

MEMORIA TFG

MOTOWIKI

Moto-Wiki

Este proyecto consiste en una página web, que contendrá información sobre motocicletas y sus fabricantes, en la que los usuarios podrán interactuar, y los dueños y colaboradores de esta tengan un sencillo control de toda la información que se maneja en esta.

Anteproyecto

Revisado de apps alternativas/parecidas, destacando sus puntos fuertes y débiles.

MotoFichas.com

- Es una página que recoge la mayoría de fichas técnicas de motocicletas del mercado.
- Su punto fuerte es la cantidad de información que tiene gracias al tiempo que lleva activa además de su popularidad.
- Su punto débil es que es una página poco amigable para personas con bajos conocimientos del tema, además de estar muy comercializada.

Motos.net

- Es una página de venta de vehículos, tanto de primera como de segunda mano. Su punto fuerte es el gran mercado que se ha creado en torno a esta aplicación. Su punto débil es que está limitada a la compra-venta del producto.

Explicación de la idea general en la que se basa vuestra app y que problemas soluciona.

-La idea principal de la aplicación será hacer una especie de Wiki para todo lo relacionado con las motocicletas, con un estilo similar a la Wikipedia, haciendo que los distintos usuarios (elegidos por los administradores) puedan añadir contenido a la web, principalmente para completar la wiki como tal, corregir errores o añadir enlaces a lugares en los que comprar/encontrar el contenido en cuestión.

OBJETIVOS

- Conseguir una web completamente funcional, con todos los apartados y requisitos cumplidos.
- Que sea una web fiel al diseño y propósito Inicial.
- Que sea una aplicación completamente manejable por los usuarios, y actualizable sin necesidad de tener conocimientos del código.
- Tener una interfaz clara, que sea accesible y de fácil uso.

ALTERNATIVAS

- Gestor de Portfolios Web -> La idea era realizar una aplicación muy polivalente para que los usuarios pudiesen montar su propio portfolio en la web, con muchas posibilidades y haciendo que el acceso a este fuese mucho más sencillo. (Finalmente descartado por su posible complejidad y búsqueda de opciones más factibles)
- Gestor de APIs -> La idea de este proyecto era similar al anterior, pero dando la posibilidad a los usuarios de montar sus propias APIs desde el servicio web, y darle acceso a estos fácilmente. (Pienso que era buena idea, pero que se podía quedar algo corta pero compleja.)
- Página Web de Reservas de Restaurante -> Es una de las ideas más típicas para este tipo de proyectos, te permite usar todo lo aprendido sin complejidad alguna. Finalmente descartada por ser muy típica, y buscar algo más original y personal.

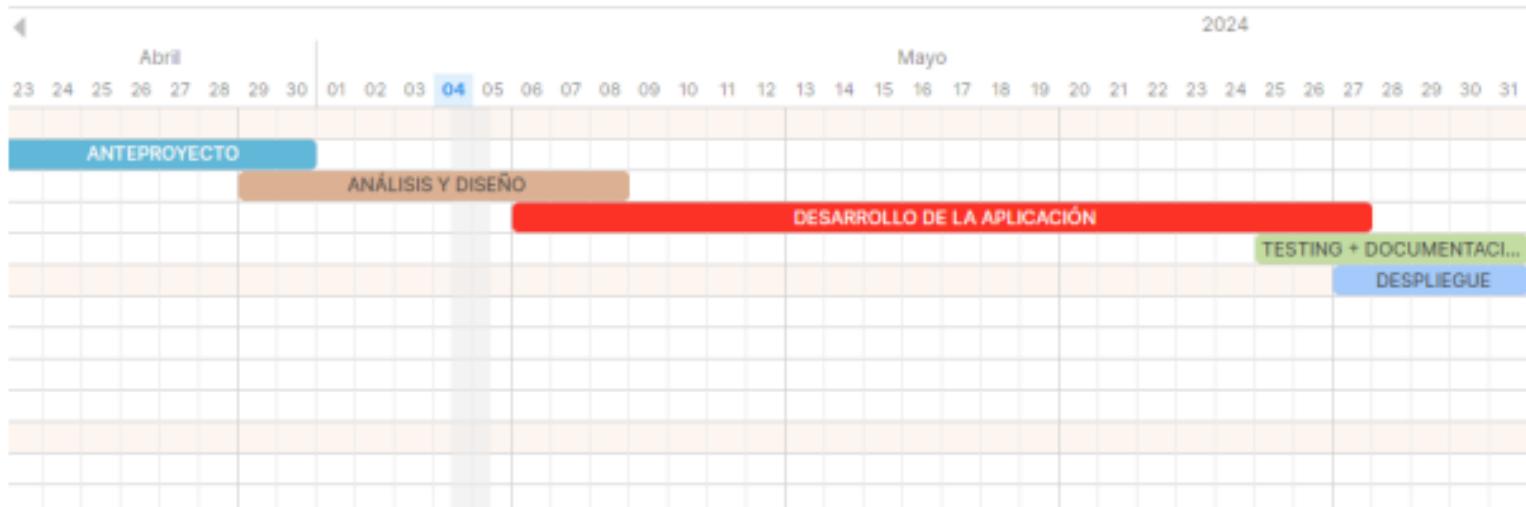
Análisis DAFO

	Debilidades	Amenazas	
Análisis Interno	Habría que buscar una forma externa de ingreso, ya que la página no generaría ingresos por sí sola.	Algunas de las empresas o servicios ya asentados podrían hacer algo similar, y contando con más popularidad podría afectar al desarrollo de nuestra idea.	Análisis Externo
	Fortalezas	Oportunidades	
	Idea fácil de desarrollar con bases ya existentes pero idea ramificada de los servicios similares, haciéndolo más amigable para el público casual.	Oportunidad de entrar al mercado como un producto distinto a lo ya existente, apartandolo de la comercialización, haciéndolo un producto informativo generalmente.	

Planificación de las diferentes fases que componen el proyecto

- **Diagrama de Gantt realizado con herramientas como Excel, Tom's Planner, diagrams.net o GanttProject.**

Realmente el anteproyecto empieza mucho antes, pero a partir del día 23 es cuando se empiezan a tomar las decisiones definitivas, como podemos ver al empezar algo más tarde hay días en los que coinciden algunas etapas, esto se debe a que algunos apartados pueden sufrir cambios a medida que se van empezando las distintas etapas.



- **Ciclo de desarrollo elegido para realizar el trabajo**

El tipo de desarrollo elegido finalmente, es una mezcla entre el desarrollo iterativo e incremental, con algunos toques del desarrollo ágil (ya que teniendo una pequeña idea de como será el proyecto, he empezado a desarrollar el diseño y a medida que avanzaba se iban determinando los requisitos de la aplicación.), basándose sobre todo en avances graduales y la iteración sobre el diseño, una vez terminado, entraríamos en la programación y desarrollo de la web con un modelo en cascada.

Análisis Software

-Requisitos Funcionales

1 - Gestión de Usuarios

- 1.1- Registrar Usuario
- 1.2- Iniciar Sesión
- 1.3- Modificar Datos Usuario
- 1.4- Marcar Motocicleta Favorita
- 1.5- Desmarcar Motocicleta Favorita

2 - Gestión de Noticias

- 2.1- Crear Noticia
- 2.2- Modificar Noticia
- 2.3- Borrar Noticia

3 - Gestión de Marcas

- 3.1 Crear Marca
- 3.2 Modificar Marca

- 3.3 Suspender/Borrar Marca

4 - Gestión de Motocicletas

- 4.1 Crear Motocicleta
- 4.2 Modificar Datos Motocicleta
- 4.3 Borrar Motocicleta
- 4.4 Crear Ofertas
- 4.5 Modificar Oferta
- 4.6 Borrar Oferta
- 4.7 Mostrar Motocicleta por Parametros

5 - Gestión de Administrador

- 5.1- Gestionar Usuarios
- 5.2- Gestionar API
- 5.3- Ver Registros de Modificaciones

6 - Sistema de Busqueda

- 6.1- Busqueda tanto por Marca como Motocicleta.
- 6.2- Sistema de Busqueda de Usuario y Logs (Para administradores).

-Requisitos No Funcionales

1 - Rendimiento

- 1.1- No cargar módulos NO necesarios.
- 1.2- Hacer llamadas API/BD solo cuando sea necesario.
- 1.3- Llamadas con demasiada informacion SOLO en páginas específicas.

2 - Seguridad

- 2.1- Tener bastante control de los usuarios con permisos (Administradores).
- 2.2- Habrá varios niveles de permisos para evitar problemas mayores.

3 - Usabilidad

- 3.1- La página contará con una interfaz muy intuitiva.
- 3.2- Contará con una gran cantidad de accesos a las distintas páginas de esta (por ejemplo enlaces a dedicadas de motocicletas/marcas) para facilitar el llegar a las distintas secciones de esta.

