2023

Documentación Aplicación web: Explora

DESARROLLO WEB ENTORNO SERVIDOR

CAROL ANDREA DUARTE HIGUERA

IES SAN ANDRES | DAW2

El Proyecto integrador consiste en hacer una aplicación para búsqueda de rutas de senderismo, en este caso mi aplicación se llama Explora. Para poder realizar diferentes operaciones como registrar usuario (añadirlo en una base de datos), editarlo o borrarlo, revisar rutas, mostrar detalle de rutas se ha hecho uso de una API tipo REST la cuál será brevemente explicada a continuación.

USUARIO Y CONTRASEÑA PARA BASE DE DATOS:

Esta base de datos tiene un usuario ***explora*** que puede acceder, cuya contraseña es la palabra ***explora***.

BASE DE DATOS EXPLORA:

La base de datos EXPLORA tiene dos tablas "routes" y "users". Son las tablas necesarias para que el usuario pueda registrarse, editar su perfil y eliminarlo. Y a su vez pueda acceder a información detallada de las rutas y filtrarlas de acuerdo a sus características.

La tabla **users** tiene los siguientes campos:

	#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
☐	1	**id** 🔑	int(11)			No	*Ninguna*		AUTO_INCREMENT	Cambiar
☐	2	**username** 🔑	varchar(50)	utf8mb4_general_ci		No	*Ninguna*			Cambiar
☐	3	**fullname**	varchar(50)	utf8mb4_general_ci		No	*Ninguna*			Cambiar
☐	4	**pass**	varchar(260)	utf8mb4_general_ci		No	*Ninguna*			Cambiar
☐	5	**email**	varchar(35)	utf8mb4_general_ci		No	*Ninguna*			Cambiar
☐	6	**height**	int(11)			No	*Ninguna*			Cambiar
☐	7	**weight**	int(11)			No	*Ninguna*			Cambiar
☐	8	**birthday**	date			No	*Ninguna*			Cambiar
☐	9	**activities**	varchar(260)	utf8mb4_general_ci		No	*Ninguna*			Cambiar

Tengo 3 usuarios registrados, Para el primero el username: **usuario1** y su contraseña es : **usuario1.**

Los campos son:

Id: un id autoincremental

Username: el nombre de usuario.

Fullname: el nombre complete del usuario.

Pass: la contraseña que se guarda de manera segura por hash();

Email: el correo electrónico del usuario.

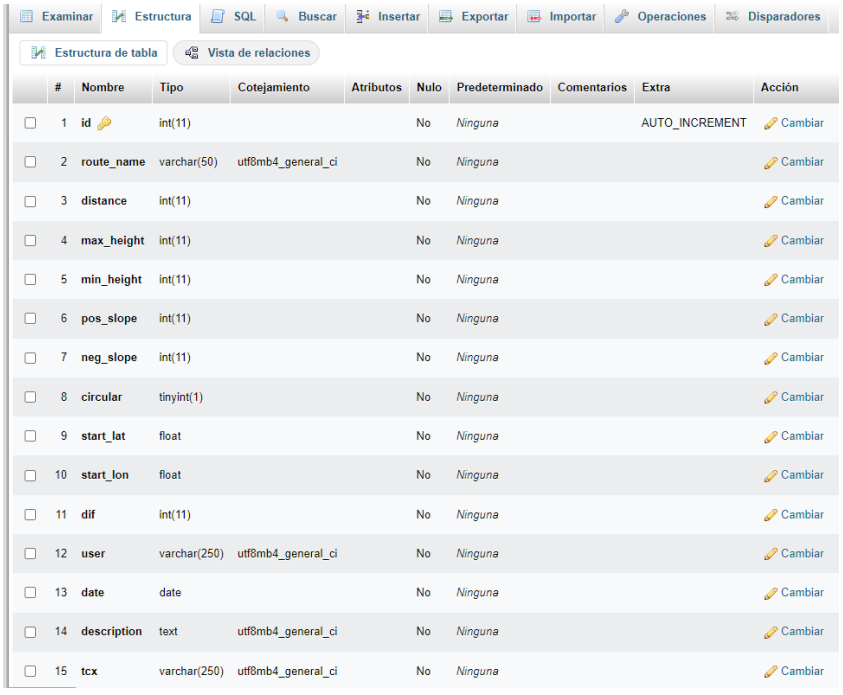
Height: la altura del usuario.

Weight: el peso del usuario.

Birthday: la fecha de nacimiento del usuario.

Activities: las actividades favoritas del usuario.

La tabla **routes** tiene los siguientes campos:



#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar
2	route_name	varchar(50)	utf8mb4_general_ci		No	Ninguna			Cambiar
3	distance	int(11)			No	Ninguna			Cambiar
4	max_height	int(11)			No	Ninguna			Cambiar
5	min_height	int(11)			No	Ninguna			Cambiar
6	pos_slope	int(11)			No	Ninguna			Cambiar
7	neg_slope	int(11)			No	Ninguna			Cambiar
8	circular	tinyint(1)			No	Ninguna			Cambiar
9	start_lat	float			No	Ninguna			Cambiar
10	start_lon	float			No	Ninguna			Cambiar
11	dif	int(11)			No	Ninguna			Cambiar
12	user	varchar(250)	utf8mb4_general_ci		No	Ninguna			Cambiar
13	date	date			No	Ninguna			Cambiar
14	description	text	utf8mb4_general_ci		No	Ninguna			Cambiar
15	tcx	varchar(250)	utf8mb4_general_ci		No	Ninguna			Cambiar

Id: el id de la ruta que en este caso es auto incremental.

