

**PROYECTO DE CICLO DE G. S. DE  
DESARROLLO DE APLICACIONES WEB**

**SHARESOUNDS**



**MIGUEL ROBLES GÁMEZ  
CURSO 2020/21**

## Sumario

1 INTRODUCCIÓN.....	3
1.1 PRESENTACIÓN Y OBJETIVOS.....	3
1.2 CONTEXTO Y PLANTEAMIENTO.....	3
1.3 ESTRUCTURA DEL DOCUMENTO.....	3
2. REQUERIMIENTOS.....	4
2.1 INTRODUCCIÓN.....	4
2.2 DESCRIPCIÓN DE LA APLICACIÓN.....	5
2.2.1 ENFOQUE DE LA APLICACIÓN.....	5
2.2.2 TIPOS DE USUARIO.....	5
2.2.3 DEPENDENCIAS DE LA APLICACIÓN.....	5
2.3 REQUERIMIENTOS.....	5
2.3.1 REQUISITOS FUNCIONALES.....	5
2.3.2 REQUISITOS NO FUNCIONALES.....	7
3. FASE DE ANÁLISIS.....	8
3.1 INTRODUCCIÓN.....	8
3.2 DIAGRAMA DE FRONTERA.....	9
3.3 CASOS DE USO.....	10
3.3.1 Caso de Uso “Registro”.....	10
3.3.2 Caso de Uso “Consultar Funcionamiento”.....	11
3.3.3 Caso de Uso “Iniciar Sesión”.....	11
3.3.4 Caso de Uso “Cerrar Sesión”.....	12
3.3.5 “Gestión de Playlist”.....	12
3.3.5.1 Caso de Uso “Crear Playlist”.....	13
3.3.5.2 Caso de Uso “Editar Playlist”.....	13
3.3.5.3 Caso de Uso “Borrar Playlist”.....	14
3.3.5.4 Caso de Uso “Leer Playlist”.....	14
3.3.5.5 Caso de Uso “Buscar Playlist Usuarios”.....	15
3.3.6 “Gestión de Música en Playlists”.....	15
3.3.6.1 Caso de Uso: “Añadir Pista”.....	16
3.3.6.2 Caso de Uso: “Borrar Pista”.....	17
3.3.6.3 Caso de Uso: “Reproducir Lista”.....	17
4. DISEÑO.....	18
4.1 INTRODUCCIÓN.....	18
4.2 CAPA DE PRESENTACIÓN.....	18
4.3 CAPA LÓGICA DE LA APLICACIÓN.....	20
4.4 CAPA DE PERSISTENCIA DE DATOS.....	20
5. IMPLEMENTACIÓN.....	21
5.1 ARQUITECTURA DE LA APLICACIÓN.....	21
5.2 TECNOLOGÍAS USADAS.....	23
6. CONCLUSIONES.....	26
6.1 POSIBLES MEJORAS.....	26
7. BIBLIOGRAFÍA.....	27
ANEXO – A.....	29
ANEXO – B.....	30
ANEXO – C.....	33

## Índice de figuras

Figura 1: Diagrama de frontera.....	10
Figura 2: Gestión de Playlist.....	13
Figura 3: Gestión de música en Playlist.....	17
Figura 4: Esquema capas de diseño del proyecto.....	19
Figura 5: Diseño de reproductor de pistas.....	20
Figura 6: Iconos Php y JavaScript.....	21
Figura 7: Diagrama “entidad-relación” de la base de datos.....	22
Figura 8: Diagrama arquitectura cliente/servidor.....	24
Figura 9: Página de inicio de la aplicación web.....	32
Figura 10: Formulario de identificación.....	33
Figura 11: Lista de Playlist.....	33
Figura 12: Búsqueda de listas.....	34
Figura 13: Reproductor funcionando.....	34

## **1 INTRODUCCIÓN**

### **1.1 PRESENTACIÓN Y OBJETIVOS**

Este documento describe el trabajo realizado para el proyecto final de Grado Superior de Desarrollo de Aplicaciones Web. El proyecto consiste en el desarrollo de una aplicación web responsiva, haciendo posible el acceso a la aplicación desde diferentes dispositivos y es accesible desde cualquier navegador por Internet.

El objetivo de dicha aplicación web es hacer posible almacenar música por los usuarios y crear listas de reproducción “playlist”<sup>9</sup> desde diferentes plataformas y/o apps.

Se trata de facilitar el acceso de los usuarios a sus preferencias musicales. La funcionalidad de la web es dividida dependiendo del tipo de usuario:

**Usuarios no registrados:** Podrán acceder a la portada de la web que muestra información general de la misma además de mostrar las funcionalidades de la aplicación, viendo lo que ofrece y analizar si se adapta a sus necesidades y preferencias, además de poder acceder a un apartado explicativo del funcionamiento de la aplicación, por último podrá registrarse en el sistema.

**Usuarios registrados:** Podrán acceder a las listas de reproducción con privacidad de tipo pública de otros usuarios registrados, así como sus propias listas. Crear, editar, leer y borrar sus listas de reproducción. Crear, leer y borrar pistas<sup>8</sup> de música dentro de sus lista de reproducción.

Las funciones de la aplicación se explicarán con más detalle en futuros apartados.

### **1.2 CONTEXTO Y PLANTEAMIENTO**

El proyecto ha sido realizado como trabajo de fin de grado de Desarrollo de Aplicaciones Web por Miguel Robles Gámez, alumno del I. E. S. Virgen del Carmen de Jaén.

Para el desarrollo de la aplicación comprobé la posibilidades ofrecidas por distintas aplicaciones ya existentes de reproducción de sonido, en este caso Youtube<sup>12</sup> y Soundcloud<sup>11</sup>, para usar sus API<sup>1</sup> con el objetivo de reproducir la música usando sus servidores, siendo la aplicación ShareSounds una biblioteca de música donde almacenar en un solo lugar música de los distintos sitios, facilitando así la organización a usuarios que sean clientes de estas aplicaciones y deseen poder acceder a todo lo que escuchan con facilidad y rapidez.