4 - Fiabilidad

- 4.1- La página solo tendrá información real y verificada por los distintos administradores de la página.
- 4.2- La página estará constantemente siendo actualizada, corrigiendo errores o imperfecciones en los datos que pueda haber.

5 - Accesibilidad

- 5.1- Se intentará en la medida de lo posible hacer la web lo más accesible para usuarios con discapacidades cumpliendo con los estándares WCAG.

6 - Eficiencia del código

- 6.1- El código estará estructurado de la forma más eficientemente posible, evitando llamadas o cargas de archivos innecesarios.
- 6.2- Estructuración de carpetas con Modelo-Vista-Controlador, haciendo muy fácil la corrección del código con comentarios y para futuras actualizaciones internas de la web.

-Casos de Uso y Diagramas de Clases

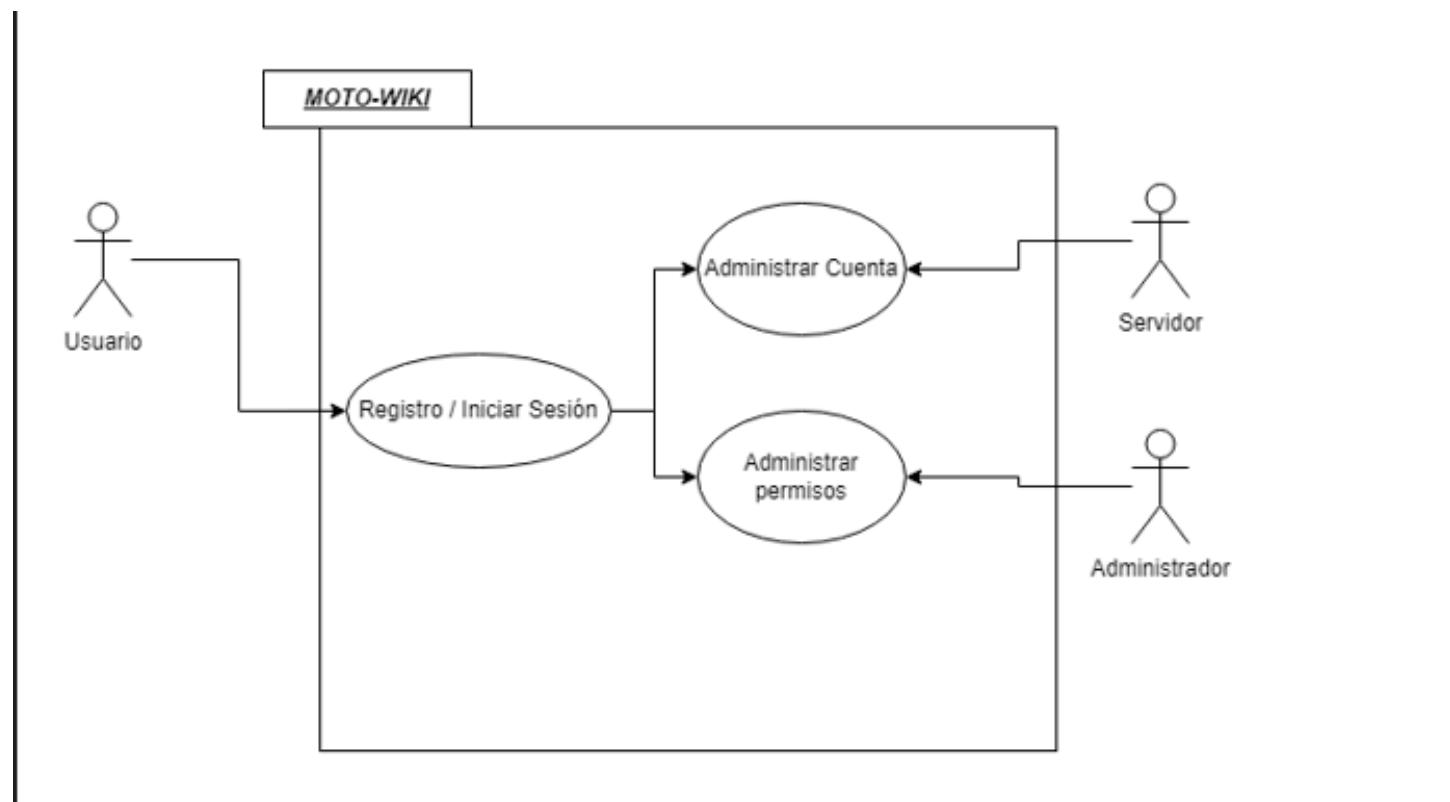
<i>dbMotoWiki</i>	<i>Usuario</i>	<i>Motocicleta</i>	<i>Fabricante</i>
<pre>:conexion :conectarBD()</pre>	<p><i>idUsuario</i></p> <p><i>nombre</i></p> <p><i>apellido1</i></p> <p><i>apellido2</i></p> <p><i>username</i></p> <p><i>email</i></p> <p><i>fechaRegistro</i></p> <p><i>fechaNacimiento</i></p> <p><i>__set()</i></p> <p><i>__get()</i></p> <p><i>comprobarSesion()</i></p> <p><i>iniciarSesion()</i></p> <p><i>cerrarSesion()</i></p> <p><i>editarDatos()</i></p> <p><i>cambiarPermisos()</i></p>	<p><i>idMotocicleta</i></p> <p><i>nombreModelo</i></p> <p><i>idfabricante</i></p> <p><i>imagenMotocicleta</i></p> <p><i>descripcion</i></p> <p><i>precioMax</i></p> <p><i>precioMin</i></p> <p><i>popularidad</i></p> <p><i>tipoCarnet</i></p> <p><i>tag</i></p> <p><i>arranque</i></p> <p><i>capacidad</i></p> <p><i>transmision</i></p> <p><i>marchas</i></p> <p><i>tipoMotor</i></p> <p><i>refrigeracion</i></p> <p><i>potencia2 (kW)</i></p> <p><i>potencia1 (CV)</i></p> <p><i>cilindrada</i></p> <p><i>tipoMoto</i></p> <p><i>anioFabricacion</i></p> <p><i>__set()</i></p> <p><i>__get()</i></p> <p><i>generarTarjeta()</i></p> <p><i>generarModulo()</i></p> <p><i>mostrarInformacion()</i></p> <p><i>insertarMotocicleta()</i></p> <p><i>modificarMotocicleta()</i></p> <p><i>borrarMotocicleta()</i></p> <p><i>toggleFavorita()</i></p>	<p><i>idFabricante</i></p> <p><i>nombreFabricante</i></p> <p><i>paisOrigen</i></p> <p><i>fechaFundada</i></p> <p><i>sitioWeb</i></p> <p><i>descripcion1</i></p> <p><i>descripcion2</i></p> <p><i>imagenFabricante</i></p> <p><i>__set()</i></p> <p><i>__get()</i></p> <p><i>mostrarInformacion()</i></p> <p><i>modificarInfoFabricante()</i></p> <p><i>insertarNuevoFabricante()</i></p> <p><i>eliminarFabricante()</i></p>

