

GranaTourz

Proyecto Final DAW

Alejandro Martín González




Resumen

El sector turístico ha aumentado en la actualidad, y cada vez más personas buscan experiencias únicas y personalizadas durante sus viajes. Es por esto que he desarrollado “GranaTourz”, que ofrece a los usuarios la posibilidad de reservar actividades proporcionadas por diversos organizadores.

El objetivo de este proyecto es diseñar, desarrollar e implementar la aplicación web “GranaTourz”, para dar una solución innovadora y práctica para conectar a los usuarios con un amplio catálogo de actividades.

La interfaz es intuitiva y fácil de usar, donde pueden explorar diversas actividades disponibles en una ubicación específica. Los organizadores pueden registrar sus actividades en la plataforma, proporcionando detalles como descripciones, horarios y más detalles sobre estas. Los usuarios pueden buscar y filtrar actividades por categorías y realizar reservas.

Además, se han incorporado funcionalidades como la posibilidad de dejar comentarios y valoraciones de las actividades, lo que permite a los usuarios tomar decisiones a la hora de realizar su reserva. “GranaTourz” busca mejorar la forma en la que las personas planifican y disfrutan de sus actividades.



The tourism sector has experienced significant growth in recent years, with more and more people seeking unique and personalized experiences during their travels. That's why I have developed “GranaTourz,” which offers users the opportunity to book activities provided by various organizers.

The objective of this project is to design, develop, and implement the web application “GranaTourz” to provide an innovative and practical solution for connecting users with a wide range of activities.

The interface is intuitive and user-friendly, allowing users to explore diverse activities available in a specific location. Organizers can register their activities on the platform, providing details such as descriptions, schedules, and additional information. Users can search and filter activities by categories and make reservations.

Moreover, additional features have been incorporated, such as the ability to leave comments and reviews about the activities, enabling users to make informed decisions when making their reservations. “GranaTourz” aims to improve the way people plan and enjoy their activities.

The “GranaTourz” web application aims to enhance the overall travel experience by providing users with a convenient platform to discover and book activities tailored to their preferences. By leveraging user-friendly design and seamless functionality, “GranaTourz” seeks to revolutionize the way travelers explore and engage with local experiences.



Índice


Resumen	2
Índice	4
Justificación	5
Justificación de la tecnología empleada	8
Requerimientos hardware y software	9
Análisis y diseño	11
Implementación	14
Relación detallada de cada uno de los ficheros que se incluyen:	15
Evaluación y prueba	21
Manual de estilos	24
Enlaces de interés	29
Bibliografía	30

Justificación

A. Características generales

El proyecto tiene las siguientes características generales que respaldan la justificación:

- **Innovación:** “GranaTourz” ofrece una solución innovadora para conectar a los usuarios con un amplio catálogo de actividades proporcionadas por diversos organizadores. A través de su interfaz intuitiva y funcionalidades avanzadas, se brinda una experiencia de reserva única y personalizada.
- **Necesidad:** En la actualidad, hay una creciente demanda de opciones para reservar actividades. “GranaTourz” aborda esta necesidad al proporcionar una plataforma centralizada donde los usuarios pueden explorar y reservar una variedad de actividades, facilitando la planificación de sus experiencias y maximizando su tiempo de disfrute.
- **Impacto:** La aplicación web mejorará la experiencia del usuario al simplificar el proceso de reserva de actividades. Los usuarios podrán acceder a un amplio catálogo de actividades, filtrarlas por categorías, ver los detalles de la actividad y realizar las reservas de manera conveniente. Esto permitirá una gestión más eficiente y una mayor satisfacción en sus viajes.
- **Ventaja competitiva:** “GranaTourz” proporcionará una ventaja competitiva tanto a los organizadores de actividades como a los usuarios al ofrecer una plataforma confiable, segura y fácil de usar. Los organizadores podrán promocionar sus actividades y alcanzar una audiencia más amplia, mientras que los usuarios disfrutarán de una amplia selección de opciones y una experiencia de reserva sin complicaciones.
- **Beneficios esperados:** Se esperan beneficios como una mayor eficiencia en la reserva de actividades, una mayor disponibilidad de opciones para los usuarios, una mayor visibilidad y alcance para los organizadores, una mayor satisfacción a la hora de la planificación y disfrute de las actividades.



En general, la aplicación web proporcionará una solución innovadora y práctica para la reserva de actividades.

B. Restricciones generales


En el caso de la aplicación web, existen algunas restricciones generales:

- Restricciones de capacidad: Las actividades cuentan con una capacidad limitada en términos de la cantidad de personas que pueden participar. Esto hace que se tenga que requerir a una gestión de cupos o límites de reserva para garantizar una experiencia óptima para los usuarios, en este caso solo se permite reservar una vez una actividad determinada.
- Restricciones geográficas: Ya que esta aplicación está diseñada para que los organizadores puedan publicar generalmente actividades en la provincia de Granada.

C. Aspectos a cubrir junto con los que no se van a tratar

Aspectos a cubrir:

- Registro de usuarios: La aplicación permitirá a los usuarios crear una cuenta y registrarse para acceder a funcionalidades como realizar reservas, ver sus reservas o dejar comentarios.
- Búsqueda y filtrado de actividades: Los usuarios podrán explorar y buscar actividades por categorías, esto proporcionará al usuario que puedan encontrar las actividades deseadas de manera eficiente.
- Detalles de la actividad: La aplicación mostrará información detallada sobre cada actividad, como la descripción, horarios, fechas, ubicación y disponibilidad. Esto ayudará a los usuarios a tomar decisiones informadas al realizar sus reservas.
- Gestión de reservas: Los usuarios podrán acceder a un panel de control donde podrán ver y administrar sus reservas existentes, incluyendo la posibilidad de cancelar las reservas.
- Comentarios y valoraciones: Se proporcionará la opción para que los usuarios puedan dejar comentarios y una valoración sobre las actividades, lo que ayudará a otros usuarios a tomar decisiones informadas.