### **1.3 ESTRUCTURA DEL DOCUMENTO**

Voy a describir brevemente el contenido de esta documentación:

En el apartado 1. INTRODUCCIÓN, se describe el propósito y finalidad de la aplicación, cuales son los objetivos más claros a conseguir, el tema del que se va a realizar la aplicación y cual es el contexto de esta documentación

En el apartado 2. REQUERIMIENTOS, se describirá de forma general el sistema a construir, especificando el enfoque de la aplicación, los tipos de actores que podrán realizar acciones dentro de la aplicación, las dependencias a ciertas tecnologías y/o hardware y, por último, hablar de forma un poco más específica sobre los requisitos de la aplicación, tanto los funcionales, como los no funcionales.

En el apartado 3. FASE DE ANÁLISIS, comenzaré a especificar cual será la estructura de la aplicación, para ello me ayudaré con diagramas, los cuales especificaré en su debido apartado, y comentaré en detalle las acciones que podrán realizar los actores o usuarios, así como las excepciones o errores que pueden surgir dependiendo o no de las acciones del usuario.

En el apartado 4. DISEÑO, especificaré los patrones de diseño usados para crear cada una de las capas necesarias para el funcionamiento del sistema, y describiré en detalle la información de cada una de ellas, ayudándome cuando sea necesario de esquemas, diagramas e imágenes, ya sean creados por mi o sacados de otras fuentes<sup>4</sup> con su respectiva mención en la bibliografía.

En el apartado 5. IMPLEMENTACIÓN, hablaré en detalle sobre la arquitectura sobre la que se basa el sistema de la aplicación web, cuales son sus características, ventajas y limitaciones, así como de su implementación, tal y como dice el título del apartado,

En el apartado 6. CONCLUSIONES, comentaré como ha sido el desarrollo de la aplicación para mi personalmente, que pasos tomé previos a la realización de la documentación además de cualquier tipo de aclaración u observación que desee dar a conocer.

## **2. REQUERIMIENTOS**

### **2.1 INTRODUCCIÓN**

El fin de este apartado es presentar las funcionalidades, características, y requisitos de la aplicación web.

El objetivo de la aplicación web principalmente es el facilitar el acceso a la música a los usuarios de distintas fuentes, siendo un requisito que el usuario se sienta familiar con el reproductor y la facilidad de uso de la aplicación.

## **2.2 DESCRIPCIÓN DE LA APLICACIÓN**

A continuación se presentará una descripción general del sistema, hablando de su enfoque, funciones, tipos de usuario y dependencias de forma más específica

### **2.2.1 ENFOQUE DE LA APLICACIÓN**

La aplicación se enfoca en facilitar la organización de la música a los usuarios registrados, ofreciendo un sistema sencillo de usar, incluso para aquellos usuarios sin tanta experiencia en el uso de aplicaciones de este estilo. Será posible utilizar la aplicación en todo tipo de dispositivos con acceso a un navegador web.

### **2.2.2 TIPOS DE USUARIO**

- **Usuario No Registrado:** Este tipo de usuarios solamente tendrán acceso a la información de uso de la aplicación web, , además de la posibilidad de registrarse en el sistema.

- **Usuario Registrado:** Este tipo de usuarios tendrán acceso a todo el funcionamiento del sistema de listas de reproducción y reproducción de música, además de poder acceder a playlists públicas de otros usuarios.

### **2.2.3 DEPENDENCIAS DE LA APLICACIÓN**

Las aplicación web funciona independientemente del hardware del usuario, siendo accesible desde cualquier plataforma convencional, las únicas dependencias destacables son:

- Un terminal que disponga de un navegador web actual, y conexión a internet estable.
- El servidor web que aloje la aplicación web debe ser capaz de soportar PHP 7 y la base de datos MySQL.

## **2.3 REQUERIMIENTOS**

### **2.3.1 REQUISITOS FUNCIONALES**

Los requisitos funcionales del sistema son distintos dependiendo del tipo de usuario que use la aplicación:

#### **Usuario No Registrado:**

- **Consultar información del uso de la aplicación:** El usuario podrá acceder a un apartado explicativo donde se le mostrarán instrucciones del uso de la aplicación.

- **Registro:** El usuario podrá registrar sus datos en el sistema con el fin de acceder a las funcionalidades de la aplicación, se deberán de introducir el nombre de usuario, correo electrónico, y la contraseña a elección. Los datos deberán ser únicos en el sistema.

Usuario Registrado:

- **Consultar información del uso de la aplicación:** El usuario podrá acceder a un apartado explicativo donde se le mostrarán instrucciones del uso de la aplicación.
- **Cerrar Sesión:** El usuario registrado podrá terminar su sesión en el sistema, haciendo click en el botón de “Cerrar sesión” que siempre está presente en la barra de navegación.
- **Listar Playlists:** El usuario registrado podrá acceder a las playlists creadas por él mismo desde el botón “Mis Playlists” que siempre está presente en la barra de navegación.
- **Crear Playlists:** El usuario registrado podrá crear nuevas playlists usando el formulario accesible desde la barra de navegación del menú de playlists, deberá proporcionar el nombre deseado, el tipo de privacidad de la lista y una imagen que se presente en la lista de playlists.
- **Borrar Playlists:** El usuario registrado podrá borrar playlists existentes usando el botón de la “Papelera” accesible desde la lista de playlists.
- **Editar Playlists:** El usuario registrado podrá editar playlists ya existentes usando el formulario accesible desde el botón “Edición”, accesible desde la lista de playlists, deberá proporcionar un nuevo nombre en caso de que lo desee editar y el tipo de privacidad de la lista.
- **Acceder a Playlists:** El usuario registrado podrá acceder a sus playlists privadas y/o públicas haciendo click en las mismas desde la lista de playlists.
- **Buscar Playlists:** El usuario registrado podrá buscar playlists de otros usuarios desde el formulario accesible usando el botón “Compartir”, accesible desde la lista de playlists del usuario, deberá ofrecer el nombre de la/las playlist/s que desee encontrar, o el nombre del usuario del cual desee encontrar sus listas.