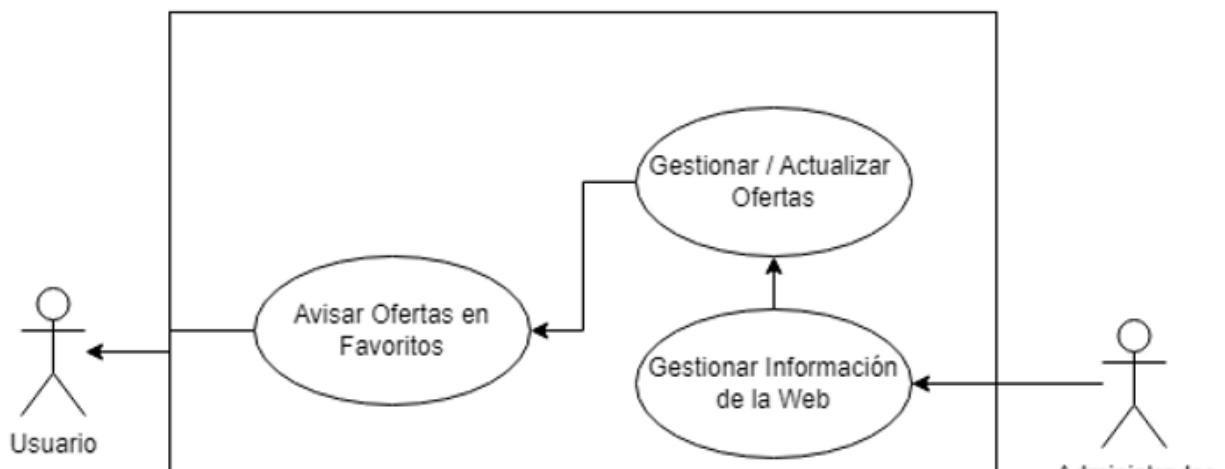
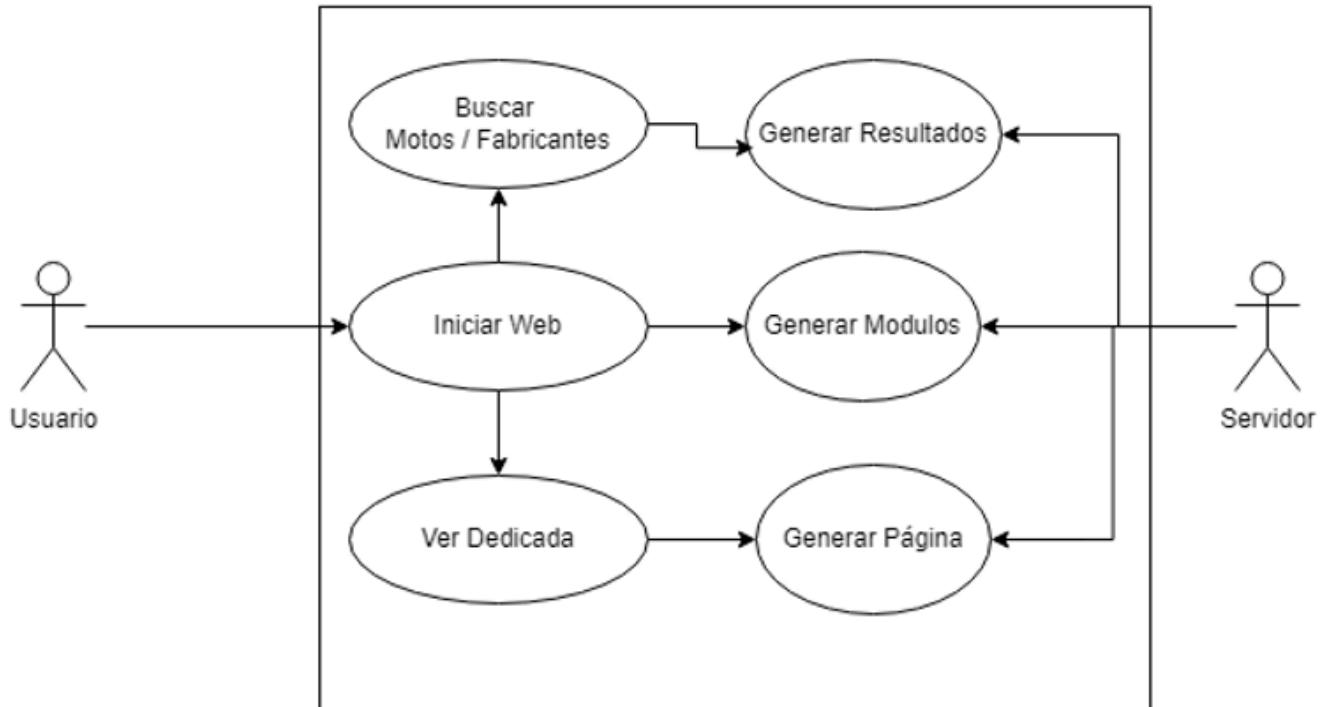
<i>Noticias</i>
<u><i>idNoticia</i></u>
idUsuario
imagenNoticia
fechaPublicacion
entrada3
entrada2
entrada1
tituloNoticia
<u><i>__get()</i></u>
<u><i>__set()</i></u>
modificarNoticia()
borrarNoticia()
crearNoticia()

<i>Oferta</i>
<u><i>idOferta</i></u>
idMotocicleta
precio
enlaceOferta
<u><i>__get()</i></u>
<u><i>__set()</i></u>
mostrarOfertas()
insertarOferta()
caducarOferta()
modificarOferta()

<i>Registros / Logs</i>
<u><i>idRegistro</i></u>
idUsuario
descripcionCambios
tipoCambio
fechaCambio
<u><i>__set()</i></u>
<u><i>__get()</i></u>
generarRegistro()

<i>Favoritos</i>
<u><i>idUsuario</i></u>
<u><i>idMotocicleta</i></u>
<u><i>__set()</i></u>
<u><i>__get()</i></u>
mostrarFavoritos()



