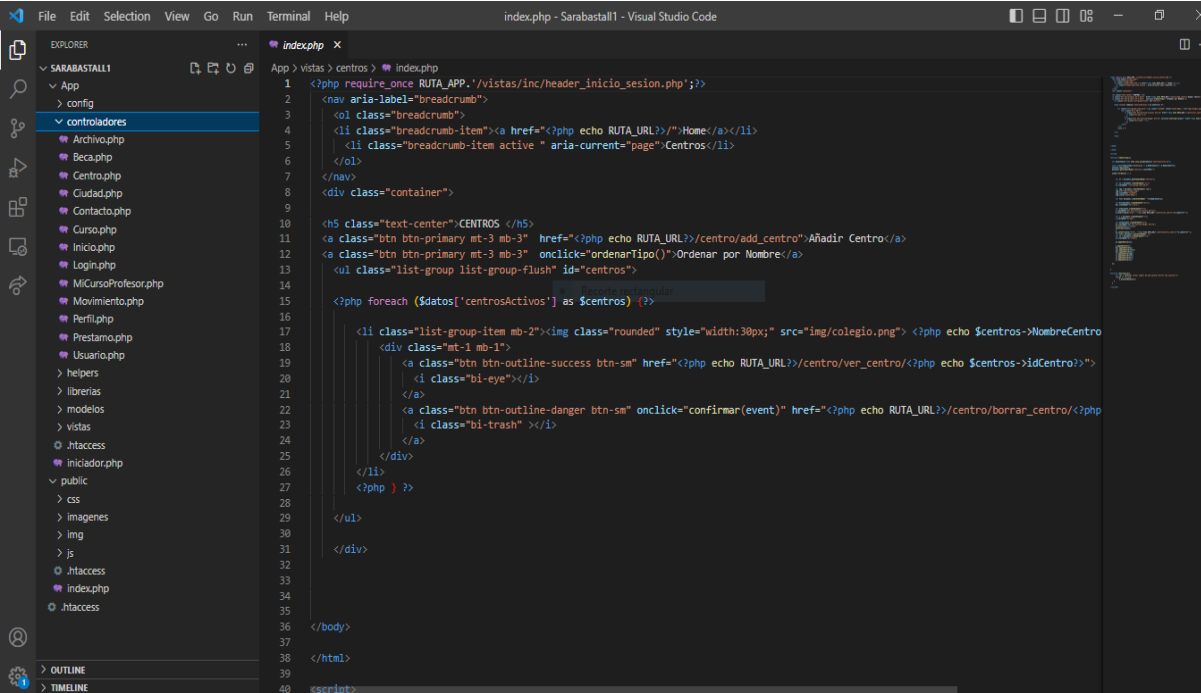
[https://www.canva.com/design/DAFX0t7Qw2A/YbWwmSuYcSmlhyjK5Wjilg/watch?utm\\_content=DAFX0t7Qw2A&utm\\_campaign=designshare&utm\\_medium=link2&utm\\_source=sharebutton](https://www.canva.com/design/DAFX0t7Qw2A/YbWwmSuYcSmlhyjK5Wjilg/watch?utm_content=DAFX0t7Qw2A&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton)

Este boceto estaba sujeto a cambios y el resultado terminó teniendo la misma base pero diferentes detalles en cada apartado.

Decidimos eliminar el menú lateral y convertirlo en un modal para dejar más espacio.

### 4. Visual Studio Code

En VSC, hemos seguido el MVC o Modelo-Vista-Controlador es un patrón de arquitectura de software que, utilizando 3 componentes (Vistas, Models y Controladores) separa la lógica de la aplicación de la lógica de la vista en una aplicación.




```
1 <?php require_once RUTA_APP.'/vistas/inc/header_inicio_sesion.php';>
2 <nav aria-label="breadcrumb">
3   <ol class="breadcrumb">
4     <li class="breadcrumb-item"><a href="<?php echo RUTA_URL?>"/">Home</a></li>
5     <li class="breadcrumb-item active" aria-current="page">Centros</li>
6   </ol>
7 </nav>
8 <div class="container">
9
10  <h5 class="text-center">CENTROS </h5>
11  <a class="btn btn-primary mt-3 mb-3" href="<?php echo RUTA_URL?>/centro/add_centro">Añadir Centro</a>
12  <a class="btn btn-primary mt-3 mb-3" onclick="ordenarTipo()">Ordenar por Nombre</a>
13  <ul class="list-group list-group-flush" id="centros">
14
15    <?php foreach ($datos['centrosActivos'] as $centros) {?>
16
17      <li class="list-group-item mb-2"> <?php echo $centros->NombreCentro
18        <div class="mt-1 mb-1">
19          <a class="btn btn-outline-success btn-sm" href="<?php echo RUTA_URL?>/centro/ver_centro/<?php echo $centros->idCentro?>">
20            <i class="bi-eye"></i>
21          </a>
22          <a class="btn btn-outline-danger btn-sm" onclick="confirmar(event)" href="<?php echo RUTA_URL?>/centro/borrar_centro/<?php
23            <i class="bi-trash" ></i>
24          </a>
25        </div>
26      </li>
27    </ul>
28  </div>
29
30  <?php } ?>
31
32 </div>
33
34 </body>
35
36 </html>
37
38 </script>
```

Hemos complementado varios lenguajes de programación, como son PHP y JavaScript, además de HTML.

## 5. Docker

Para la implementación del servidor, se ha utilizado Docker que nos permite crear contenedores, los cuales contienen la imagen de cualquier servicio que se quiera implementar. En este caso ha sido una imagen de Nginx (richarvey/nginx-php-fpm), además de otros servicios como MySQL y PHPMyAdmin.

 comandos: Bloc de notas

Archivo Edición Formato Ver Ayuda

|---COMANDOS BAJAR IMÁGENES DE DOCKERHUB---

```
docker pull mysql
```

```
docker pull myadmin
```

```
docker pull richarvey/nginx-php-fpm
```

---COMANDOS CREAR VOLUMENES---

```
docker volume create nginx-data
```

```
docker volume create nginx-conf
```

```
docker volume create mysql-data
```

```
docker volume create mysql-conf
```

---COMANDOS ARRANCAR CONTENEDORES CON VOLUMENES---

```
docker run --name mysql -p 3306:3306 -v mysql-data:/var/lib -v mysql-conf:/etc/mysql -e
```

```
MYSQL_ROOT_PASSWORD=alumno -d mysql
```

```
docker run --name myadmin -d --link mysql:db -p 8080:80 phpmyadmin
```

```
docker run --name nginx -p 80:80 -p 443:443 -v nginx-data:/var/www/html -v
```

```
nginx-conf:/etc/nginx/sites-enabled -d richarvey/nginx-php-fpm
```

---COMANDOS INICIAR AUTOMATICAMENTE CONTENEDORES---

```
docker update --restart=always nginx
```

```
docker update --restart=always mysql
```

```
docker update --restart=always myadmin
```

---COMANDO PARA GENERAR CERTIFICADO---

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/key.key -out
```

```
/etc/ssl/certs/cert.crt
```

---COMANDO COPIAR DE LOCAL A CONTENEDOR---

```
docker cp /etc/ssl/certs/cert.crt nginx:/usr/local/nginx
```

```
docker cp /etc/ssl/private/key.key nginx:/usr/local/nginx
```

---COMANDO ENTRAR EN EL CONTENEDOR---

```
docker exec -it nginx bash
```