Aspectos que no se van a tratar:

- Gestión financiera: No existirán métodos de pago, porque esta aplicación está basada en la nueva tendencia de los free tours es decir que los usuarios pueden asistir a las actividades gratis y dejar un donativo en el caso de que estos lo deseen.

D. Estudio de las prestaciones de la herramienta que se propone frente a otras existentes de la misma categoría

Tras un estudio frente a otras aplicaciones existentes de la misma categoría revela una serie de ventajas y diferencias significativas. Algunas de las principales prestaciones que la diferencia son:

- Interfaz intuitiva y fácil de usar: Ha sido diseñada con una interfaz intuitiva y amigable para el usuario, lo que facilita la navegación y la realización de reservas de actividades. En comparación con otras aplicaciones existentes, esta aplicación destaca por su simplicidad y accesibilidad.
- Amplio catálogo de actividades: Se ofrece un amplio catálogo de actividades proporcionadas por diversos organizadores.
- Comentarios y valoraciones: Se ha implementado un sistema de comentarios y valoraciones, esto proporciona a los usuarios una información confiable a la hora de tomar decisiones informadas al reservar actividades.

Justificación de la tecnología empleada

En el proyecto he utilizado diferentes tecnologías:

- JavaScript: Es un lenguaje de programación ampliamente utilizado en el desarrollo web para agregar interactividad y dinamismo a las páginas. Lo he utilizado para las validaciones de formularios en el lado del cliente.
- PHP: Es un lenguaje de programación del lado del servidor utilizado en el desarrollo web. Lo he utilizado para backend de la aplicación para manejar las solicitudes del cliente, procesar formularios, además de establecer la conexión con la base de datos. Esto me ha permitido realizar consultas, obtener resultados y manipular los datos almacenados en la base de datos.
- MySQL: Es un sistema de gestión de bases de datos relacionales que utiliza SQL como lenguaje para interactuar con la base de datos. Lo he utilizado para la creación de la base de datos, crear la sintaxis para manipular los datos de la aplicación como inserción, actualización y eliminación de registros en las tablas.
- Bootstrap: Es un framework frontend que facilita el desarrollo de interfaces de usuario adaptadas a todos los dispositivos y atractivas. Lo he utilizado en la aplicación para la creación del diseño en general como son las vistas, formularios, menús.
- JQuery: Es un framework que simplifica la manipulación del DOM (Document Object Model), la interacción con elementos HTML, el manejo de eventos y el envío de solicitudes AJAX. Lo he usado para darle funcionalidad al menú desplegable.
- PHPMailer: Es un framework de PHP que facilita el envío de correos electrónicos desde aplicaciones web. Lo he utilizado para enviar los distintos correos como el de verificación de cuenta al registrarse como usuario, y en la página de contacto para darle la funcionalidad para permitir enviar correos.



Requerimientos hardware y software

Los requerimientos que vamos a tener en parte del hardware son los siguientes:

- Servidor: Se necesita un servidor donde alojar la aplicación web. Puede ser en un servidor físico o un servicio de alojamiento en la nube. Es importante tener en cuenta el rendimiento del procesador, la memoria RAM y el espacio de almacenamiento.
- Red: En caso de tener un servidor físico es otro punto a tener en cuenta. Es necesario tener una conexión de red estable y de alta velocidad para garantizar un rendimiento óptimo de la aplicación web tanto para los usuarios como para el servidor.
- Escalabilidad: Esto es un factor importante porque si se espera que entre mucho tráfico a la aplicación siempre hay que tener en cuenta de que se pueda escalar el hardware para controlar esa carga.

Los requerimientos por parte de software son los siguientes:

- Sistema operativo del servidor: Se puede utilizar un sistema operativo compatible con PHP Y MySQL, como Linux o Windows Server. Hay que tener en cuenta que cumpla los requisitos de las versiones de PHP y MySQL.
- Servidor web: Es necesario un servidor web compatible con PHP, como Apache o Nginx. Es importante configurar el servidor web para que pueda procesar archivos PHP.
- PHP: Instalar una versión compatible de PHP en el servidor.
- MySQL: Es necesario instalar y configurar un servidor de base de datos en el servidor. Además de comprobar que cumpla los requisitos de la versión de MySQL que se está

utilizando, también es necesario configurar correctamente las credenciales de acceso y los permisos de usuario.

- Navegador web: Los usuarios necesitarán un navegador web moderno y compatible, como Google Chrome, Mozilla Firefox o Microsoft Edge, para acceder y utilizar la aplicación web correctamente. Es recomendable siempre tener actualizados los navegadores.