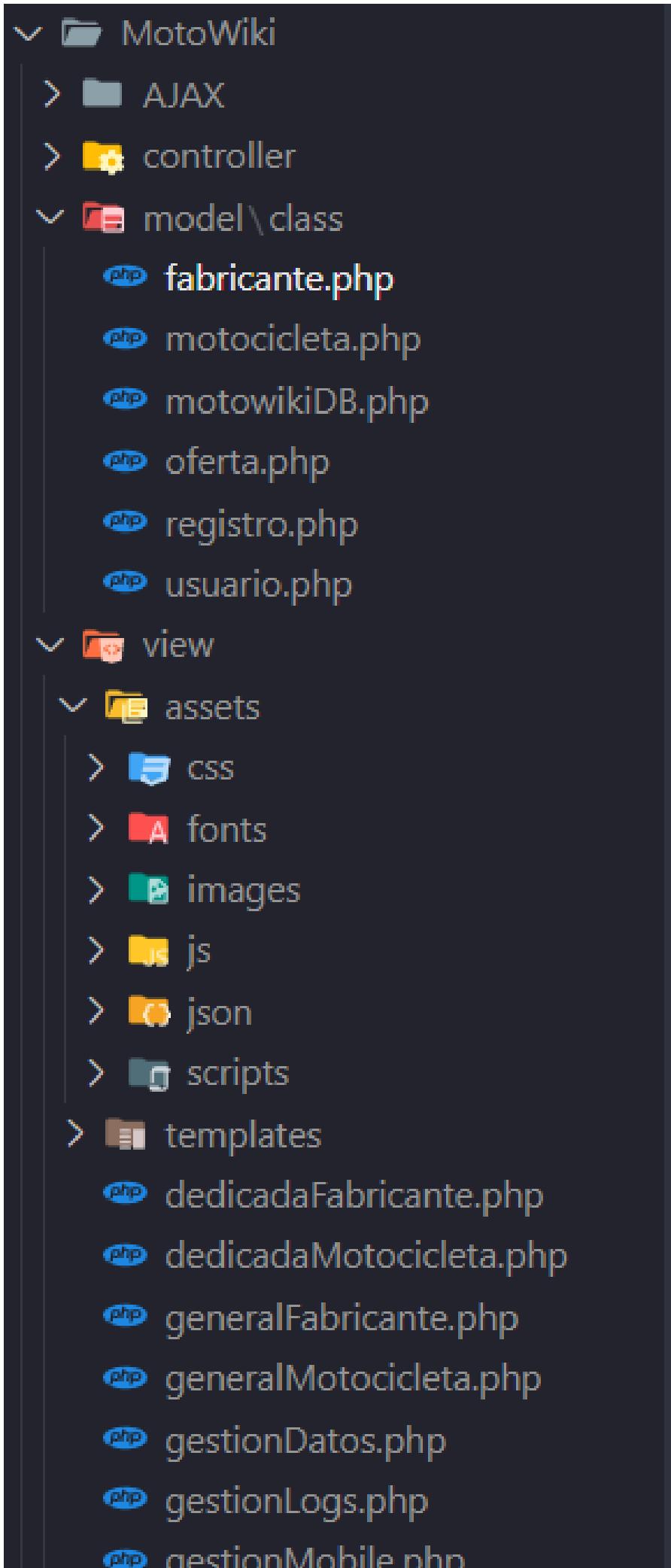


Diseño Lógico

-Modelo Vista Controlador

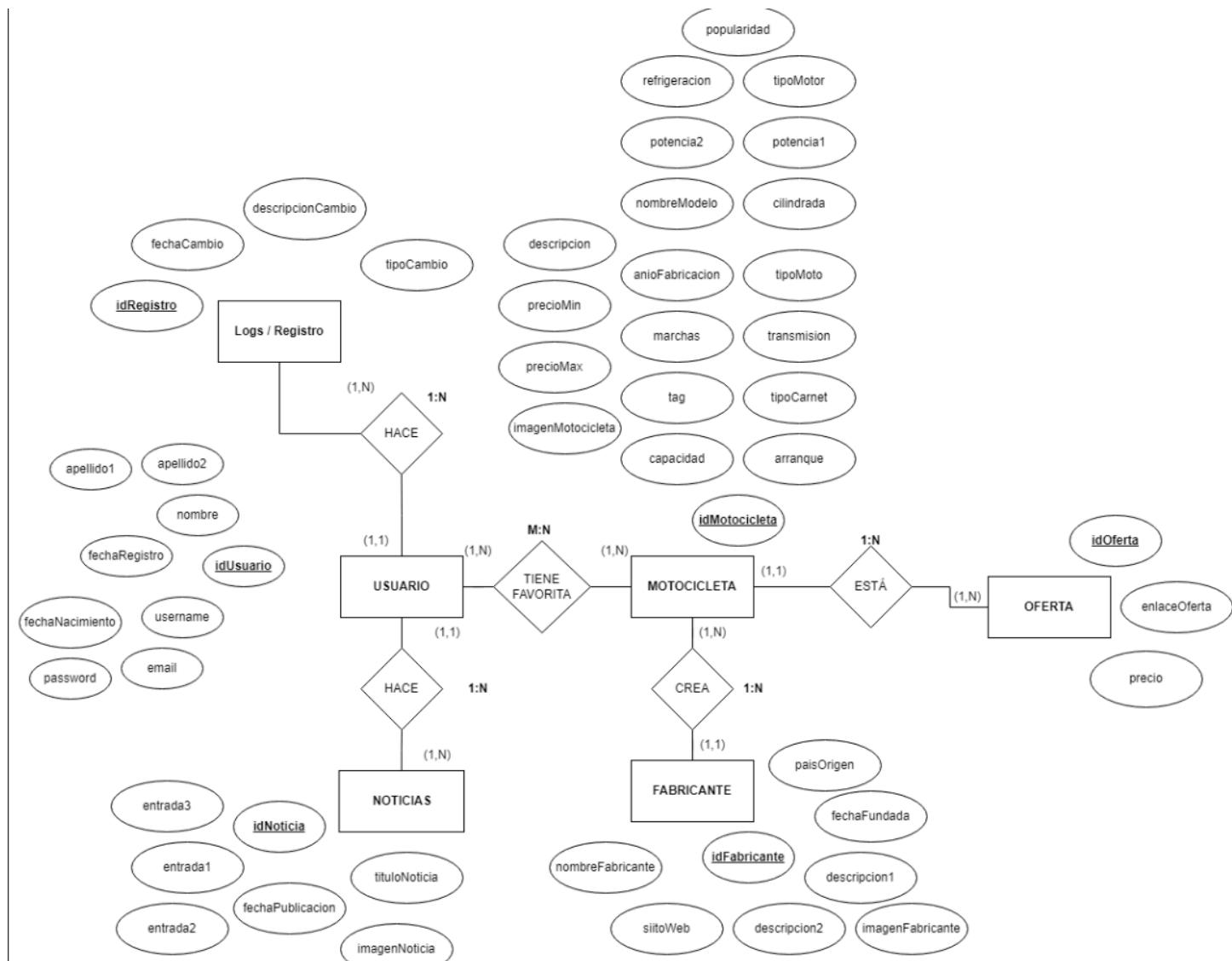
Este modelo vista controlador incluye la carpeta AJAX, añadida por mi, en esta carpeta se incluyen todos los archivos PHP, que hacen alguna función similar a la de

un controlador y los archivos a los que se les hacen peticiones AJAX, dejando de esta forma algo más limpio la carpeta controlar con los controladores reales.



- `gestionUsuarios.php`
- `inicio.php`
- `perfilUsuario.php`
- `registro-inicioSesion.php`
- `index.php`

-Modelo Relacional y Entidad Relación



Favoritas	
FK	<u>idUsuario</u>
FK	<u>idMotocicleta</u>

Logs / Registro	
PK	<u>idRegistro</u>
	fechaCambio
	tipoCambio
	descripcionCambios
FK	<u>idUsuario</u>

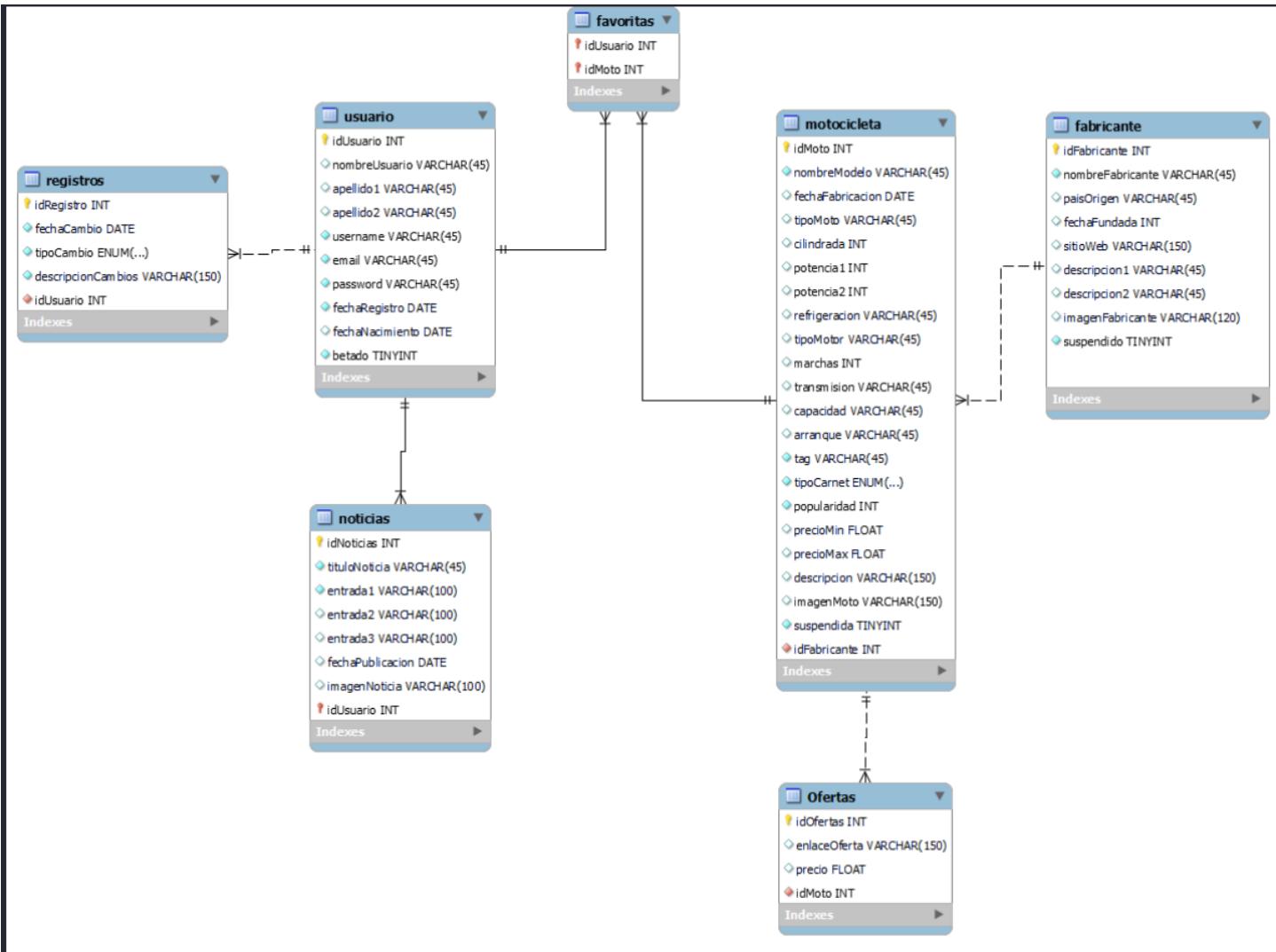
Usuario	
PK	<u>idUsuario</u>
	nombre
	apellido1
	apellido2
	username
	email
	password
	fechaRegistro
	fechaNacimiento

Fabricante	
PK	<u>idMarca</u>
	nombreMarca
	paisOrigen
	fechaFundada
	sitioWeb
	descripcion1
	descripcion2
	imagenFabricante

Motocicleta	
PK	<u>idMotocicleta</u>
	nombreModelo
	anioFabricacion
	tipoMoto
	cilindrada
	potencia1 (CV)
	potencia2 (kW)
	refrigeracion
	tipoMotor
	marchas
	transmision
	capacidad
	arranque
	tag
	tipoCarnet
	popularidad
	precioMin
	precioMax
	descripcion
	imagenMotocicleta
FK	<u>idfabricante</u>

Noticia	
PK	<u>idNoticia</u>
	tituloNoticia
	entrada1
	entrada2
	entrada3
	fechaPublicacion
	imagenNoticia
FK	<u>idUsuario</u>

Ofertas	
PK	<u>idOferta</u>
	enlaceOferta
	precio
FK	<u>idMotocicleta</u>



- Configuración VirtualHost

Para la configuración del VirtualHost tenemos que hacer 3 pasos.