- **Añadir Pista:** El usuario registrado podrá añadir pistas de música a sus listas, desde el botón “Añade tu Música” accesible desde la lista de reproducción en cuestión, los datos solicitados variarán dependiendo de la fuente a elegir al añadir las pistas.
- **Borrar Pista:** El usuario registrado podrá borrar pistas de música de sus listas, desde el botón de la “Papelera”, accesible desde la lista de reproducción en cuestión.
- **Reproducir Pista:** El usuario registrado podrá seleccionar que pista reproducir haciendo click en el nombre de la misma, accesible desde la lista de reproducción en cuestión.

### 2.3.2 REQUISITOS NO FUNCIONALES

- **Rendimiento:** el servidor debe de proporcionar los datos necesarios en un tiempo aceptable y constante.
- **Disponibilidad:** el servicio debe de estar disponible 24 horas al día y 7 días a la semana, siempre que se disponga de una conexión a internet estable y un equipo desde el que acceder.
- **Estabilidad:** El servicio debe ser estable y evitar las caídas de servicio en la medida de lo posible.
- **Accesibilidad:** la aplicación debe ser accesible para usuarios familiarizados con el uso de este tipo de servicios, como para usuarios sin tanta experiencia en el uso de estas tecnologías, debe ser visualmente sencilla de comprender.
- **Documentación:** la aplicación contará con un apartado en el que se muestre cómo usar la misma.
- **Mantenibilidad:** la aplicación debe ser fácilmente mantenible, una de las prioridades será la escalabilidad de la aplicación, facilitando el añadir nuevas fuentes en futuras versiones y por consecuencia el mantenimiento de la aplicación web.
- **Escalabilidad:** desde el primer momento en el desarrollo será necesario tener en cuenta la posible adición de nuevas características.



- **Seguridad:** la aplicación debe de ser segura, se filtrarán los datos ofrecidos por el usuario y se podrá hacer uso de la aplicación una vez que el usuario se haya registrado y sus datos queden guardados en el sistema.
- **Privacidad:** Los datos de los usuarios registrados serán completamente privados y ningún otro usuario tendrá acceso a los datos del resto de usuarios.

### **3. FASE DE ANÁLISIS**

#### **3.1 INTRODUCCIÓN**

En al fase de análisis se busca ofrecer un conocimiento razonable del sistema a exponer, para realizar el análisis de esta aplicación web voy a hacer uso del UML (Unified Modeling Language). El UML dispone de una gran cantidad de diagramas que permiten presentar la funcionalidades, requisitos y complejidad de un sistema de forma sencilla y visual, en este documento vamos a utilizar el “Diagrama de frontera de Sirecope” y los diagramas de casos de uso, en el caso de que necesite exponer con más detalle las funcionalidades de un caso de uso.

### 3.2 DIAGRAMA DE FRONTERA

En primer lugar, me gustaría presentar de una forma visual y sencilla el funcionamiento completo de la aplicación, para ello voy a usar un “Diagrama de Frontera”, el cual funciona perfectamente para esta misma función:

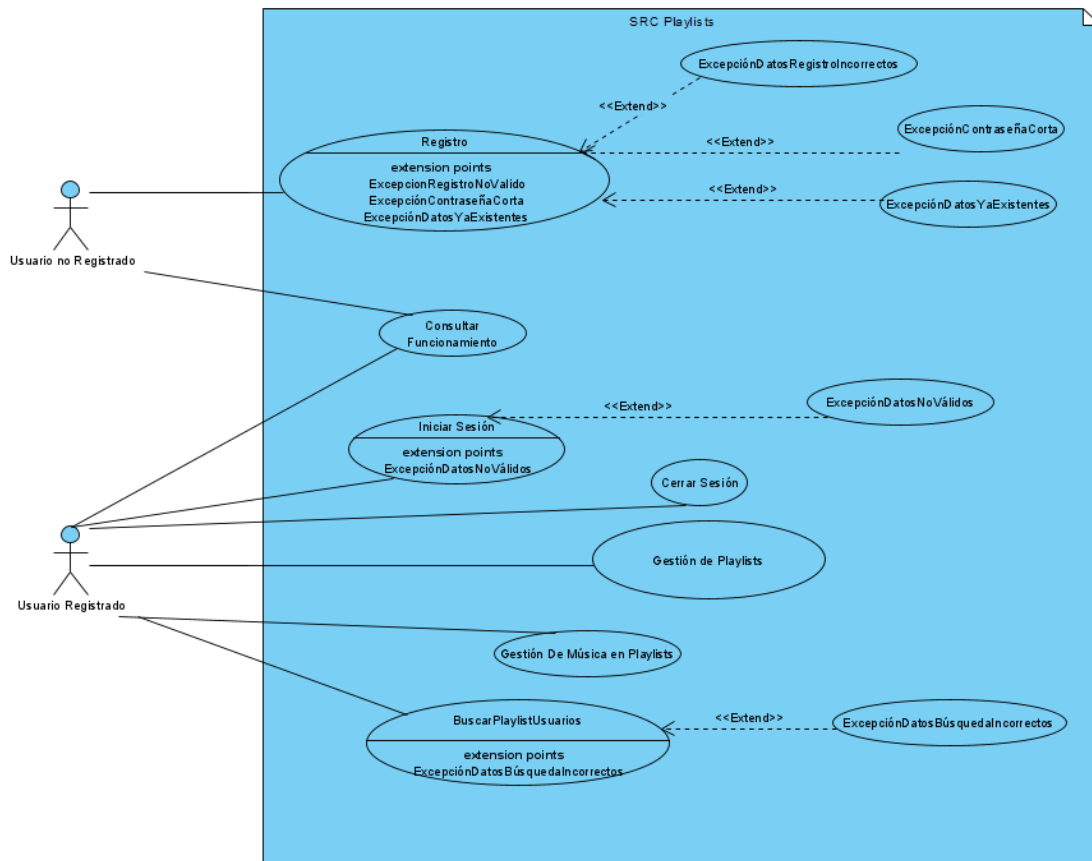


Figura 1: Diagrama de frontera.