Para poner este proyecto en producción he utilizado un plan de hostinger con los siguientes detalles en el hosting web:

Detalles del hosting		Mejorar plan
Espacio del disco	100 GB	
RAM	1024 MB	
Núcleos de CPU	1	
Inodos	400000	
Complementos/sitios web	100	
Procesos activos	40	
Procesos de entrada	20	
Ancho de banda	Ilimitado	

Análisis y diseño

Diagrama de casos de uso:

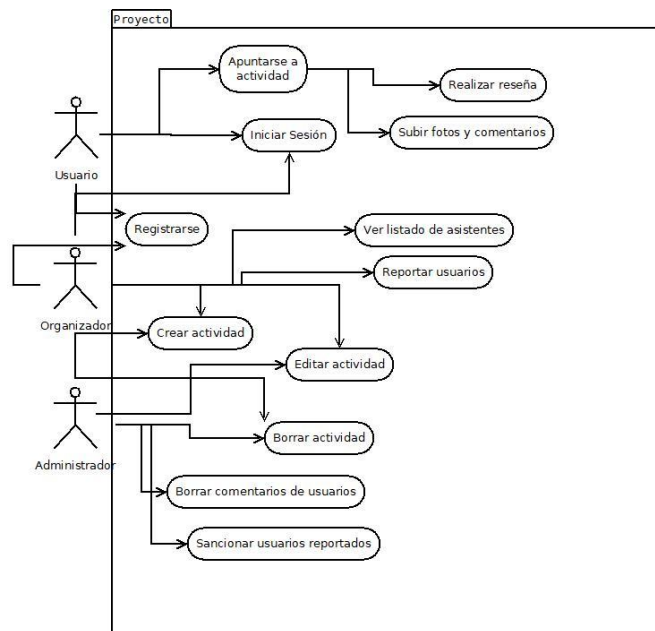
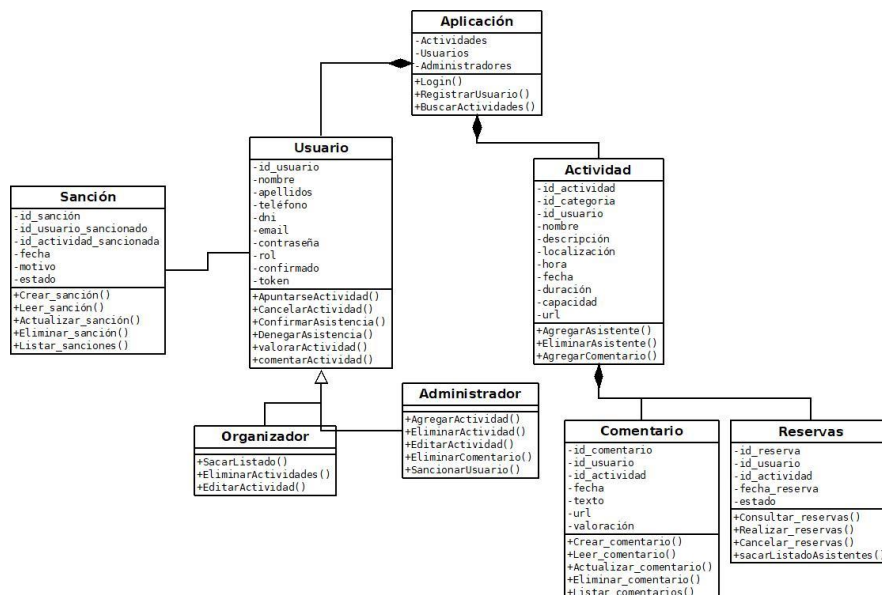


Diagrama de clases:



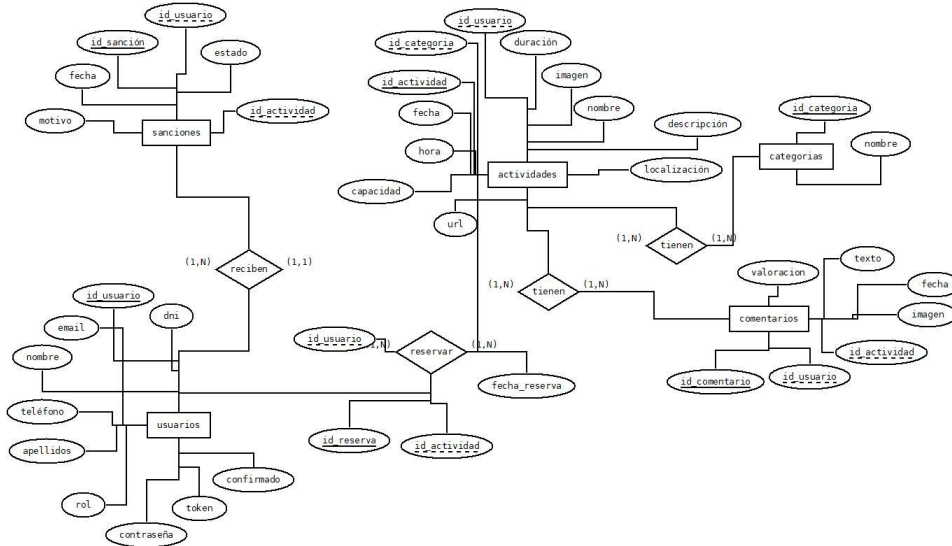


Descripción de la BD:

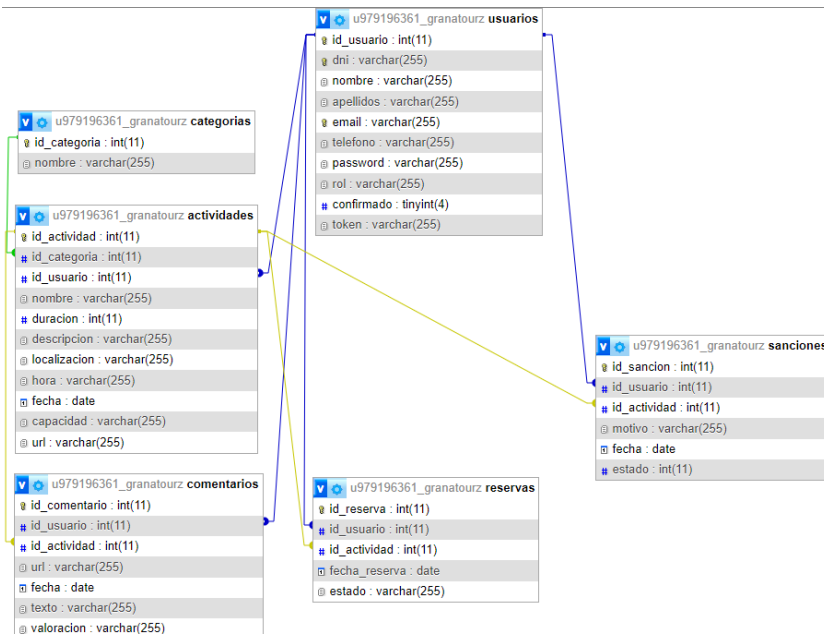
La base de datos está compuesta por seis tablas:

