



MEMORIA DEL PROYECTO

ANTONIO VÍLCHEZ GARCÍA

Índice

1. Contenido del proyecto	2
2. Funcionamiento del proyecto	2
3. Explicación del Web Service	9

1. Contenido del proyecto

El código fuente (codigo fuente.zip) contiene una carpeta llamada **clase_pw** que está dividida en cuatro carpetas.

La carpeta **images** contiene todas las imágenes de la red social como el logotipo y las imágenes de fondo.

La carpeta **src** contiene todos los archivos necesarios para que la red social funcione. Dentro de src hay tres carpetas más: La carpeta leaflet es para que se visualicen los mapas con las rutas. Las carpetas js y img corresponde al código AJAX para los desplegables de las provincias y localidades de España.

La carpeta **Users_images** contiene la imagen por defecto que se le asignará a cada usuario nuevo. También contendrá una carpeta por cada usuario que se registre en la red social. Las carpetas que se crearán se llamarán Usuarios seguido del id (Usuarios[ID]). A su vez dentro de esa carpeta si el usuario añade una publicación se crearan dos carpetas, una llamada rutas donde se guardará el .gpx y otra llamada imagenes seguido del número de publicación de ese usuario.

La carpeta **ws** contiene los archivos del web service implementado con SOAP. Los archivos son: cliente_soap.php y servidor_soap.php.

2. Funcionamiento del proyecto

La base de datos que guardará toda la información se llama **clase_pw** y contiene las tablas usuarios, amigos, actividad, deportes, imagenes, rutas. El usuario de acceso a la base de datos se denomina ' practica ' con contraseña ' practica '.

Las tablas contienen esta información:

```
-- phpMyAdmin SQL Dump
-- version 5.2.0
-- https://www.phpmyadmin.net/
--
-- Servidor: 127.0.0.1
-- Tiempo de generación: 12-07-2023 a las 16:53:16
-- Versión del servidor: 10.4.27-MariaDB
-- Versión de PHP: 8.2.0

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";
--
-- Base de datos: 'clase_pw'
--
--
-- Estructura de tabla para la tabla 'actividad'
--
```

```

CREATE TABLE 'actividad' (
    'id' int(11) NOT NULL,
    'id_usuario' int(11) NOT NULL,
    'titulo' varchar(100) DEFAULT NULL,
    'tipo_actividad' varchar(50) NOT NULL,
    'ruta_gpx' varchar(255) DEFAULT NULL,
    'ruta_imagenes' varchar(400) NOT NULL,
    'companeros_actividad' text NOT NULL,
    'aplausos' int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
-- -----

--
-- Estructura de tabla para la tabla 'amigos'
--

CREATE TABLE 'amigos' (
    'id' int(11) NOT NULL,
    'id_usuario' int(11) NOT NULL,
    'id_amigo' int(11) NOT NULL,
    'estado' enum('pendiente','rechazada','aceptada') NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
-- -----

--
-- Estructura de tabla para la tabla 'deportes'
--

CREATE TABLE 'deportes' (
    'id' int(11) NOT NULL,
    'nombre' varchar(300) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
-- -----

--
-- Estructura de tabla para la tabla 'imagenes'
--

CREATE TABLE 'imagenes' (
    'id' int(11) NOT NULL,
    'id_usuario' int(11) NOT NULL,
    'nombre' varchar(300) NOT NULL,
    'tamaño' double NOT NULL,
    'alto' double NOT NULL,
    'ancho' double NOT NULL,
    'ruta' varchar(300) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
-- -----

--
-- Estructura de tabla para la tabla 'rutas'
--

```

```

CREATE TABLE 'rutas' (
  'id' int(11) NOT NULL,
  'ruta' varchar(300) NOT NULL,
  'id_usuario' int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
-- -----
--
-- Estructura de tabla para la tabla 'usuarios'
--

CREATE TABLE 'usuarios' (
  'id' int(11) NOT NULL,
  'nombre' varchar(50) NOT NULL,
  'apellidos' varchar(100) NOT NULL,
  'usuario' varchar(100) NOT NULL,
  'email' varchar(100) NOT NULL,
  'contraseña' varchar(50) NOT NULL,
  'fecha_nacimiento' date NOT NULL,
  'actividad_preferida' varchar(50) NOT NULL,
  'provincia' varchar(50) NOT NULL,
  'localidad' varchar(50) NOT NULL,
  'pais' varchar(50) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Índices para tablas volcadas
--

--
-- Indices de la tabla 'actividad'
--
ALTER TABLE 'actividad'
  ADD PRIMARY KEY ('id'),
  ADD KEY 'fk_publicaciones_usuario' ('id_usuario');

--
-- Indices de la tabla 'amigos'
--
ALTER TABLE 'amigos'
  ADD PRIMARY KEY ('id'),
  ADD KEY 'fk_amigos_usuario' ('id_usuario'),
  ADD KEY 'fk_amigos_usuario1' ('id_amigo');

--
-- Indices de la tabla 'deportes'
--
ALTER TABLE 'deportes'
  ADD PRIMARY KEY ('id');
--