Route_name: El nombre de la ruta.

Distance: La distancia total en recorrer la ruta.

Max_height: La altitud máxima.

Min_height la altitud mínima.

Pos_slope: la suma de las distancias ascendidas durante un recorrido.

Neg_slope: la suma de las distancias descendidas durante el recorrido.

Circular: es de tipo booleano, 0 y 1 para indicar si la ruta es circular o no.

Start_lat: La coordenada de latitud de comienzo de la ruta.

Start_lon: La coordenada de longitud de comienzo de la ruta.

Dif: El grado de dificultad de la ruta: de 0 a 5.

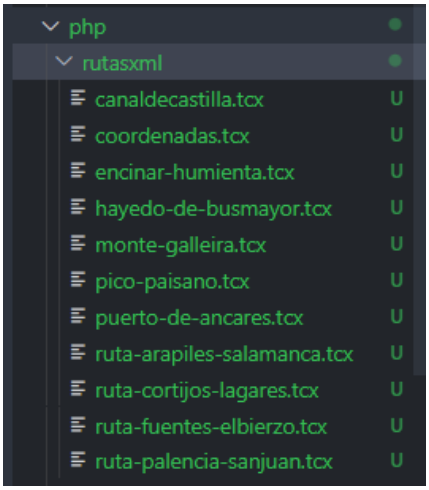
0-Facil 1-Moderado 2-Difícil 3-Muy difícil 4-Expertos

Description: Una descripción breve de la ruta.

User: el nombre de usuario que ha subido la ruta.

Date: La fecha en que sea realizada la ruta.

Tcx: aquí guardo el nombre del archivo que contiene la información de la ruta, en formato xml.



Estos archivos están guardados en la carpeta *php/ rutasxml*., donde tengo los ficheros de extensión tcx que corresponden a cada una de las rutas:

En total tengo 10 registros en la tabla rutas.

Estos archivos .tcx contienen información de las rutas en xml y fueron descargadas de wikiloc.es.

Los archivos TCX son implementadas por el software Garmin Training Center a través de XML bases de datos basados en que se puede utilizar para almacenar datos de fitness, con contenido detallado de latitud y longitud de distintos puntos de las rutas.

Dentro de la carpeta de proyecto Integrador se encuentra la API, que contiene 7 carpetas:

api/clases

En index.php de esta carpeta se encuentra los parámetros de conexión a la base de datos.

api/ users

Desde la cual se puede hacer uso de los métodos 'GET', 'POST' y 'DELETE'. 'GET':
`POST' : se accede a este recurso para editar el perfil del usuario, desde el scrip-perfil-usuario.js.

'DELETE': se accede a este recurso para que el usuario sea capaz de eliminar su propia cuenta desde el script-perfil-usuario.js

api/ routes

'GET' : se accede a este recurso para :

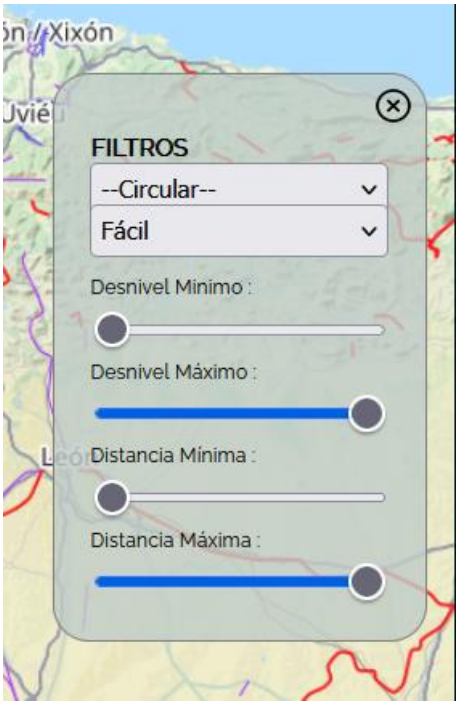
-Cargar las tarjetas con los detalles de las rutas en el script-pagina-principal.js

-Buscar rutas por nombre indicando la ruta desde la landing page.

-Filtrar las rutas desde la página de mis rutas.

-Mostrar más información desde la página de detalle de rutas.