- Actividades: Tiene una estructura con los siguientes campos: id_actividad (Clave primaria), id_categoria (Clave Foránea), id_usuario(Clave Foránea), nombre, duración, descripción, localización, hora, fecha, capacidad, url.
- Categorías: Tiene una estructura con los siguientes campos: id_categoria (Clave primaria), nombre.
- Comentarios: Tiene una estructura con los siguientes campos: id_comentario (Clave primaria), id_usuario(Clave foránea),id_actividad(Clave foránea), url, fecha, texto, valoración.
- Reservas: Tiene una estructura con los siguientes campos: id_reserva (Clave primaria), id_usuario(Clave foránea), id_actividad(Clave foránea),fecha_reserva.
- Sanciones: Tiene una estructura con los siguientes campos: id_sanción (Clave primaria), id_usuario(Clave foránea), id_actividad(Clave foránea),motivo,fecha,estado.
- Usuarios: Tiene una estructura con los siguientes campos: id_usuario(Clave primaria), dni, nombre, apellidos, email, teléfono, password, rol , confirmado,token.

Modelo Entidad/Relación:



Diseño lógico de tablas y relaciones:



Implementación

En este proyecto de “GranaTourz”, se utilizan diversos elementos y tecnologías para su implementación. A continuación voy a detallar los principales:

- Archivos PHP: Se emplean archivos php para el desarrollo del backend de la aplicación, estos archivos contienen código php que se ejecuta en el servidor y se encarga de procesar las solicitudes del cliente, interactuar con la base de datos y generar respuestas dinámicas.
- JavaScript: Se utiliza para la programación del lado del cliente, en este caso sobre todo se ha usado a la hora de la validación de formularios.
- Librería Bootstrap: Se hace uso de la librería Bootstrap, la cual proporciona un conjunto de estilos predefinidos y componentes HTML y CSS. Esto facilita el diseño y la maquetación, asegurando una apariencia coherente y adaptable a los diferentes dispositivos. Se ha usado tanto para mostrar los datos, como para el tema del menú.
- Librería PHPMailer: Para el envío de correos electrónicos. Esta permite enviar correos desde el servidor de forma sencilla y enviar mensajes de correo electrónico con facilidad. Se ha usado tanto para el envío de correo al registrarse como para validar la cuenta y para el formulario de contacto.
- Conexión a la base de datos MySQL: Para almacenar y recuperar datos relacionados con las reservas de actividades, se utiliza una conexión a la base de datos MySQL. Se establece una conexión mediante código PHP y se realizan consultas SQL para interactuar con la base de datos, como para almacenar los nuevos registros, actualizar información existente o recuperar datos específicos.
- Archivo .htaccess: Se emplea el archivo .htaccess para configurar reglas de redireccionamiento y reescritura de URL. Esto permite tener las URL's amigables para los usuarios y facilitar la gestión de las rutas de la aplicación.

- Archivo .env: Se utiliza el archivo .env para almacenar variables de entorno y configuración sensible de la aplicación. Estas variables incluyen credenciales de la base de datos, y otros datos confidenciales. El archivo .env se carga en el código PHP mediante un proceso de carga de entorno.

Estos elementos y tecnologías mencionados son algunos de los que se han empleado a la hora de desarrollar el proyecto de “GranaTourz”. Cada uno de ellos cumple un papel importante en la implementación y funcionalidad de la aplicación.

Relación detallada de cada uno de los ficheros que se incluyen:

Para una mejor organización y estructura del proyecto “GranaTourz”, se ha dividido en varios directorios y se han utilizados los siguientes:

- Directorio “Controllers”: En este directorio se encuentran los controladores, que son los responsables de manejar las solicitudes del cliente y coordinar las acciones correspondientes. Los controladores interactúan con los modelos, servicios y repositorios para procesar los datos y generar respuestas adecuadas. En la aplicación tengo los siguientes controladores:

→ ActividadController.php

(Se encuentra la clase, y sus métodos. Ej: crear_actividad() esta función recoge los datos de los formularios y llama al servicio como intermediario con el repositorio que es el que los mete en la base de datos)

→ CategoriaController.php

(Se encuentra la clase, y sus métodos. Ej: crear_categoria() esta función recoge los datos del formulario por método POST, llama al servicio como intermediario con el repositorio que es el que los mete en la base de datos)

- ComentarioController.php

(Se encuentra la clase, y sus métodos. Ej: crear_comentario() esta función recoge los datos del formulario por método POST, llama al servicio como intermediario con el repositorio que es el que los mete en la base de datos)
- ContactoController.php

(Se encuentra la clase, y sus métodos. Ej: enviar_mensaje() esta función recoge los datos del formulario por método POST, este utiliza la clase Email para usar su método para enviar el correo pasándole los datos del formulario)
- ReservaController.php