```

```

-- Indices de la tabla 'imagenes'
--
ALTER TABLE 'imagenes'
  ADD PRIMARY KEY ('id'),
  ADD KEY 'fk_imagenes_usuarios' ('id_usuario');

--
-- Indices de la tabla 'rutas'
--
ALTER TABLE 'rutas'
  ADD PRIMARY KEY ('id'),
  ADD KEY 'fk_rutas_usuarios' ('id_usuario');

--
-- Indices de la tabla 'usuarios'
--
ALTER TABLE 'usuarios'
  ADD PRIMARY KEY ('id') USING BTREE;

--
-- AUTO_INCREMENT de las tablas volcadas
--

--
-- AUTO_INCREMENT de la tabla 'actividad'
--
ALTER TABLE 'actividad'
  MODIFY 'id' int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT de la tabla 'amigos'
--
ALTER TABLE 'amigos'
  MODIFY 'id' int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT de la tabla 'deportes'
--
ALTER TABLE 'deportes'
  MODIFY 'id' int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT de la tabla 'imagenes'
--
ALTER TABLE 'imagenes'
  MODIFY 'id' int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT de la tabla 'rutas'
--
ALTER TABLE 'rutas'
  MODIFY 'id' int(11) NOT NULL AUTO_INCREMENT;

```

```

--
-- AUTO_INCREMENT de la tabla 'usuarios'
--
ALTER TABLE 'usuarios'
    MODIFY 'id' int(11) NOT NULL AUTO_INCREMENT;

--
-- Restricciones para tablas volcadas
--

--
-- Filtros para la tabla 'actividad'
--
ALTER TABLE 'actividad'
    ADD CONSTRAINT 'fk_publicaciones_usuario' FOREIGN KEY ('id_usuario') REFERENCES 'usuarios' ('id') ON DELETE CASCADE;

--
-- Filtros para la tabla 'amigos'
--
ALTER TABLE 'amigos'
    ADD CONSTRAINT 'fk_amigos_usuario' FOREIGN KEY ('id_usuario') REFERENCES 'usuarios' ('id') ON DELETE CASCADE,
    ADD CONSTRAINT 'fk_amigos_usuario1' FOREIGN KEY ('id_amigo') REFERENCES 'usuarios' ('id') ON DELETE CASCADE;

--
-- Filtros para la tabla 'imagenes'
--
ALTER TABLE 'imagenes'
    ADD CONSTRAINT 'fk_imagenes_usuarios' FOREIGN KEY ('id_usuario') REFERENCES 'usuarios' ('id') ON DELETE CASCADE;

--
-- Filtros para la tabla 'rutas'
--
ALTER TABLE 'rutas'
    ADD CONSTRAINT 'fk_rutas_usuarios' FOREIGN KEY ('id_usuario') REFERENCES 'usuarios' ('id') ON DELETE CASCADE;
COMMIT;

```

El usuario cuándo acceda a mi red social le saldrá el **index.php** donde podrá elegir si quiere iniciar sesión (login.php) o registrarse (register.php).

Supongamos que elige registrarse, entonces pasará a **register.php** donde se le pedirá de forma obligatoria un Nombre y Apellido, un Usuario (entre 8 y 15 caracteres y números), un Correo Electrónico, una Contraseña (entre 6 y 10 caracteres), una Fecha de nacimiento, una Actividad preferida, una Localidad, Provincia y País. Una vez se registre se creará en la carpeta Users.images una carpeta llamada Usuarios seguido del id de ese usuario. A continuación le muestro una foto de como se crea automáticamente una carpeta para cada usuario registrado:

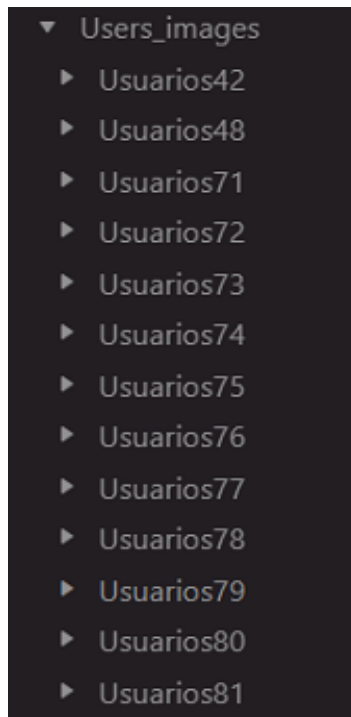


Figura 1: Users_images.