En el diagrama se puede observar el funcionamiento general de la aplicación, podemos ver dos actores claros: el usuario no registrado y el usuario registrado, de la misma forma vemos las relaciones que tienen con el sistema, a lo que llamaremos a partir de ahora por su nombre en estos diagramas: casos de uso.

Por otra parte podemos observar que funciones del sistema pueden presentar errores al ejecutarse, aquellos casos de uso que cuentan con una relación del tipo “<<Extend>>”, con una “Excepción” pueden presentar fallos variando del tipo de función.

Precisamente lo que hace bien este diagrama es, por otro lado, también una desventaja, ya que no se puede definir con profundidad cada caso de uso, por lo que aquellos que tiene relación acaban juntándose en un solo caso, este es el caso de los casos de uso de “Gestión de Playlists”

y “Gestión de Música en Playlists”, los cuales son mucho mas extensos y debería de hablar de ellos en específico en un apartado concreto, este es el fin del siguiente apartado: 3.3 Casos de uso.

### 3.3 CASOS DE USO

Los objetivos de los casos de uso son ser herramientas simples para describir el comportamiento del software o de los sistemas.

Un caso de uso contiene una descripción textual de todas las maneras que los actores previstos podrían trabajar con el software o el sistema. Los casos de uso no describen ninguna funcionalidad interna del sistema, ni explican cómo se implementará. Simplemente muestran lo que el actor hace o debe hacer para realizar una operación, además de los problemas que pueden surgir durante la ejecución de la función y los pasos que sigue el sistema para solucionarlos.

Los casos de uso cuentan con las siguientes características:

- **Actor:** Representa al usuario realizando la acción y define su tipo.
- **Condiciones de entrada:** Los requisitos para que el caso de uso inicie.
- **Condiciones de salida:** El resultado final esperado por en caso de que no se recoja ninguna excepción causada por algún error.
- **Flujo de Eventos:** Define las pautas seguidas por el actor y el sistema para llegar de forma exitosa a las condiciones de salida.

En primer lugar hablaré de los casos de uso del Diagrama de Frontera (Figura), y cuando alcance a los casos de uso a ampliar (“Gestión de Playlist” y “Gestión De Música”), hablaré de cada uno de sus casos de uso en detalle, incorporando para cada uno de ellos un diagrama de casos de uso.

#### 3.3.1 Caso de Uso “Registro”

- Actor: Usuario no registrado.
- Condiciones de entrada: El usuario no se encuentra registrado en la aplicación.
- Condiciones de salida: El usuario es registrado en la aplicación.
- Flujo de eventos:
  1. El usuario decide registrarse.
  2. La aplicación solicita la información necesaria.
  3. El usuario proporciona la información solicitada.

4. El sistema verifica la información y en caso de que reconozca como válidos los datos, se informa al usuario del correcto registro.

**- Excepciones:**

**- Excepción Datos Registro Incorrectos:** los datos proporcionados no son válidos.

1. La aplicación muestra mensaje de error “Los datos son incorrectos”
2. La aplicación vuelve a solicitar los datos del usuario. (Paso 2)

**- Excepción Contraseña Corta:** la contraseña proporcionada es menor de 6 caracteres.

1. La aplicación muestra el mensaje de error: “La contraseña debe ser mayor de 6 caracteres”
2. La aplicación vuelve a solicitar los datos al usuario. (Paso 2)

**-Excepción Datos Ya Existentes:** los datos proporcionados pertenecen a otro usuario.

1. La aplicación muestra el mensaje: “Los datos ya existen en el sistema”
2. La aplicación vuelve a solicitar los datos al usuario. (Paso 2)

### **3.3.2 Caso de Uso “Consultar Funcionamiento”**

- Actor: Usuario no registrado y usuario registrado.
- Condiciones de entrada: El usuario desea conocer el funcionamiento de la aplicación.
- Condiciones de salida: El usuario adquiere la información que necesita.
- Flujo de eventos:
  1. El usuario necesita información del funcionamiento de la aplicación.
  2. El usuario entra en el apartado “Cómo usar Sharesounds”
  3. La aplicación le muestra la información que necesita.

### **3.3.3 Caso de Uso “Iniciar Sesión”**

- Actor: Usuario registrado.
- Condiciones de entrada: El usuario desea identificarse en la aplicación.
- Condiciones de salida: El usuario se identifica en la aplicación con éxito.
- Flujo de eventos:
  1. El usuario se encuentra registrado en la aplicación.
  2. El usuario desea identificarse.
  3. La aplicación solicita los datos al usuario.
  4. El usuario proporciona la información solicitada.
  5. El sistema verifica la información y el usuario se identifica con éxito.

**-Excepciones:**

- **ExcepciónDatosNoVálidos:** Los datos proporcionados no son válidos.

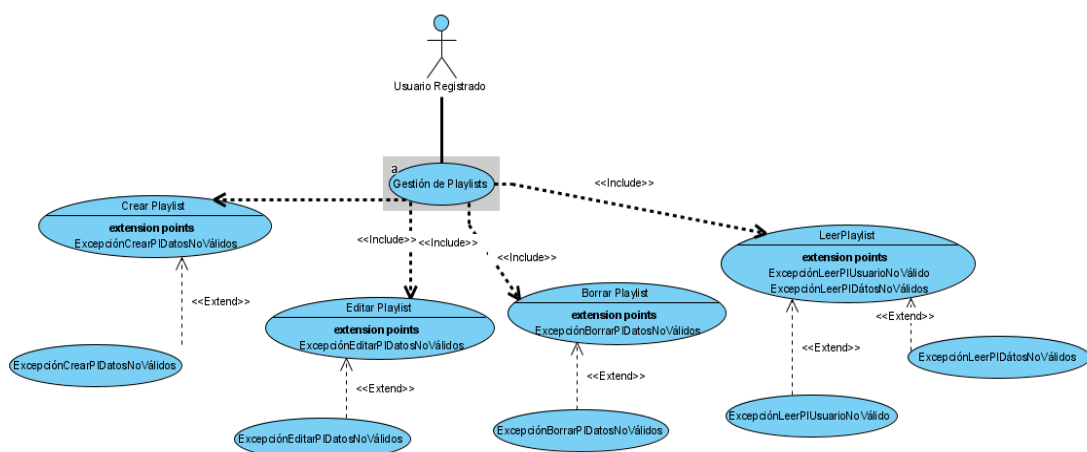
1. La aplicación muestra mensaje de error “Los datos son incorrectos”
2. La aplicación vuelve a solicitar los datos al usuario. (Paso 2)