(Se encuentra la clase, y sus métodos. Ej: consultar_reservas() utiliza la sesión del id del usuario para realizar una consulta a la base de datos para realizar un select y ver qué reservas ha realizado en caso de que no tenga se muestra un mensaje con el pages)
- SancionController.php

(Se encuentra la clase, y sus métodos. Ej: proponer_sancion() recoge los datos del formulario por el método POST, para rellenar en la base de datos. Utiliza INSERT)
- UsuarioController.php

(Se encuentra la clase, y sus métodos. Ej: save() recoge los datos del usuario para registrarlo en la base de datos. Utiliza INSERT)
- Directorio “Lib”: En este directorio se almacenan archivos de utilidad personalizados que se utilizan en la aplicación web. Estos archivos contienen funciones y clases que se pueden reutilizar en diferentes partes del proyecto para tareas específicas. Se encuentran los siguientes archivos:
 - BaseDatos.php

(Se encuentra la clase, y sus métodos para conectarse a la Base de datos)
 - Email.php

(Se encuentra la clase, y sus métodos para realizar los envíos de correos)
 - Pages.php

→ Router.php

- Directorio “Models”: En este directorio se encuentran los modelos, que son las clases que se definen para encapsular los datos (propiedades) y comportamientos (métodos). En este directorio se encuentran los siguientes modelos:

- actividad.php (La clase con sus propiedades, y métodos)
- categoria.php (La clase con sus propiedades, y métodos)
- comentario.php (La clase con sus propiedades, y métodos)
- reservas.php (La clase con sus propiedades, y métodos)
- sancion.php (La clase con sus propiedades, y métodos)
- usuarios.php (La clase con sus propiedades, y métodos)

- Directorio “public”: Este directorio es accesible públicamente y contiene los archivos estáticos de la aplicación, como los scripts JavaScript e imágenes. También incluye el archivo index.php, además del archivo .htaccess para gestionar las rutas y redireccionamientos. En la carpeta public se encuentra lo siguiente:

- Directorio images donde se cargan las imágenes que se introducen.
- Directorio css
- Directorio js-script que dentro incluye otros dos directorios que contienen las validaciones de los formularios y otro para el botón de scroll.
- Archivo .htaccess
- Archivo index.php (Ej: Router::add por POST confirmar_sanción)

- Directorio “Repositories”: Aquí se encuentran los repositorios, que se encargan de abstraer y manejar la interacción con la base de datos. Los repositorios contienen los métodos para realizar las consultas y operaciones relacionadas con la persistencia de datos. En este directorio se encuentran los diferentes archivos:

→ ActividadRepository.php

(Ej: crear_actividad() realiza un insert con los datos que se pasan por el formulario)



→ CategoriaRepository.php

(Ej: crear_categoria() realiza un insert con los datos que se pasan por el formulario)

→ ComentarioRepository.php

(Ej: crear_comentario() realiza un insert con los datos que se pasan por el formulario)

→ ReservaRepository.php

(Ej: realizar_reserva() realiza un insert con los datos que se pasan por el formulario)

→ SancionRepository.php

(Ej: confirmar_sanción() realiza un update para confirmar la sanción al usuario)

→ UsuarioRepository.php

(Ej: save() realiza un insert para registrar al usuario en la base de datos)

- Directorio “Services”: En este directorio se encuentran los servicios, los cuales actúan como intermediarios entre los controladores y los repositorios. Estos llaman a los métodos del repositorio correspondiente para acceder a los datos almacenados en la base de datos y realizar las operaciones necesarias. En este directorio se encuentran los siguientes archivos:

→ ActividadService.php

→ CategoriaService.php

→ ComentarioService.php

→ ReservaService.php

→ SancionService.php

→ UsuarioService.php

- Directorio “vendor”: En este directorio se almacenan las dependencias externas de la aplicación, gestionadas a través de un sistema de gestión de paquetes como Composer. Aquí se instalan y mantienen las bibliotecas utilizadas en el proyecto.

- phpdotenv (permite cargar las variables de entorno desde el archivo .env)
- autoload.php (permite cargar automáticamente las clases y archivos)
- phpmailer (permite enviar correos electrónicos)
- composer


- Directorio “Views”: En este directorio se encuentran las vistas, que contienen la estructura y presentación visual de la aplicación. Además en vistas como por ejemplo: crear_actividad.php se encuentran formularios que se utilizan para recoger los datos por el método post. Tienen la siguiente estructura:

- Directorio “actividades”
 - ◆ crear_actividad.php
 - ◆ editar_actividad.php
 - ◆ editar_actividad2.php
 - ◆ editar_actividad3.php
 - ◆ listar_actividades.php
 - ◆ listarXcategorias.php
 - ◆ ver_actividad.php

- Directorio “ayuda”
 - ◆ contacto.php
 - ◆ normas.php

- Directorio “categorias”
 - ◆ crear_categoria.php

- Directorio “layout”
 - ◆ 404.php
 - ◆ footer.php
 - ◆ header.php
 - ◆ mensaje.php



→ Directorio “organizadores”

- ◆ ver_inscritos.php
- ◆ ver_listado.php

→ Directorio “reservas”

- ◆ crear_reserva.php
- ◆ mis_reservas.php

→ Directorio “sanciones”