-Primero crear nuestro archivo de configuracion en /etc/apache2/sites-enabled o en sites-available y activarlo con el comando a2enable y configurarlo a nuestro antojo.

```

david@david-VirtualBox: ~
GNU nano 6.2                               /etc/apache2/sites-enabled/motowiki.es.conf *
<VirtualHost *:80>
    ServerName motowiki.es
    ServerAlias www.motowiki.es

    DocumentRoot /var/www/MotoWiki
    DirectoryIndex index.php

</VirtualHost>

```

-Despues desactivaremos el sitio default y crearemos o subiremos nuestro proyecto a la carpeta especificada en el DocumentRoot, en mi caso en /var/www/MotoWiki.

david@david-VirtualBox: ~

```
david@david-VirtualBox:~$ sudo ls /var/www  
html MotoWiki
```

Y finalmente editamos el archivo /etc/hosts para añadirle la redirección en local de la dirección a la IP correspondiente (al ser en local 127.0.0.1)

david@david-VirtualBox: ~

```
GNU nano 6.2                                     /etc/hosts  
127.0.0.1      motowiki.es www.motowiki.es  
127.0.1.1      david-VirtualBox  
192.168.1.100   www.sitioejemplo.com  
172.17.0.2      www.proyectoPHP.com  
# The following lines are desirable for IPv6 capable hosts  
::1      ip6-localhost ip6-loopback  
fe00::0 ip6-localnet  
ff00::0 ip6-mcastprefix  
ff02::1 ip6-allnodes  
ff02::2 ip6-allrouters
```

GUIA DE ESTILO

En este apartado se comentará por encima la guía de Estilo.

Se incluirán los apartados más destacables, y se indicará donde se ubica la guía completa.

Dentro de los archivos se incluirán 2 Guías de Estilos, la primera que tendrá la Guía de estilos Original, y la segunda que tendrá algunos cambios más cercanos a lo definitivo.

La paleta de colores principal es la siguiente:



Aunque en algunos puntos de la página se usan otros colores para textos o botones como los siguientes:



A continuación veremos algunos de los iconos, componentes y el logo del proyecto:



PERFIL MARCAS MOTOS

ID MOTO	MODELO	FECHA FABRICACION	TIPO MOTO	CILINDRADA
1402	S 1000 RR	2022	Sport	999

MODELOS SIMILARES



YBR 125 ED



YBR 125 ED



YBR 125 ED



YBR 125 ED



Busqueda por Motocicleta

Estos son algunos de los componentes más importantes de la web, buscador, módulos, tablas o la barra navegadora.

El resto de la GUIA DE ESTILOS se encuentra dentro del zip con el proyecto, exactamente en la carpeta de ‘ ./Info Extra/GUIA DE ESTILOS/ ’

PINCHA AQUI PARA VER EL [WIREFRAME COMPLETO](#) .

ASPECTOS DESTACABLES DEL PROYECTO

En este apartado podremos ver algunos de los aspectos más destacables del proyecto junto con algo de su código.

- Primero de todo mencionar que la página cuenta con un total de 7 vistas para el usuario normal y 3 vistas extras para los administradores (2 para colaboradores).

Las vistas son:

- Inicio
- Registro/Inicio Sesión
- Perfil
- General + Dedicada Motocicleta
- General + Dedicada Fabricante

Para colaboradores / administradores:

- Gestión de Usuarios (NO COLABORADORES)
- Gestión de Datos
- Gestión de Registros

A continuación iremos hablando de los puntos más destacables de las vistas mencionadas.

Inicio

Inicio contiene una pequeña presentación de lo que será la web, incluye, además de la barra navegador que nos acompañará siempre el Buscador de Motos / Fabricantes. Este funciona mediante una petición fetch a otro archivo, el cuál devolverá y activara el contenedor con los resultados posibles de la búsqueda.

Hay que comentar que hay variables de endpoint para este buscador, dependiendo si busca usuarios, o motocicletas, fabricantes, ambas o incluso si la respuesta tiene que estar adaptada a las vistas.



```

function llamadaConsultaBusqueda(modo,string){
    data = {
        modo: modo,
        textoBusqueda: string
    };

    url = "../AJAX/busquedaMotocicletaMarca.php"

    fetch(url, {
        method: 'POST', // Método de la solicitud
        headers: {
            'Content-Type': 'form-data' // Indica que se enviarán datos en formato JSON
        },
        body: JSON.stringify(data) // Convertir el objeto de datos a una cadena JSON
    })

    .then(response => response.text())

    .then(responseData =>{
        console.log(responseData)
        // console.log("Hola")
        document.getElementById("contenedorResultados").innerHTML =responseData

        /* Creamos divs y los metemos en el contenedor de Resultados. */
    })
}

```



busquedaMotocicletaMarca.php

```

MotoWiki > AJAX > busquedaMotocicletaMarca.php > ...
40         exit;
41     }
42
43
44
45
46 if(isset($datosEnviados['gestionDatos']) && $datosEnviados['gestionDatos'] == true){
47     if(isset($result1)){
48         while ($row = $result1->fetch_assoc()) {
49             $contenidoMostrado .= '<div class="resultadoBuscador" onclick="crearRecuadrosGestionDatos(`moto`, `'.$row['idMoto'].'`)">' . $row['nombreMoto'] . '</div>';
50         }
51     }
52
53     if(isset($result2)){
54         while ($row = $result2->fetch_assoc()) {
55             $contenidoMostrado .= '<div class="resultadoBuscador" onclick="crearRecuadrosGestionDatos(`fabricante`, `'.$row['idFabricante'].'`)">' . $row['nombreFabricante'] . '</div>';
56         }
57     }
58 }else{
59     if(isset($result1)){
60         while ($row = $result1->fetch_assoc()) {
61             $contenidoMostrado .= '<div class="resultadoBuscador" data-src="./motocicleta.php?idMoto='.$row['idMoto'].'">' . $row['nombreMoto'] . '</div>';
62         }
63     }
64
65     if(isset($result2)){
66         while ($row = $result2->fetch_assoc()) {
67             $contenidoMostrado .= '<div class="resultadoBuscador" data-src="./fabricante.php?idFabricante='.$row['idFabricante'].'">' . $row['nombreFabricante'] . '</div>';
68         }
69     }
70 }

```

Justo después encontramos otro de los componentes que nos acompañara por gran parte de la página.

Funcionan de manera que se llama a una función con distintos parámetros y devuelve una lista de motocicletas, con las cuales se crea un contenedor con su información y su enlace.

Incluye el toggle de Favoritos para los usuarios registrados, los usuarios no registrados serán redireccionados al Registro.

 **Administrador**

[PERFIL](#) [MARCAS](#) [M](#)

Últimas Novedades



Meteor 350 **Interceptor 650** **Himalayan** **Continental GT** **Classic 350**

Más populares



```
<main class="inicioContenido">

    <section class="moduloHorizontal">
        <h2>Últimas Novedades</h2>
        <hr>

        <div class="contenedorTarjetas">

            <?= Motocicleta::generarModulo($modo="nuevas"); ?>

        </div>

    </section>

    <section class="moduloHorizontal">
        <h2>Más populares</h2>
        <hr>

        <div class="contenedorTarjetas">

            <?= Motocicleta::generarModulo($modo="populares"); ?>

        </div>
    </section>

    <section class="moduloHorizontal">
        <h2>Más Baratas</h2>
        <hr>

        <div class="contenedorTarjetas">

            <?= Motocicleta::generarModulo($modo="baratas"); ?>

        </div>
    </section>
```

```
motocicleta.php ×
MotoWiki > model > class > motocicleta.php > PHP Intelephense > Motocicleta > obtenerNuevas
6  class Motocicleta {
    ...
329      public static function generarModulo($modo,$Fabricante = null,$idUsuario = null,$objetoMoto = null){
330          ...
331          if (!in_array(strtolower($modo), ['populares', 'nuevas','baratas','favoritas','similares'])) {
332              throw new InvalidArgumentException('El modo no es correcto.');
333          }
334
335          if(!empty($_SESSION)){
336              $idUsuario = $_SESSION['idUser'];
337          }
338          // print_r($modo);
339          // print_r($objetoMoto);
340
341          $motosModulo = match ($modo) {
342              "populares" => Motocicleta::obtenerPopulares(isset($Fabricante) ? $idFabricante = $Fabricante : $idFabricante = null),
343              "nuevas" => Motocicleta::obtenerNuevas(isset($Fabricante) ? $idFabricante = $Fabricante : $idFabricante = null),
344              "baratas" => Motocicleta::obtenerPorPrecio("DESC",$Fabricante),
345              "favoritas" => ($idUsuario == null) ? "" : Usuario::obtenerTusFavoritas($idUsuario),
346              "similares" => ($objetoMoto == null) ? "" : $objetoMoto->modelosSimilares(),
347          };
348
349          $tarjetas = "";
350
351          // print_r($motosModulo);
352
353          if($motosModulo != "" && $motosModulo != false){
354              foreach ($motosModulo as $key => $objetoMotoBucle) {
355
356                  if(isset($_GET['idMoto']) && $_GET['idMoto'] == $objetoMotoBucle->__get("idMoto")){
357
358                      }else{
359                          $tarjetas .= '<div class="tarjetaMotoMarca">
360                              <div class="contenedorImagenMotoMarca">
361                                  '.(empty($_SESSION) ? '<i class="fa-regular fa-star" data-src="./registro-inicioSesion.php?&idMoto=' . $objetoMotoBucle->__get("idMoto") . '" loading="lazy" alt="FotoMotocicleta">' :
362                                      '' .
363                                      '<a href="./motocicleta.php?idMoto='.$objetoMotoBucle->__get("idMoto").'">' .<h2>' . $objetoMotoBucle->__get("nombre") . '</h2>' .
364                                      '</a>' .
365                              '</div>';
366                      }
367                  }
368
369          }
370
371          $tarjetas .= '<div class="tarjetaMotoMarca">
372              <div class="contenedorImagenMotoMarca">
373                  '.(empty($_SESSION) ? '<i class="fa-regular fa-star" data-src="./registro-inicioSesion.php?&idMoto=' . $objetoMotoBucle->__get("idMoto") . '" loading="lazy" alt="FotoMotocicleta">' :
374                                      '' .
375                                      '<a href="./motocicleta.php?idMoto='.$objetoMotoBucle->__get("idMoto").'">' .<h2>' . $objetoMotoBucle->__get("nombre") . '</h2>' .
376                                      '</a>' .
377                              '</div>';
378
379          
```