Guardo toda la información del usuario en la tabla **usuarios**.

Una vez realizado el registro pasará al inicio de sesión, **login.php**. En el login se le pedirá que introduzca el email y contraseña y se comprobará.

Una vez inicie sesión le aparecerá la página principal, **web.php**. En **web.php** le aparecerá una barra de navegación horizontal en la parte superior, donde podrá editar su foto de perfil (**cambiar_foto.php**), ver su perfil (**perfil.php**), añadir publicación (**anadir_publicacion.php**), buscar usuarios (**../ws/cliente_soap.php**) y cerrar sesión (**cerrar_sesion.php**). Debajo de la barra de navegación aparece las publicaciones que han hecho los amigos del usuario con toda la información (título, actividad, compañero, etc...) y un botón para dar aplausos (**dar_aplauso.php**).

En **cambiar_foto.php** el usuario podrá cambiar su foto de perfil las veces que quiera.

En **perfil.php** el usuario podrá ver toda su información (nombre, apellido, usuario, localidad, provincia, país, actividad, email y amigos). Tendrá un botón para editar su perfil (**editar_perfil.php**) en dónde podrá cambiar la información que el usuario quiera. Hay otro botón para ver sus publicaciones (**publicaciones.php**) donde podrá ver las publicaciones que el usuario ha publicado y si desea puede eliminarlas (**eliminar_publicacion.php**). El usuario podrá ver y eliminar los amigos que desee (**eliminar_amigo.php**).

En **anadir_publicación.php** el usuario podrá publicar una ruta, añadiéndole el título, tipo de actividad, archivo .gpx, el compañero y las imágenes. Si el usuario realiza una publicación, dentro de su carpeta local (creada anteriormente cuando se registra) se crearan dos carpetas, una llamada rutas donde se guardará el .gpx y otra llamada imagenes seguido del número de publicación de ese usuario donde se guardaran las imágenes. Guardo la información de la publicación en la tabla **actividad**.

Podemos verlo en este ejemplo:

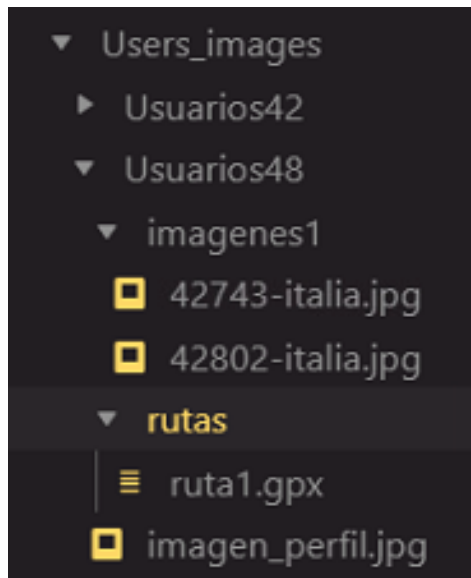


Figura 2: Carpeta rutas e imagenes.

Como se puede ver el usuario con ID 48 ha publicado una actividad y se ha creado una carpeta 'rutas' y una carpeta 'imagenes' seguido del numero de publicación, como es la primera se le añade el 1. La ruta se guarda igual que la carpeta 'imagenes' con el nombre ruta seguido del número de publicación.

La información de la ruta la guardo en la tabla **rutas**.

La información de cada imagen la guardo en la tabla **imagenes**.

En **cliente_soap.php** se podrá buscar usuarios y saldrá su nombre, apellidos y actividad preferida. El archivo cliente_soap.php pasa el nombre y apellidos a servidor_soap.php y este le envía la información con el protocolo SOAP que viene integrado en PHP. Una vez salga el resultado aparecerá un botón que dirá Realizar más acciones", si se pulsa (**buscar.php**) se podrá ver el perfil del usuario buscado si son amigos. Si no son amigos puedes enviarle una solicitud de amistad (**enviar_solicitud.php**). En la base de datos tengo hecho una tabla amigos, si un usuario manda una solicitud a otro en la base de datos aparece el id del usuario que ha mandado la solicitud, el id del usuario al que han mandado la solicitud y un estado, que en este caso sería 'pendiente'. Si el al que han mandado la solicitud acepta, el estado pasaría 'aceptado' y serían amigos. Si rechaza, pasaría a 'rechazado' y no serían amigos.