- ◆ ver_propuestas.php

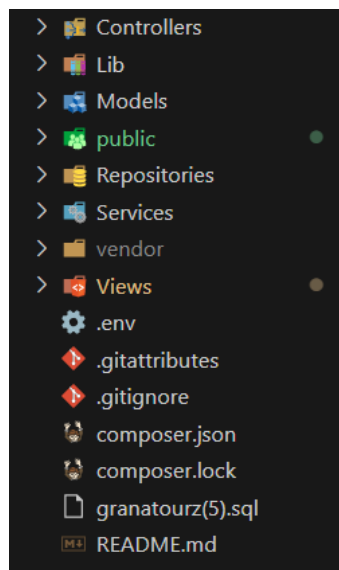
→ Directorio “usuarios”

- ◆ editar_datos.php
- ◆ login.php
- ◆ registro.php

Además de estos directorios que se encuentran en el directorio raíz de la aplicación están los siguientes archivos:

- Archivo .env: En este archivo se almacenan las variables de entorno de la aplicación, se cargan utilizando la biblioteca “phpdotenv”.
- Archivo composer.json: Es el archivo de configuración utilizado por Composer, define las dependencias del proyecto es decir los paquetes de terceros necesarios para el funcionamiento de la aplicación.
- Archivo composer.lock: Es un archivo generado por Composer. Se crea automáticamente.

Imagen con la estructura del proyecto:



Evaluación y prueba

La evaluación y prueba de la aplicación web y de su base de datos es una parte fundamental para garantizar la funcionalidad, calidad y la seguridad de esta. A continuación voy a comentar las diferentes pruebas que se han realizado:

- Pruebas de integración:

→ Prueba de conexión: Se ha comprobado que el código PHP pueda establecer la conexión de manera exitosa con la base de datos. Ejemplo:

Esta función en el caso de no poder conectar con la Base de Datos mostraría en la aplicación el error.

```

private function conectar(): PDO {
    try{
        $opciones = array(
            PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
            PDO::MYSQL_ATTR_FOUND_ROWS => true
        );

        $conexion = new PDO("mysql:host={$this->servidor};dbname={$this->base_datos};charset=utf8", $this->usuario, $this->pass, $opciones);

        return $conexion;
    }catch(PDOException $e){
        echo 'Ha surgido Un error y no se puede conectar a la base de datos. Detalle: '.$e->getMessage();
        exit;
    }
}

```

→ Prueba de consultas: Se han ejecutado consultas SQL en el código PHP para verificar que se pueden enviar correctamente a la base de datos y obtener los resultados esperados. Ejemplo:

En esta función se muestra el método save() para guardar los datos del usuario, se puede comprobar en la Base de datos insertando en el formulario de registro y ver así como se ha registrado correctamente, en el método del controlador también he mandado un mensaje en el caso de que haya sido correcta la operación.

```

public function save(array $usuario) {
    // PARA INSERTAR DATOS EN LA BASE DE DATOS
    $sql = "INSERT INTO usuarios (dni, nombre, apellidos, email, telefono, password, rol, confirmado) VALUES (:dni, :nombre, :apellidos, :email, :telefono, :password, :rol, 0)";
    $dni = $usuario['dni'];
    $nombre = $usuario['nombre'];
    $apellidos = $usuario['apellidos'];
    $email = $usuario['email'];
    $telefono = $usuario['telefono'];
    $password = password_hash($usuario['password'], PASSWORD_BCRYPT, ['cost' => 4]); // para cifrar la contraseña // cost es las veces que se cifra
    $rol = $usuario['rol'];

    $consult = $this->conexion->prepare($sql);
    $consult->bindParam(':dni', $dni, PDO::PARAM_STR);
    $consult->bindParam(':nombre', $nombre, PDO::PARAM_STR);
    $consult->bindParam(':apellidos', $apellidos, PDO::PARAM_STR);
    $consult->bindParam(':email', $email, PDO::PARAM_STR);
    $consult->bindParam(':telefono', $telefono, PDO::PARAM_STR);
    $consult->bindParam(':password', $password, PDO::PARAM_STR);
    $consult->bindParam(':rol', $rol, PDO::PARAM_STR);

    try {
        $success = $consult->execute();

        // Cerrar la consulta
        $consult->closeCursor();

        if (!$success) {
            echo "Error al ejecutar la consulta.";
        }
    } catch (PDOException $err) {
        echo "Error: " . $err->getMessage();
    }
}