Perfil y Registro

Tanto el perfil como el registro son páginas sencillas, cuentan con la información necesaria y poco más.

El Perfil cuenta con el botón de cerrar sesión y editar tus datos, que incluye un pequeño pop-up y un módulo con tus motos favoritas.



PERFIL DE USUARIO

TUS DATOS

Username
davidTestAdmin

Nombre
davidTestAdmin

Log Out

Tipo de Usuario
admin

Fecha Nacimiento
2000-10-

Tus Favoritas



Motos

Scooter

Touring

Scooter

Nombre X

Apellido 1:

Apellido 2:

Username:

Email:

Nueva Contraseña:

Fecha Nacimiento (formato: dd/mm/aaaa)

Confirmar Cambios

Mientras tanto el Registro cuenta con un botón para cambiar entre el Registro y el Inicio de Sesión, estos tienen sus comprobaciones y expresiones regulares.

REGISTRO

[INICIAR SESIÓN](#)

Nombre

Input Campo

Username

Input Campo

Apellido 1

Input Campo

Apellido 2

Input Campo

Contraseña

Input Campo

Confirmar Contraseña

Input Campo

Email

Input Campo

Fecha Nacimiento

dd/mm/aaaa



REGISTRAR USUARIO

General Motocicleta - Fabricante

Estas páginas son algo más importantes, incluyen en ambos casos un buscador del respectivo tipo.

En el caso de Fabricante se muestran todos, ya que no hay una cantidad considerable como para crear un mal funcionamiento de la página.

Mientras que en el caso de Motocicleta hay un sistema de Paginación hecho con JavaScript mediante peticiones fetch. Al final de la página hay dos botones, que realizan la misma petición pero cambiando el índice de la paginación.

TODOS LOS FABRICANTES

Busqueda por Fabricante



Kawasaki



Honda



Yamaha



Suzuki



Ducati



KTM



Aprilia



Royal Enfield



Harley-Davidson



BMW Motorrad

TODAS LAS MOTOS

Busqueda por Motocicleta



S 1000 R



R18 The Wall



R nineT Scram



S 1000 RR



C 400 GT



Tri Glide Ultra



Tricity 155



YBR125 ED



C 400 X



S 1000 XR

```

document.getElementById("botonDespues").addEventListener("click", (ev)=>{
    paginacion++;
    document.getElementById("contenedorGeneral").innerHTML = "";

    const data = {
        paginacion: paginacion,
        cantidad: cantidadMotos
    };
    fetch('../AJAX/generarTarjetasPaginadas.php', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify(data) // Convertir los datos a formato JSON
    })
    .then(response => response.text())
    .then(data => {
        console.log(data);
        if(data == "" || data == null){
            paginacion--;
        }else{
            document.getElementById("contenedorGeneral").innerHTML = data;
            document.getElementById("parrafoPaginacion").innerHTML = paginacion;
        }
    })
    .catch((error) => {
        console.error('Error:', error);
    });
});

```

Dedicada Motocicleta - Fabricante

Estas páginas se podrían decir que son casi identicas.

Las diferencias Reales de estas páginas son:

En fabricante puede haber 2 descripciones.

En motocicleta podemos Ver las Ofertas de éstas, un selector de variantes de la motocicleta y una tabla con sus respectivos datos, en caso de ser administrador tenemos la posibilidad de añadir ofertas.

Además en el apartado de Motocicletas se incluye la funcionalidad del TAG, haciendo que varias motocicletas estén conectadas y salgan en VARIANTES con sus enlaces, sirve sobre todo para motocicletas MUY SIMILARES o mismo modelo de años distintos.

DATOS PRINCIPALES

ID MOTO	MODELO	FECHA FABRICACION	TIPO MOTO	CILINDRADA	POTENCIA (HP)	POTENCIA (KW)	REFRIGERACIÓN
1398	R nineT Scrambler	2022	Naked bike	1170	109	80	Oil & air



A side-profile photograph of a BMW R nineT Scrambler motorcycle. The bike features a dark grey frame, black leather seat, and black wheels with multi-spoke designs. It has a prominent front fender and a minimalist, rugged aesthetic.

Ver Ofertas

VARIANTES DEL MODELO

YBR125 ED

YBR125

YBR125 Custom

Honda

*SEDE:
Japón*

Fundada: 1948



Honda es una influyente marca japonesa de motocicletas y automóviles, fundada en 1948. Es conocida por su fiabilidad, innovación y diseño eficiente. Honda ha sido pionera en la implementación de tecnologías avanzadas y sostenibles, como los motores híbridos y eléctricos. Sus motocicletas abarcan desde modelos deportivos hasta scooters y touring, ganando numerosos campeonatos en competiciones de motociclismo. Honda se distingue por su compromiso con la excelencia y la satisfacción del cliente, siendo una de las marcas más confiables y respetadas del mundo.

Honda, una marca japonesa líder en motocicletas y automóviles desde 1948, destaca por su fiabilidad, innovación y diseño eficiente. Reconocida por su compromiso con la excelencia y la satisfacción del cliente, Honda es pionera en tecnologías avanzadas y sostenibles, incluidos los motores híbridos y eléctricos. Su amplia gama de motocicletas, desde deportivas hasta scooters y touring, ha ganado numerosos campeonatos en competiciones de motociclismo, consolidando su posición como una marca respetada y confiable en todo el mundo.

Últimas Novedades



Ofertas de la Moto

X

PRECIO	ENLACE	
1000 €	ENLACE 1	

Crear Nueva Oferta

Ofertas de la Moto

X

Precio:

Enlace:

Confirmar Oferta

ANOTACION: Hay muchas de las motocicletas que no tienen descripción o imagen, la página esta adaptada para evitar estos errores y darle una salida, en los textos con uno provisional y en las imágenes una adaptación por el tipo de Moto que sea.

Gestion Usuarios

Esta página también es sencilla, pero muy importante, todos sus cambios se hacen por peticiones fetch, y solo pueden entrar los administradores.

Tenemos las opciones de activar o vetar un usuario, y la de cambiarle los permisos a estos.

The screenshot shows a list of users under the heading 'GESTION DE USUARIOS'. Each user entry includes their ID, name, current role, and a dropdown menu for changing roles or a red 'VETAR' button. The users listed are:

ID	Nombre	Role	Opciones
(ID: 11)	davidTestAdmin	Administrador	VETAR
(ID: 12)	davidTestColab	Colaborador	VETAR
(ID: 13)	davidTestUser	Usuario	VETAR
(ID: 14)	davidraTestUser	Colaborador	ACTIVAR
(ID: 15)	dawAdmin123	Administrador	VETAR
(ID: 16)	dawColab123	Colaborador	VETAR
(ID: 17)	dawUser123	Usuario	VETAR

```
function cambiarPermisos(){      You, hace 6 días • Cambiar Permisos Funciona
    todosSelects = document.querySelectorAll(".selectUsuario");

    console.log(todosSelects);

    todosSelects.forEach(element => {
        url = "../AJAX/cambioPermisos.php"
        element.addEventListener("change", (ev)=>{
            idUsuario = ev.target.getAttribute("data-idUser");
            permisos = ev.target.value;

            fetch(url, {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({ idUsuario: idUsuario,permisos: permisos })
            })
            .then(response => response.text())
            .then(responseData =>{
                console.log(responseData);
            })
        })
    });
}
```

```

function toggleVetado(button) {      You, hace 6 días • Vetado Funciona
💡
    url = "../AJAX/vetar-activarUsuario.php";
    idUsuario = button.getAttribute("data-idUser");

    console.log(idUsuario);

    if(button.classList.contains("botonAzul")){
        button.classList.remove("botonAzul");
        button.classList.add("botonNaranja");
        button.innerText = "VETAR"

        fetch(url, {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({ idUsuario: idUsuario, modo: "activar" })
        })
        .then(response => response.text())
        .then(responseData =>{
            console.log(responseData);
        })
    }

    }else if(button.classList.contains("botonNaranja")){
        button.classList.remove("botonNaranja");
        button.classList.add("botonAzul");
        button.innerText = "ACTIVAR"

        fetch(url, {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({ idUsuario: idUsuario, modo: "vetar" })
        })
        .then(response => response.text())
        .then(responseData =>{
            console.log(responseData);
        })
    }
}

```

Gestion Datos

Esta se podría decir que es la página más compleja, por las complicaciones que tiene de cara a comprobaciones, evitar errores y el backend.