**3.3.4 Caso de Uso “Cerrar Sesión”**

- Actor: Usuario registrado.
- Condiciones de entrada: El usuario desea cerrar su sesión en la aplicación.
- Condiciones de salida: La sesión del usuario termina exitosamente.
- Flujo de eventos:
  1. El usuario decide cerrar su sesión.
  2. El usuario solicita a la aplicación el cierre de sesión.
  3. La aplicación cierra la sesión.

**3.3.5 “Gestión de Playlist”**

Voy a ampliar en este caso de uso, ya que en el Diagrama de Frontera (Figura ) no me cabía la posibilidad de entrar en detalle sobre los casos de uso dentro de la gestión de Playlists, por lo tanto incorporo un diagrama de casos de uso de “Gestión de Playlists”.



*Figura 2: Gestión de Playlist.*

A partir de aquí describiré los casos de uso de “Gestión de Playlists”

### 3.3.5.1 Caso de Uso “Crear Playlist”

- Actor: Usuario registrado.
- Condiciones de entrada: El usuario desea crear una playlist.
- Condiciones de salida: El usuario crea una playlist correctamente.
- Flujo de eventos:
  1. El usuario decide crear una playlist.
  2. El usuario solicita a la aplicación crear una playlist.
  3. El sistema solicita los datos necesarios para la creación de la misma.
  4. El usuario ofrece los datos requeridos a la aplicación.
  5. El sistema procesa los datos.
  6. La playlist es creada.

#### -Excepciones:

**-Excepción CrearPIDatosNoVálidos:** los datos introducidos no son válidos.

1. La aplicación muestra el mensaje de error: “Los datos no son válidos”
2. El usuario ofrece los datos requeridos a la aplicación. (Paso 4)

### 3.3.5.2 Caso de Uso “Editar Playlist”

- Actor: Usuario registrado.
- Condiciones de entrada: El usuario desea editar una playlist.
- Condiciones de salida: El usuario edita una playlist correctamente.
- Flujo de eventos:
  1. El usuario decide editar una playlist.
  2. El usuario solicita a la aplicación editar una playlist.
  3. El sistema solicita los datos necesarios para la edición de la misma.
  4. El usuario ofrece los datos requeridos a la aplicación.
  5. El sistema procesa los datos.
  6. La playlist es editada.

#### -Excepciones:

**-Excepción EditarPIDatosNoVálidos:** los datos introducidos no son válidos.

1. La aplicación muestra el mensaje de error: “Los datos no son válidos”
2. El usuario ofrece los datos requeridos a la aplicación. (Paso 4)

### 3.3.5.3 Caso de Uso “Borrar Playlist”

- Actor: Usuario registrado.
- Condiciones de entrada: El usuario desea borrar una playlist.
- Condiciones de salida: El usuario borra una playlist correctamente.
- Flujo de eventos:
  1. El usuario decide borrar una playlist.
  2. El usuario decide qué playlist desea borrar.
  3. El usuario solicita a la aplicación borrar la playlist seleccionada.
  4. El sistema solicita confirmación del usuario para borrar la playlist seleccionada.
  5. El usuario confirma al sistema su elección.
  6. El sistema lo borra.

#### -Excepciones:

**-Excepción Borrar Playlist Datos No Válidos:** Los datos introducidos no son válidos.

1. La aplicación muestra el mensaje de error: “No se ha podido completar la solicitud”
2. El usuario vuelve a solicitar a la aplicación borrar la playlist seleccionada. (Paso 3)

### 3.3.5.4 Caso de Uso “Leer Playlist”

- Actor: Usuario registrado.
- Condiciones de entrada: El usuario desea acceder una playlist.
- Condiciones de salida: El usuario accede a una playlist correctamente.
- Flujo de eventos:
  1. El usuario decide acceder a una playlist.
  2. El usuario selecciona a qué playlist desea acceder.
  3. El usuario solicita a la aplicación acceder a la playlist seleccionada.
  4. El sistema procesa la solicitud.
  5. El sistema muestra al usuario la playlist seleccionada.
  6. El usuario accede a la playlist.

#### -Excepciones:

**-Excepción Leer Playlist Usuario No Válido:** la privacidad de la playlist es privada y el usuario no es el dueño de la playlist.

1. La aplicación muestra el mensaje de error: “No tiene permiso para acceder a la playlist”
2. El usuario selecciona a qué playlist desea acceder. (Paso 2)

**-ExcepciónLeerPIDatosNoVálidos:** los datos de la playlist seleccionada no existen o no son válidos.

1. La aplicación muestra el mensaje de error: “Los datos de la lista no son válidos”
2. El usuario selecciona a qué playlist desea acceder. (Paso 2)

### 3.3.5.5 Caso de Uso “Buscar Playlist Usuarios”

- Actor: Usuario registrado.
- Condiciones de entrada: El usuario desea encontrar playlist de otros usuarios registrados.
- Condiciones de salida: El usuario encuentra una playlist de otro usuario.
- Flujo de eventos:
  1. El usuario desea encontrar playlist de otros usuarios registrados.
  2. El sistema solicita los datos al usuario.
  3. El usuario ofrece a la aplicación los datos solicitados.
  4. La aplicación procesa los datos proporcionados.
  5. La aplicación muestra los resultados de la búsqueda al usuario.
  6. El usuario elige la playlist que desee.