```

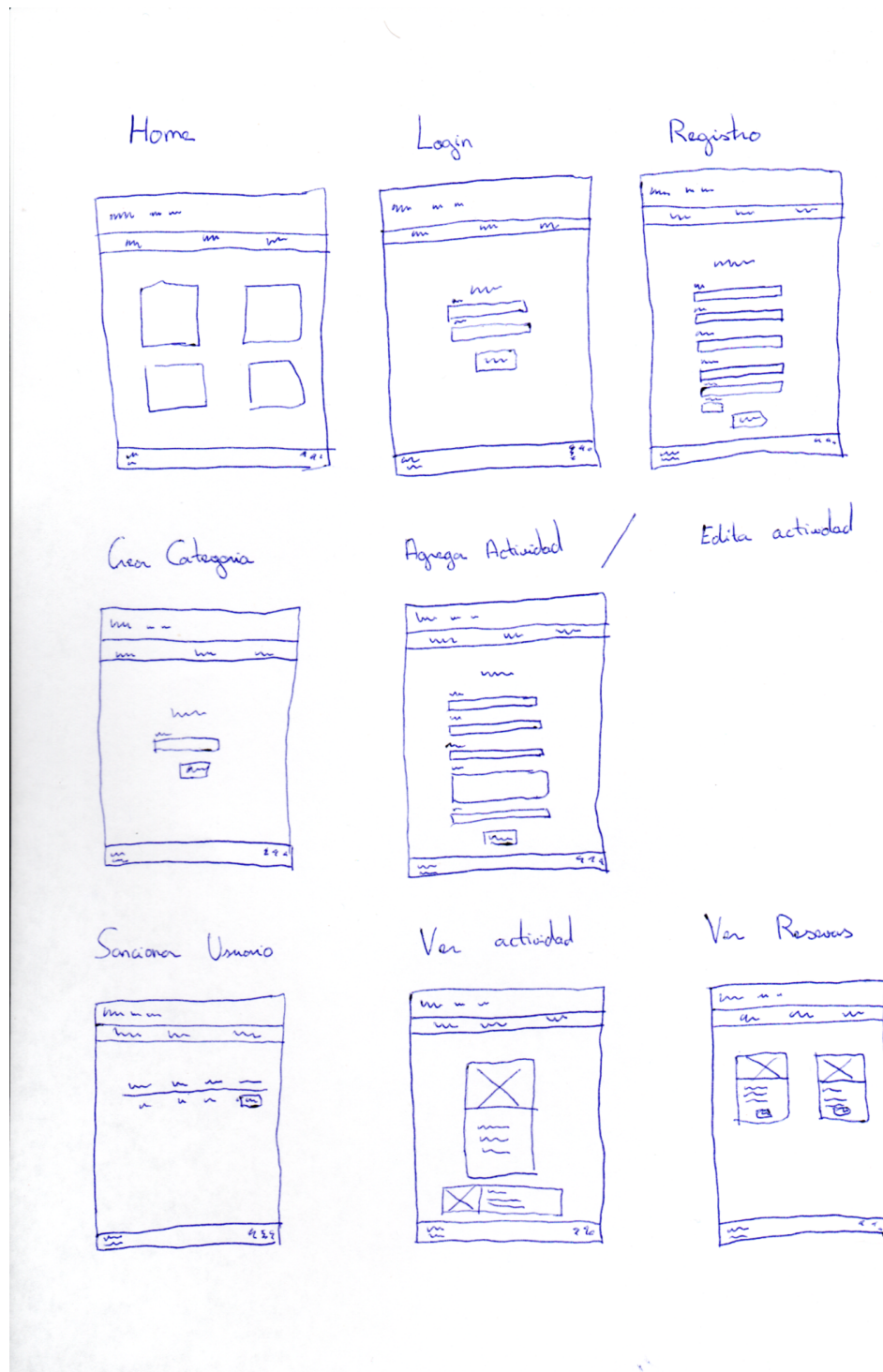
→ Prueba de errores: Se han controlado los errores de conexión o de consultas en la base de datos para verificar que la aplicación maneje adecuadamente estos casos. En cada método de los repositorios he añadido el try catch para manejar los errores en caso de que falle la operación.

- Prueba de validaciones de formularios: Existen validaciones que abarcan desde evitar la entrada de caracteres especiales en algunos campos, permitir solo números en determinados casos, hasta verificar la validez de un DNI mediante el cálculo de la letra correspondiente. Ejemplo: Si se introduce en el formulario un dni incorrecto mostrará el mensaje, o de alguna validación que se incluye en este archivo.

```
function validarFormularioRegistro() {  
    var formulario = document.querySelector('form');  
  
    // Validar DNI  
    var dniPattern = /^[0-9]{8}[TRWAGMYFPDXBNJZSQVHLCKET]{1}$/i;  
    var letrasDNI = 'TRWAGMYFPDXBNJZSQVHLCKET';  
  
    if (!dniPattern.test(formulario['data[dni]'].value)) {  
        document.getElementById('dni-error').textContent = 'El formato del DNI no es válido.';  
        return false;  
    }  
  
    var dni = formulario['data[dni]'].value.toUpperCase();  
    var letra = dni.charAt(dni.length - 1);  
    var numeros = dni.substr(0, dni.length - 1);  
  
    if (letrasDNI.charAt(numeros % 23) !== letra) {  
        document.getElementById('dni-error').textContent = 'El DNI no es válido.';  
        return false;  
    }  
  
    // Validar Nombre y Apellidos  
    var nombreApellidosRegex = /^[a-zA-ZñáéíóúÁÉÍÓÚÑR',.-]+$/;  
  
    var nombre = formulario['data[nombre]'].value;  
    var apellidos = formulario['data[apellidos]'].value;  
  
    if (!nombreApellidosRegex.test(nombre)) {  
        document.getElementById('nombre-error').textContent = 'El nombre solo debe contener letras y espacios.';  
        return false;  
    }  
  
    if (apellidos && !nombreApellidosRegex.test(apellidos)) {  
        document.getElementById('apellidos-error').textContent = 'Los apellidos solo deben contener letras y espacios.';  
        return false;  
    }  
  
    // Validar Email  
    var emailPattern = /^[^\\s@]+@[^\\s@]+\\.([^\\s@]+)$/;  
  
    if (!emailPattern.test(formulario['data[email]'].value)) {  
        document.getElementById('email-error').textContent = 'El formato del email no es válido.';  
        return false;  
    }  
  
    // Validar Teléfono  
    var telefonoPattern = /^\\d{9}$/;  
  
    if (!telefonoPattern.test(formulario['data[telefono]'].value)) {  
        document.getElementById('telefono-error').textContent = 'El formato del teléfono no es válido.';  
        return false;  
    }  
  
    // Validar Contraseña  
    if (formulario['data[password]'].value.length < 6) {  
        document.getElementById('password-error').textContent = 'La contraseña debe tener al menos 6 caracteres.';  
        return false;  
    }  
}
```

Manual de estilos

A. Sketches



Editar datos

Contacto

Notas

Ver listado

B. Criterios de accesibilidad

Para mejorar la accesibilidad de la aplicación web, se han implementado las siguientes estrategias:

- **Accesibilidad del formulario:** Usan etiquetas claras y asociadas correctamente a los campos de entrada. Proporcionan mensajes de error descriptivos y sugerencias para ayudar a los usuarios a completar los formularios de manera precisa.
- **Etiquetas y descripciones alternativas:** Se proporcionan etiquetas y descripciones alternativas para elementos como imágenes. Esto permite a los usuarios con discapacidades visuales puedan acceder y comprender el contenido.
- **Navegación clara y consistente:** Se ha utilizado una estructura de navegación lógica y se proporciona la ubicación actual del usuario dentro de la aplicación.