Los parámetros de búsqueda utilizados para los filtros de la página principal y la página de las rutas del usuario son:



- Route_name: Para el nombre de la ruta utilicé 'LIKE '%" . $route_name . "%"', con el fin de que la búsqueda del nombre fuera más flexible y no fuera el nombre completo el que tuviera que mostrar.
- distance: Para obtener la distancia total de la ruta.
- Circular: que devuelve un 0 o 1.
- Dif: que devuelve un número del 0 al 5 en relación a la dificultad de la ruta.
- minDist: en la que se calcula si la distancia es mayor a ese parámetro.
- maxDist: en la que se calcula si la distancia es menor a ese parámetro.
- minSlope: en la que se calcula si la desnivel es mayor a ese parámetro.
- MaxSlope: en la que se calcula si la distancia es menor a ese parámetro.

Para ver el detalle de ruta el usuario debe hacer click en el título de la ruta. Y para acceder a la ventana modal con los filtros debe hacer click en el botón Ver Filtros. En la página de detalle de ruta se utilizan además los parámetros:

- Description: Contiene un texto corto de la ruta
- User: con el id del usuario
- Date: la fecha en que fue realizada la ruta
- Max-height: la altura máxima
- Min-height: la altura mínima.

'POST' que fue definido para hacer pruebas, ya que actualmente no hay una página que se encargue de acceder a este recurso para que el usuario pueda cargar la ruta, en este caso se tendría que dar la opción de que el usuario pudiera subir un fichero gpx con su ruta realizada.

api/checkuser

Cuando el método del servidor es ‘GET’ En este index.php hago una llamada a la uri:

```
http://localhost/dwes/proyectoIntegrador/api/users
```

se comprueba sI se ha enviado un username, y si es así se agrega este parámetro a la uri:

```
$uri .=  "?username=" . $username;
```

Si no hay un resultado quiere decir que el usuario puede escoger este username ya que no se encuentra, de lo contrario, si hay un resultado, se devuelve un array mediante json_encode un mensaje de usuario existente.

api/edituser

El método del servidor es PUT ya que la finalidad es editar los datos del usuario.

El input que se recibe se guarda en la variable $json:

```
$json = json_decode(file_get_contents('php://input'), true);
```

La sentencia SQL en este caso es UPDATE. Con los datos que han sido recogidos.

De ser editado el perfil de usuario el método devuelve el id del usuario en un echo json_encode, para ser recibido en la parte de cliente.

api/login

El método utilizado es `POST`y su objetivo es:

1. Comprobar que el username y el pass corresponden a la información almacenada en la base de datos.

2. Crear un token que pueda utilizarse para que el usuario desde el inicio de sesión tenga un token que le permita navegar por las distintas páginas manteniendo la información personal. Este token se almacena en sesión Storage por un método de javascript y también se remueve cuando el usuario cierra sesión.

Para crear el token utilicé la librería Firebase mediante el comando
composer require firebase/php-jwt

api/register

Para almacenar los datos del usuario el método requerido es `POST`.

El input que viene en forma de objetoJSON se guarda y decodifica por medio de:

```
$json = json_decode(file_get_contents('php://input'), true);
```

Para que las actividades -que son un array- se guarden dentro de la base de datos se utiliza la función implode:

```
$json['activities'] = implode(",", $json['activities']);
```

Permitiendo que los objetos se guarden de esta manera:

La sentencia SQL utilizada es de tipo insert. No es necesario insertar el ID ya que es generado por la base de datos de forma autoincremental.

PÁGINA DE DETALLE DE RUTAS

Esta página tiene extensión php.

Para mostrar la ruta envió la información relacionada con la ruta desde script-pagina-principal.js

Cuando se dibujan la información de las rutas en las tarjetas, el título de cada tarjeta tiene un enlace:

```
<a href="pagina-detalle-rutas.php?idruta=${idruta}&tcx=${tcx}">${dato.route_name}</a>
```

En la página de detalle de rutas recojo el id de la ruta y envío el tcx que contiene el nombre del fichero con extensión tcx.

En $file guardo el fichero .tcx

```
$file = 'php/rutasxml/'.$tcx.'.tcx';
```

La función *simplexml_load_file* me permite poder recorrer este archivo php para poder mostrar los puntos de las rutas.

```
$xml = simplexml_load_file($file);
```

Accedo al xml para obtener 4 datos:

- El nombre de la ruta.
- La posición inicial (latitud)
- La posición inicial(longitud)
- Y para cada “Trackpoint” la latitud y la longitud que indica los diferentes puntos de recorrido de la ruta.

En la función initMap() le indico cuales son las propiedades lat y lng, para que cuando el usuario ingrese a la ruta le muestre con un zoom 14 el detalle de recorrido de la ruta.

```
function initMap() {
      let map = new google.maps.Map(document.getElementById("map"), {
        center: {
          lat: <?php echo $beginLat ?>,
          lng: <?php echo $beginLon ?>
        },
        zoom: 14,
        mapTypeId: 'terrain',
        //  mapTypeId:google.maps.MapTypeId.SATELLITE
      });
```

Los puntos están guardados en el array $coord. Y se muestran en el mapa gracias a la función *JSON.parse:*

```
 let points = JSON.parse('<?php echo json_encode($coord);?>');
```

MANERAS DE MEJORAR LA APLICACIÓN

- Para mejorar la seguridad se podría generar un id de manera única con la función *uniqueid().*
- Crear un formulario para que cada usuario pueda cargar la ruta que ha realizado.