Cuenta con 4 Funcionalidades:

- Crear Nueva Moto
- Crear Nuevo Fabricante

- Editar Moto Existente
- Editar Fabricante Existente

(Incluyendo vetado de Motocicletas, para que estas no se vean en caso de tener algún error, e incluso borrarlas si hiciera falta, incluye Fabricante).

Todos estos dentro de un formulario que se activa y rellena dependiendo del caso, incluye modificación de imagen mostrándola, reusado de proyectos anteriores.

La verdadera complicación de la página es la anidación de fetch y evitar problemas de asincronía y de elementos inexistentes en cuanto a los listeners.

Cuando se hace el cambio el botón de Confirmar cambia su texto para confirmar los cambios.

Nueva Moto
Nuevo Fabricante



Nombre Fabricante:

Pais Origen

Fecha Fundada

Sitio Web

DESCRIPCION 1

DESCRIPCION 2

Cancelar
Crear Nuevo Fabricante

Precio Maximo:

Precio Maximo

Altura:

1125

Peso:

238,1

Fabricante:

Kawasaki

FECHA FABRICACIÓN:

2022

DESCRIPCION 1

Ninja H2 Carbon: Versión exclusiva del H2 con carrocería de fibra de carbono.

Borrar Motocicleta

Suspender Motocicleta

Confirmar Cambios

```
function crearRecuadrosGestionDatos(modo,idCambio = null){
    data = {
        modo: modo, /* moto / fabricante */
        idCambio: idCambio
    };
    url = "../AJAX/recuadroCambios.php"

    fetch(url, {
        method: 'POST', // Método de la solicitud
        headers: {
            'Content-Type': 'form-data' // Indica que se enviarán datos en formato JSON
        },
        body: JSON.stringify(data) // Convertir el objeto de datos a una cadena JSON
    })

    .then(response => response.text())

    .then(responseData =>{
        console.log(responseData)

        document.getElementById("contenedorResultados").hidden = true;
        document.getElementById("contenedorCambios").hidden = false;
        document.getElementById("contenedorCambios").innerHTML =responseData

        crearListenerCambioImagen()

        if(document.getElementById("botonConfirmarCambios")){
            document.getElementById("botonConfirmarCambios").addEventListener("click",(ev)=>{
                formulario=document.getElementById("formularioDatos");

                formData = new FormData(formulario);

                fetch("../AJAX/confirmarCambios.php",{
                    method: 'POST',
                    body: formData
                })
                .then(response => response.text())
                .then(data => {
                    console.log(data);

                    // Cambiar el texto del botón
                    document.getElementById("botonConfirmarCambios").innerText = "Datos Cambiados";

                    // Restaurar el texto original después de 3 segundos
                    setTimeout(function(){
                        document.getElementById("botonConfirmarCambios").innerText = "Confirmar Cambios";
                    }, 3000);

                })
                .catch(error) => {
                    console.error('Error:', error);
                });
            });
        }
    })
}
```

```

function crearRecuadrosGestionDatos(modo,idCambio = null){
    .then(responseData =>{

        if(document.getElementById("generarNuevo")){
            document.getElementById("generarNuevo").addEventListener("click",(ev)=>{
                formulario=document.getElementById("formularioDatos");

                formData = new FormData(formulario);

                fetch("../AJAX/crearMotoFabricante.php",{
                    method: 'POST',
                    body: formData
                })
                .then(response => response.text())
                .then(data => {
                    console.log(data);
                    location.reload()      You, ayer • Terminada Lógica
                })
                .catch((error) => {
                    console.error('Error:', error);
                });
            });
        }

        if(document.getElementById("botonBorrar")){
            document.getElementById("botonBorrar").addEventListener("click",(ev)=>{

                fetch("../AJAX/borrarDatos.php", {
                    method: 'POST', // Método de la solicitud
                    headers: {
                        'Content-Type': 'form-data' // Indica que se enviarán datos en formato JSON
                    },
                    body: JSON.stringify(data) // Convertir el objeto de datos a una cadena JSON
                })

                .then(response => response.text())
                .then(data => {
                    console.log(data);
                    location.reload()
                })
                .catch((error) => {
                    console.error('Error:', error);
                });
            });
        }

        if(document.getElementById("botonCancelar")){
            document.getElementById("botonCancelar").addEventListener("click",(ev)=>{
                document.getElementById("contenedorCambios").innerHTML = "";
                document.getElementById("contenedorCambios").hidden = true;
            });
        }
    })
}

```

Registros de Modificaciones

Esta es la última página en la que se muestran una serie de registros, para en caso de haber muchos de estos no saturar la carga de la página.

Incluye un botón que carga todos los registros.

REGISTRO DE MODIFICACIONES				
VER TODOS LOS REGISTROS				
ID USUARIO	NOMBRE	CAMBIO	TIPO CAMBIO	FECHA CAMBIO
11	davidTestAdmin	Se han modificado los datos del Fabricante: 4	fabricante	2024-05-31
11	davidTestAdmin	Se han modificado los datos del Fabricante: 8	fabricante	2024-05-31
11	davidTestAdmin	Se han modificado los datos del Fabricante: 6	fabricante	2024-05-31
11	davidTestAdmin	Se han modificado los datos del Fabricante: 6	fabricante	2024-05-31
11	davidTestAdmin	Se han modificado los datos del Fabricante: 1	fabricante	2024-05-31
11	davidTestAdmin	Se han modificado los datos del Fabricante: 10	fabricante	2024-05-31
11	davidTestAdmin	Se han modificado los datos del Fabricante: 7	fabricante	2024-05-31
11	davidTestAdmin	Se han modificado los datos del Fabricante: 9	fabricante	2024-05-31
11	davidTestAdmin	Se han modificado los datos del Fabricante: 9	fabricante	2024-05-31
11	davidTestAdmin	Se han modificado los datos del Fabricante: 5	fabricante	2024-05-31
11	davidTestAdmin	Se han modificado los datos del Fabricante: 2	fabricante	2024-05-31
11	davidTestAdmin	Se han modificado los datos del Fabricante: 3	fabricante	2024-05-31
11	davidTestAdmin	Se ha cambiado los permisos a admin al usuario: 15	usuario	2024-05-31
11	davidTestAdmin	Se ha cambiado los permisos a user al usuario: 15	usuario	2024-05-31

OTROS PUNTOS A DESTACAR

Recogida de Datos, Imágenes etc.

La API usada para la recogida de información ha sido la sección de Motocicletas de NINJA APIs.

Motorcycles API

[API Directory](#)[Aircraft](#)[Airlines](#)[Airports](#)[Air Quality](#)[Animals](#)[Baby Names](#)[Bucket List](#)[Calories Burned](#)[Cars](#)[Cats](#)[Celebrity](#)[Chuck Norris](#)[City](#)[Cocktail](#)[Commodity Price New](#)

The Motorcycles API provides highly-detailed technical data on tens of thousands of different motorcycle models from hundreds of manufacturers.

/v1/motorcycles

HTTP GET

Returns up to 30 motorcycle results matching the input name parameters. For searches that yield > 30 results, please use the `offset` parameter.

Parameters

Either `make` or `model` must be set.

`make` (optional) - name of manufacturer/brand. Supports partial matching (e.g. `Harley` will match `Harley-Davidson`).

`model` (optional) - name of motorcycle model. Supports partial matching (e.g. `Ninja` will match `Ninja 650`).

`year` (optional) - release year of motorcycle model. Must be in the form of `YYYY` (e.g. `2022`).

La recogida de datos se ha hecho mediante un archivo PHP, al cuál se le cambiaban una serie de parámetros para encajar en las consultas y introducir la información necesaria a la Base de Datos propia del proyecto.