**-Excepciones:**

**-ExcepciónDatosDeBusquedaIncorrctos:** Los datos proporcionados no son válidos.

1. La aplicación muestra el mensaje: “No se han encontrado listas con estos datos”
2. El usuario debe introducir nuevamente los datos solicitados. (Paso 2)

### 3.3.6 “Gestión de Música en Playlists”

Voy a ampliar en este caso de uso, ya que en el Diagrama de Frontera (Figura ) no cabía la posibilidad de entrar en detalle sobre los casos de uso dentro de la gestión de música en playlists, por lo tanto incorporo un diagrama de casos de uso de “Gestión de Música en Playlists”.



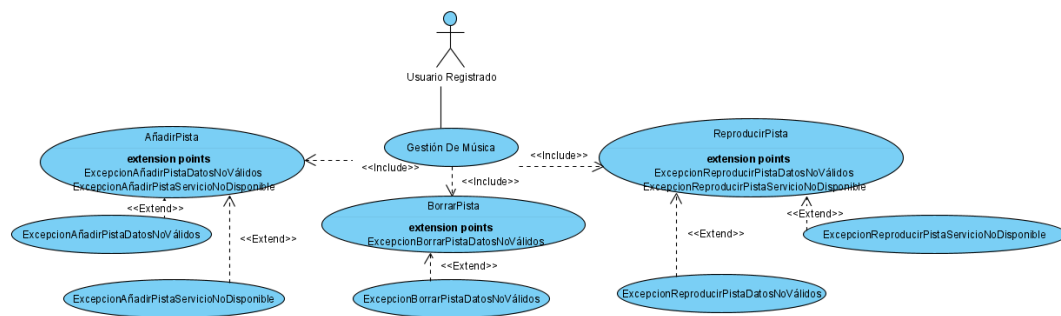


Figura 3: Gestión de música en Playlist

A partir de aquí describiré los casos de uso de “Gestión de Playlists”

### 3.3.6.1 Caso de Uso: “Añadir Pista”

- Actor: Usuario registrado.
- Condiciones de entrada: El usuario desea añadir pistas a su playlist.
- Condiciones de salida: El usuario añade una pista a su playlist exitosamente.
- Flujo de eventos:
  1. El usuario desea añadir pistas a su playlist.
  2. El sistema solicita los datos de la fuente de la pista a añadir.
  3. El sistema solicita los datos necesarios al usuario.
  4. El usuario ofrece a la aplicación los datos solicitados.
  5. La aplicación añade la pista correctamente.

#### -Excepciones:

**-ExcepciónAñadirPistaDatosNoVálidos:** Los datos de la pista proporcionados no son válidos.

1. La aplicación muestra el mensaje: “Los datos no son válidos”
2. El sistema solicita los datos de la fuente de la pista a añadir. (Paso 2)

**-ExcepciónAñadirPistaServicioNoDisponible:** El servicio del que se solicita la pista no responde.

1. La aplicación muestra el mensaje: “El servicio del que se solicita la pista no responde. Inténtelo más tarde”
2. El sistema solicita los datos de la fuente de la pista a añadir. (Paso 2)

### 3.3.6.2 Caso de Uso: “Borrar Pista”

- Actor: Usuario registrado.

- Condiciones de entrada: El usuario desea borrar pistas de su playlist.
- Condiciones de salida: El usuario borra una pista de su playlist exitosamente.
- Flujo de eventos:
  1. El usuario desea borrar pistas de su playlist.
  2. El usuario selecciona qué pista quiere borrar.
  3. El usuario pide al sistema que borre la pista.
  4. El sistema borra la pista seleccionada por el usuario.

**-Excepciones:**

**-ExcepciónBorrarPistaDatosNoVálidos:** Los datos de la pista proporcionados no son válidos.

1. La aplicación muestra el mensaje: “Los datos no son válidos”
2. El usuario selecciona qué pista quiere borrar. (Paso 2)

**3.3.6.3 Caso de Uso: “Reproducir Lista”**

- Actor: Usuario registrado.
- Condiciones de entrada: El usuario desea reproducir pistas de una playlist.
- Condiciones de salida: El usuario reproduce pistas de una playlist exitosamente.
- Flujo de eventos:
  1. El usuario desea reproducir pistas de una playlist.
  2. El usuario elige la pista que desea reproducir.
  3. El sistema procesa la solicitud.
  4. El sistema reproduce la pista que el usuario ha elegido.

**-Excepciones:**

**-ExcepciónReproducirPistaDatosNoVálidos:** Los datos de la pista proporcionados no son válidos.

1. La aplicación muestra el mensaje: “Los datos no son válidos”
2. El usuario elige la pista que desea reproducir. (Paso 2)

**-ExcepciónReproducirPistaServicioNoDisponible:** El servicio del que se solicita la pista no responde.

1. La aplicación muestra el mensaje: “El servicio del que se solicita la pista no responde. Inténtelo más tarde”
2. El usuario elige la pista que desea reproducir. (Paso 2)

## 4. DISEÑO

### 4.1 INTRODUCCIÓN

El objetivo de esta fase es analizar y definir cada una de las capas necesarias para el funcionamiento de la aplicación, las capas a enumerar son:

- **Capa de Presentación:** Donde se define la estructura de la interfaz gráfica de la aplicación.

- **Capa lógica de la aplicación:** Donde se define el comportamiento de los componentes software y se especifican sus funciones.

- **Capa de persistencia de datos:** La capa de persistencia corresponde con la base de datos de la aplicación, en este apartado se describirá como se creará la base de datos, así como su estructura.