C. Criterios de usabilidad

Se han implementado los siguientes criterios de usabilidad en la aplicación web:

- **Facilidad de aprendizaje:** La aplicación es fácil de aprender y comprender para los usuarios nuevos. La navegación, funciones y controles son intuitivos y no requieren un esfuerzo significativo para comprender cómo se utiliza la aplicación.
- **Consistencia:** La aplicación tiene un diseño coherente y comportamiento en todas las secciones y pantallas. Los elementos en general tienen un aspecto uniforme, lo que facilita la comprensión y la navegación para los usuarios.
- **Retroalimentación informativa:** La aplicación proporciona una retroalimentación clara y relevante sobre las acciones realizadas por el usuario. Por ejemplo a la hora de registrarse o reservar una actividad.

D. Fuentes principales y secundarias

He utilizado la fuente Segoe UI es una fuente sans-serif que se caracteriza por su legibilidad y claridad en la pantalla. Su diseño es cuidadoso y su enfoque en la legibilidad la convierten en una buena opción. Tiene las siguientes características como son:

- Estilo moderno y limpio
- Diseño consistente
- Legibilidad en pantalla

E. Tamaños principales.

En la aplicación he usado Bootstrap que me proporciona clases de tamaños de columnas predefinidas como col-md que he ido usando en las vistas para mostrar la información. Además de usar elementos como el container-fluid.

F. Mapa de colores del proyecto

Para los colores he usado los colores que vienen con Bootstrap que son los siguientes:

- Primary (Header):
 - RGB (0,123,255)
 - Hexadecimal: #007BFF
 - Color: Azul
- Dark (Footer):
 - RGB (33,37,41)
 - Hexadecimal: #212529
 - Color: Negro
- Light(Letra en header):
 - RGB (248,249,250)
 - Hexadecimal: #F8F9FA
 - Color: Blanco
- Warning (Mensajes):
 - RGB (255,193,7)
 - Hexadecimal: #FFC107
 - Color: Oro

G. Dispositivos / vistas para las que se ha diseñado el proyecto

Bootstrap ofrece puntos de ruptura predefinidos que se ajustan a las características y dimensiones de los dispositivos. Ejemplo:

En la siguiente imagen se muestra la vista de `mis_reservas.php`

```
<div class="container mt-3">
  <div class="row row-cols-1 row-cols-md-2 g-4 m-5">
    <?php if(isset($reservas)):?>
      <?php foreach($reservas as $reserva):?>
        <div class="col mb-4">
          <div class="card h-100">
            " class="card-img-top" alt="Reserva" />
            <div class="card-body d-flex flex-column">
              <div class="card-title"><?php echo $reserva['nombre'];</div>
              <p class="card-text flex-grow-1"><?php echo $reserva['descripcion'];</p>
              <ul class="list-group list-group-flush text-center">
                <li class="list-group-item"><strong>Fecha:</strong> <?php echo $reserva['fecha'];</li>
                <li class="list-group-item"><strong>Fecha de reserva:</strong> <?php echo $reserva['fecha_reserva'];</li>
                <li class="list-group-item"><strong>Localización:</strong> <?php echo $reserva['localizacion'];</li>
                <li class="list-group-item"><strong>Duración:</strong> <?php echo $reserva['duracion'];</li>
              </ul>
              <?php if(isset($_SESSION['usuario'])):?>
                <li class="list-group-item">
                  <form action="cancelar_reserva" method="post">
                    <input type="hidden" name="data[id_reserva]" value="<?php echo $reserva['id_reserva'];>" />
                    <input type="hidden" name="data[id_actividad]" value="<?php echo $reserva['id_actividad'];>" />
                    <input type="hidden" name="data[id_usuario]" value="<?php echo $_SESSION['id_usuario'];>" />
                    <button type="submit" class="btn btn-primary bg-danger">Cancelar</button>
                  </form>
                </li>
              </if>
            </div>
          </div>
        </div>
      <?php endforeach; ?>
    <?php endif; ?>
  </div>
</div>
```

H. Software utilizado

En la aplicación se ha utilizado Visual Studio Code como entorno de desarrollo, Bootstrap como framework de front-end, PHP y JavaScript como lenguajes de programación, MySQL como gestor de bases de datos, Apache como servidor web y Git para el control de versiones.

I. Mejoras posibles

En vez de implementar el botón de scroll hacia arriba, podría haberse tratado de implementar la paginación de actividades para no cargar mucho la página en caso de que tenga mucho tráfico. Además de mejorar el sistema de sanciones a los usuarios.



Enlaces de interés

Repositorio de Github

<https://github.com/amartingonz/granatourz-daw>

Proyecto en producción:

<https://granatourz.com/>



Bibliografía

Documentación de Bootstrap

<https://getbootstrap.com/docs/5.3/getting-started/introduction/>

Documentación de PHP

<https://www.php.net/docs.php>

Documentación JavaScript, PHP , SQL

<https://www.w3schools.com/>

Documentación encontrada en la plataforma del curso (PDF).