En total se han recogido alrededor de 700 Motocicletas de unos 10 Fabricantes.

```

script_trerMotocicletas.php M X
pruebasAPI > scripts > script_trerMotocicletas.php > ...
7  <?php
8
9  $conexion = mysqli_connect("localhost","root","","motowiki");
10
11 // for ($i = 0; $i < 300; $i += 30) {
12
13 $key="XkPnmvHTmjVd0veY4ZQEcw==9XPtzIVTabrW67Yd";
14
15 // echo "<br>$i<br>";
16
17 |                                     /* CAMBIAR MAKE + OFFSET EN CASO DE SER NECESARIO */ You, aho
18 $url="https://api.api-ninjas.com/v1/motorcycles?make=Enfield&offset=60";
19
20 // Inicializar cURL
21 $curl = curl_init($url);
22
23 // Configurar opciones de cURL
24 curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
25 curl_setopt($curl, CURLOPT_HTTPHEADER, array(
26   'X-Api-Key: '.$key, // Reemplaza YOUR_API_KEY_HERE con tu clave API
27 ));
28
29 // Ejecutar la solicitud y obtener la respuesta
30 $response = curl_exec($curl);
31
32 // Verificar si hay errores
33 if(curl_errno($curl)){
34   echo 'Error:' . curl_error($curl);
35 }
36
37 // Cerrar cURL
38 curl_close($curl);
39
40 // Imprimir la respuesta
41 $motos=json_decode($response,true);
42

```

En cambio las imágenes se han intentado traer de distintas formas, mediante Scripts en python y php, pero no ha funcionado nada, por lo que al no tener imágenes la API usada se han recogido una serie de imágenes manualmente. Todas las motocicletas que no tienen imagen propia tendrán una por defecto.

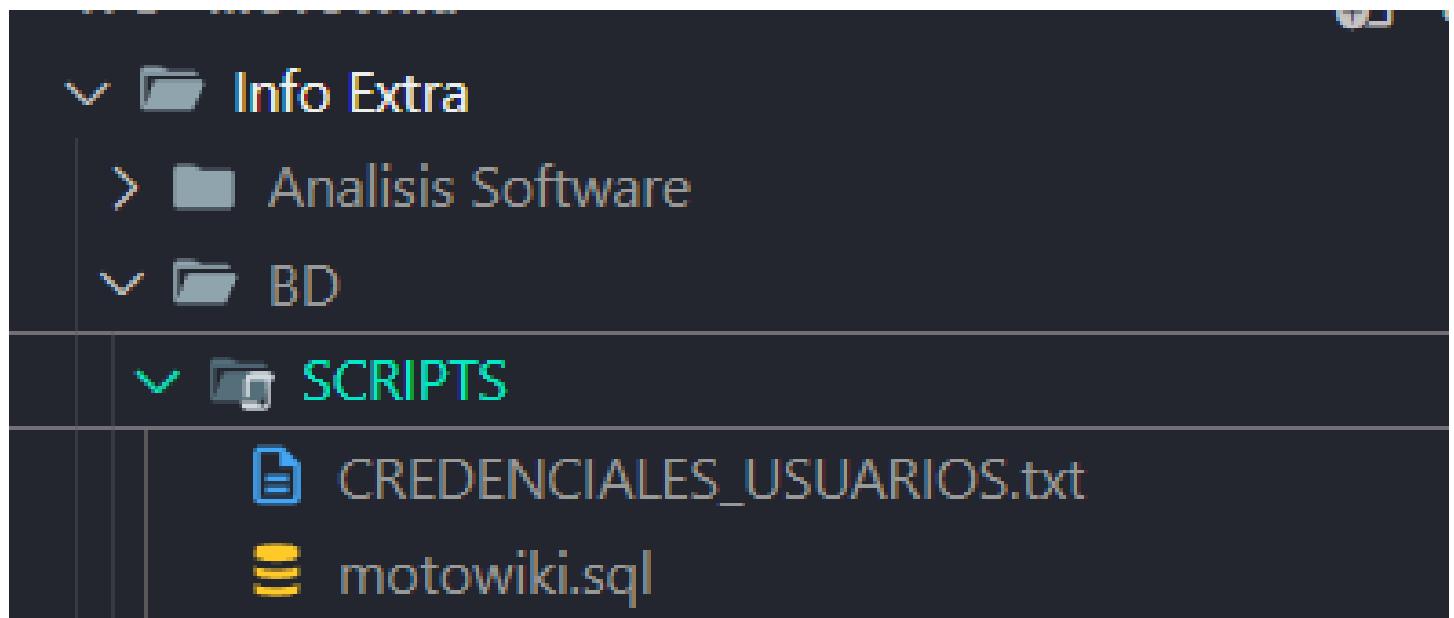
PONER EN MARCHA EN LOCAL

Aquí vamos a explicar los 3 pasos que hay que hacer para poner en marcha el proyecto en local.

Primero antes de nada descargaremos el .zip, el cuál incluye una serie de carpetas (3), que incluyen datos del proyecto y las entregas (Info Extra), el PROYECTO como tal (MotoWiki), y una carpeta con pruebas (pruebasAPI).

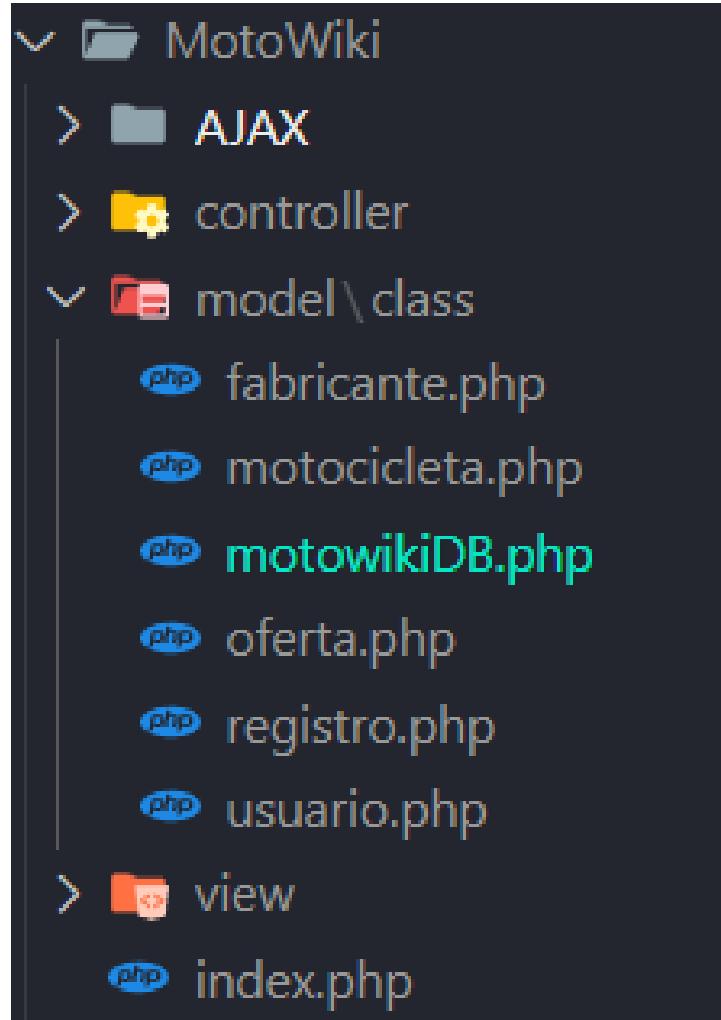
1- EJECUTAR SCRIPT ‘motowii.sql’ EN SGBD (phpmyAdmin por ejemplo).

Inlcuye un archivo txt con las credenciales de los usuarios existentes para hacer pruebas.



2 - CAMBIAR CONFIGURACION DE CONEXION A LA BASE DE DATOS.

Pondriamos en la clase de MotoWikiDB.



motowikiDB.php X

MotoWiki > model > class > motowikiDB.php > PHP Intelephense > MotowikiDB > conexiónDB

```
4 class MotowikiDB{
    public static function conexiónDB(){
        $usuario = "root";
        $password = "";
        $db = "motowiki";
        $dirección = "localhost";      DavidCIFP01, hace 2 semanas • Creando Clases + S
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38 }
```

3- FINALMENTE EJECUTARIAMOS EN NUESTRO SERVER EL ARCHIVO INDEX.PHP

The screenshot shows a code editor interface with a sidebar labeled "EXPLORADOR" containing a tree view of the project structure:

- TFG - MOT... (selected)
- Info Extra
- MotoWiki (expanded)
 - AJAX
 - controller
 - model\class
 - view
- index.php

The main pane shows the contents of the selected file, "index.php":

```
1 <?php
2
3
4 header("Location: ./controller/inicio.php");
5
6 ?>
```

Details at the top of the editor pane indicate the file is "index.php" under "MotoWiki", last modified by "DavidCIFP01" a week ago, with 1 author.

Con estos pasos el proyecto debería funcionar correctamente.

(En caso de subir el proyecto a un Hosting solo se subiria la carpeta MotoWiki).