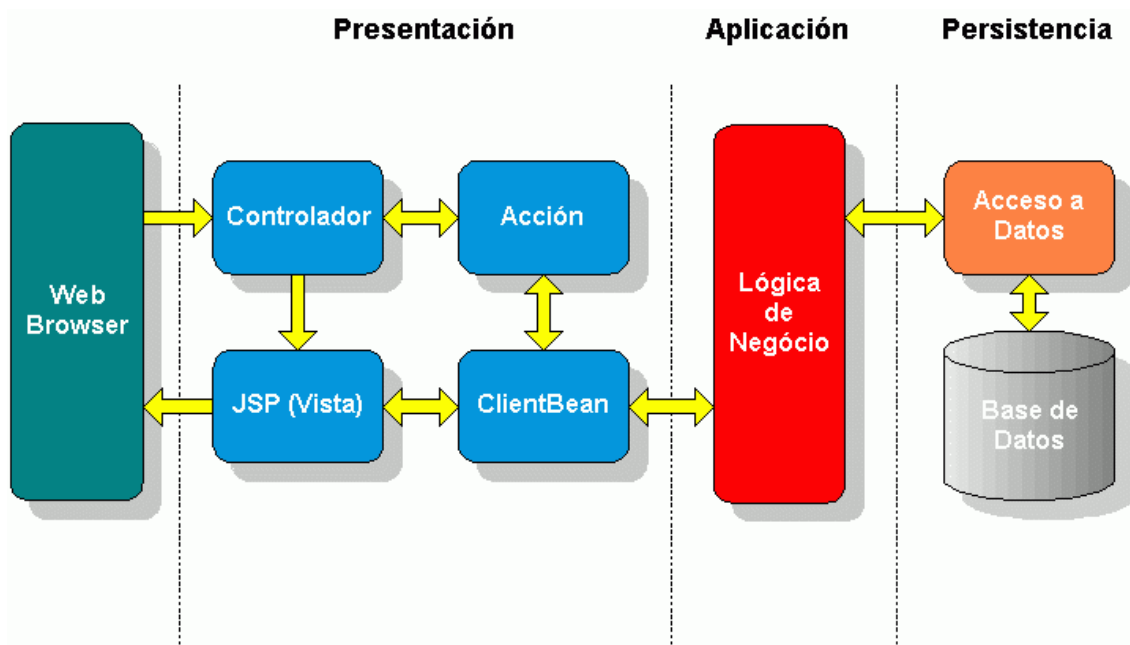


Figura 4: Esquema capas de diseño del proyecto

### 4.2 CAPA DE PRESENTACIÓN

El diseño web implica el trabajo relacionado con el “layout”<sup>6</sup> y diseño de páginas online, así como con la producción de contenido, aunque generalmente se aplica a la creación de sitios web.

Normalmente se compone principalmente de dos tecnologías, HTML y CSS, de las que hablaré en más detalle en futuros apartados.

Algunos elementos clave del diseño web son:

- **Responsividad:** Busca que la web sea completamente disponible desde cualquier dispositivo, ya sea smartphone, tablet, monitor, o televisor.
- **Escaneabilidad:** Define la facilidad de lectura de tu sitio web, por ejemplo, separando el contenido de la misma en bloques fácilmente identificables.
- **Tipografía:** Estilo de los textos y elementos de la web, se busca que tengan una cierta conformidad.
- **Velocidad de Carga:** Define el tiempo que le toma a la aplicación o sitio web en mostrar datos al usuario y permitirle interactuar con la misma.

Dejando claro en qué consiste la capa de presentación, veamos como he organizado el apartado en el que los usuarios pasarán mas tiempo, el apartado del reproductor de pistas.

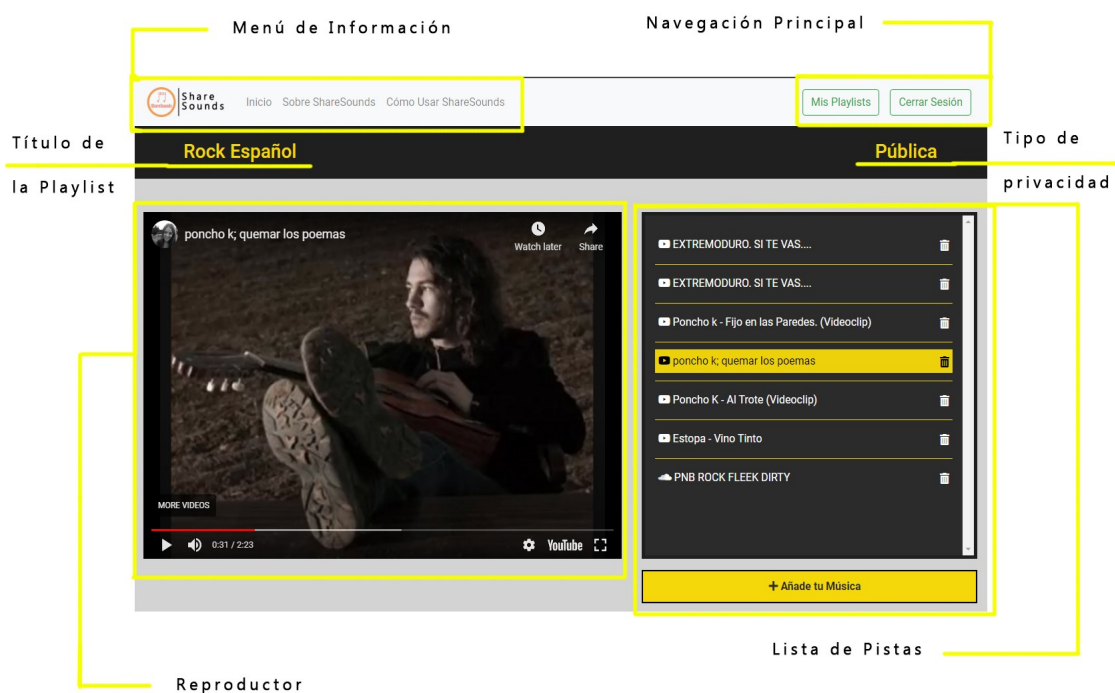


Figura 5: Diseño de reproductor de pistas.