Un ejemplo de la tabla amigos es:


































← T →				id	id_usuario	id_amigo	estado
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	23	42	48	aceptada
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	28	72	48	aceptada
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	29	73	48	aceptada
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	30	74	48	aceptada
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	31	48	75	aceptada
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	32	76	48	aceptada
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	33	77	48	aceptada
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	34	79	48	aceptada
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	35	80	48	aceptada
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	36	81	48	aceptada
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	37	48	78	aceptada

Figura 3: Tabla amigos.

Debajo de la barra de navegación horizontal en **web.php** aparecerá las publicaciones (actividades) que los amigos del usuario han publicado, con la ruta, la información (título, actividad, compañero) y las imágenes. El usuario podrá aplaudir (**dar_aplausos.php**) en las publicaciones que le guste de sus amigos.

Las publicaciones (actividades) aparecerán paginadas de 10 en 10 y siempre se mostrará la última publicación de cada amigo.

El administrador tendrá una cuenta separada de las normales de usuarios. El correo para acceder es **admin@admin.com** y la contraseña es **admin123**.

El administrador tendrá una interfaz distinta al resto, es el archivo **administrador.php**. Dentro de este archivo el administrador podrá buscar un usuario por nombre y apellido. Una vez que los busque le saldrá el nombre, apellidos, actividad preferida e ID del usuario buscado y podrá ver sus publicaciones y eliminarlas si desea el administrador (**publicaciones_administrador.php**), ver el perfil (**perfil_administrador.php**), editar el perfil (**editar_perfil.php**) y dar de baja al usuario (**eliminar_usuario.php**).

También el administrador podrá añadir una actividad o eliminar un tipo de actividad para que el usuario elija una actividad preferida.

3. Explicación del Web Service

El Web Service consta de dos archivos, el servidor (**servidor_soap.php**) y el cliente (**cliente_soap.php**) que se encuentran en la carpeta **ws**. **IMPORTANTE:** Tener habilitada (descomentada) en **php.ini** la extension SOAP, **extension=soap**.

El servidor hace:

Se definen las variables namespace y serviceName que se utilizarán para configurar el servidor SOAP.

Se crea una instancia del servidor SOAP utilizando new SoapServer. El primer parámetro es null, lo que indica que se utilizará el archivo WSDL predeterminado generado automáticamente. El segundo parámetro es un array que contiene la URI del namespace.

Se registra la función getUserInfo como un método del servicio web utilizando addFunction. Esto permite que el método sea accesible a través del servicio SOAP.

Se implementa la función getUserInfo, que realiza una consulta SQL a la base de datos para obtener la información de los usuarios en función del nombre y apellidos proporcionados como parámetros. La función construye la consulta SQL en base a los parámetros recibidos y ejecuta la consulta utilizando la conexión a la base de datos.

Si se encuentran resultados, los registros de usuarios se almacenan en un array asociativo y se devuelven como respuesta al cliente SOAP.

El servidor SOAP procesa la solicitud utilizando handle(), lo que significa que está listo para recibir y responder a las solicitudes SOAP entrantes.

El cliente hace:

La página HTML muestra un formulario de búsqueda donde se pueden ingresar el nombre y apellidos de los usuarios a buscar.

Se establece la URL del servidor SOAP en la variable soapURL. Esta URL debe apuntar al archivo del servidor SOAP.

Se definen las variables namespace y serviceName que coinciden con las utilizadas en el servidor SOAP.

Se crea una instancia del cliente SOAP utilizando new SoapClient. El primer parámetro es null, lo que indica que se utilizará el archivo WSDL predeterminado generado automáticamente. El segundo parámetro es un array que contiene la ubicación (location) y el URI del namespace.

Se llama al método del servicio web utilizando client → __soapCall(). El primer parámetro es el nombre del método del servicio web a invocar, en este caso, 'getUserInfo'. El segundo parámetro es un array que contiene los argumentos del método del servicio web.

Se procesa la respuesta del servicio web. Si se encuentran resultados, se recorren los usuarios devueltos y se muestra la información en la página.