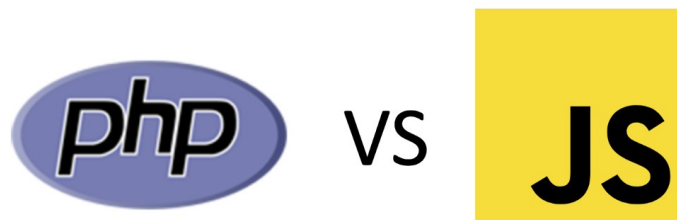
Para este apartado he optado por un diseño claro y sencillo, en el que el usuario siempre tiene a la vista los elementos importantes de la navegación y el uso del reproductor, hablaré más en el detalle de cada apartado:

- **Menú de Información:** Es el menú desde el que se accede a la información de la aplicación y de su uso, está siempre visible.
- **Navegación Principal:** Es el menú desde el que se accede a la lista de Playlists del usuario y se puede cerrar su sesión, está siempre visible.
- **Tipo de Privacidad:** Se muestra qué tipo de lista es, pública si pueden acceder a ella otros usuarios, y privada si solo puede acceder el dueño de la lista.

- **Título de la Playlist:** El nombre elegido por el usuario dueño de la playlists para su clasificación.
- **Reproductor:** El elemento donde se reproducirán las pistas, será distinto dependiendo de la fuente de la que se saque la información de la pista.
- **Lista de Pistas:** Muestra las pistas que el usuario tiene añadidas a la lista, además muestra el botón de borrar las pistas y añadir las pistas en caso de que acceda el dueño de la lista.

#### 4.3 CAPA LÓGICA DE LA APLICACIÓN

La capa de lógica de la aplicación puede definirse como todas aquellas piezas de software que, en conjunto, consiguen llevar a cabo el funcionamiento operacional de la aplicación web, en el caso de este documento, las piezas están construidas usando las tecnologías JavaScript y PHP principalmente, ocupándose cada una de ellas de un trabajo en específico.



*Figura 6: Iconos Php y JavaScript*

En nuestro caso, JavaScript se ha encargado de todas las funcionalidades ejecutadas en el equipo cliente, siendo su trabajo reproducir la lógica de la aplicación, permitir actualizaciones del sitio sin tener que refrescar la página, cambiar los estilos del documento dinámicamente, y realizar las llamadas al servidor de forma asíncrona, con el objetivo de ofrecerle datos al instante al usuario final, sin que este se de cuenta de que esta realizando peticiones al programa servidor.

PHP ha sido el principal lenguaje usado en el entorno del servidor, ha sido usado mayoritariamente para consultar datos a la capa de persistencia de datos, verificar los formularios de identificación y registro de usuarios, creación de sitios web dinámicos, dependiendo sobre todo en que tipo de usuario visualice la página, así como el encargado de devolverle los datos recibidos de la base de datos al lenguaje cliente, JavaScript.

#### 4.4 CAPA DE PERSISTENCIA DE DATOS

Una capa de persistencia encapsula el comportamiento necesario para mantener los datos. O sea: leer, escribir y borrar datos en el almacenamiento persistente o base de datos. La persistencia de la información es la parte más crítica en una aplicación de software.

Con el objetivo de explicar la forma en la que se ha organizado el sistema de persistencia de datos de la aplicación, se muestra a continuación el diagrama “entidad-relación” de la base de datos de la aplicación:

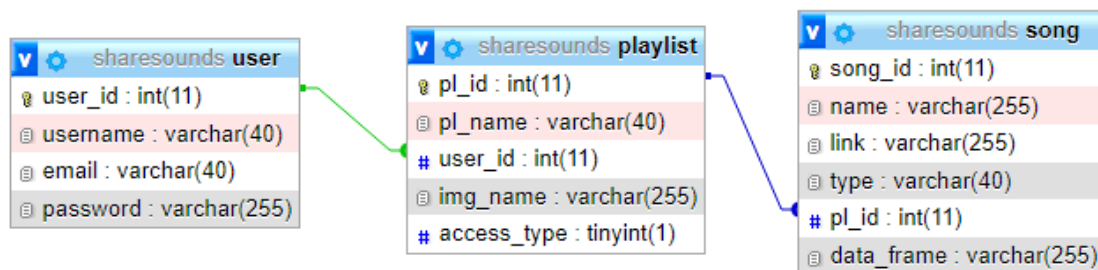


Figura 7: Diagrama “entidad-relación” de la base de datos.

El esquema es bastante explicativo por si mismo, sin embargo, listaré y describiré de forma sencilla cada tabla y sus atributos.

- **User:** Guarda información del usuario registrado.
  - **user\_id:** Almacena el número identificador del usuario registrado.
  - **username:** Almacena el nick<sup>7</sup> del usuario registrado.
  - **email:** Almacena el correo electrónico del usuario registrado.
  - **password:** Almacena la contraseña encriptada del usuario registrado.
- **playlist:** Guarda información sobre una playlist.
  - **pl\_id:** Almacena el número identificador de la playlist.
  - **pl\_name:** Almacena nombre elegido de la playlist.
  - **user\_id:** Almacena el usuario dueño de la playlist, el usuario debe existir previamente en el sistema para almacenarlo, en caso de que el usuario sea borrado, también serán borrada la/las entrada/s de ese usuario en esta tabla.
  - **img\_name:** Almacena el nombre de la imagen de la playlist.

- **access\_type:** Almacena el tipo de acceso de la playlist.
- **song:** Guarda información sobre una pista.
  - **song\_id:** Almacena el número identificador de la pista.
  - **name:** Almacena el nombre de la pista.
  - **link:** Almacena el enlace al servicio correspondiente de la pista.
  - **type:** Almacena el tipo de servicio de la pista.
  - **pl\_id:** Almacena el número identificador de la playlist, la playlist debe existir previamente en el sistema para almacenarla, en caso de que la playlist sea borrada, también serán borradas la/las entrada/s de dicha playlist en esta tabla.
  - **data\_frame:** Almacena el datos necesarios para solicitar la pista al servicio.

## 5. IMPLEMENTACIÓN

En este apartado describiré la arquitectura de la aplicación y las tecnologías utilizadas, describiré cada una de ellas con la ayuda de recursos externos, mencionados en la bibliografía.

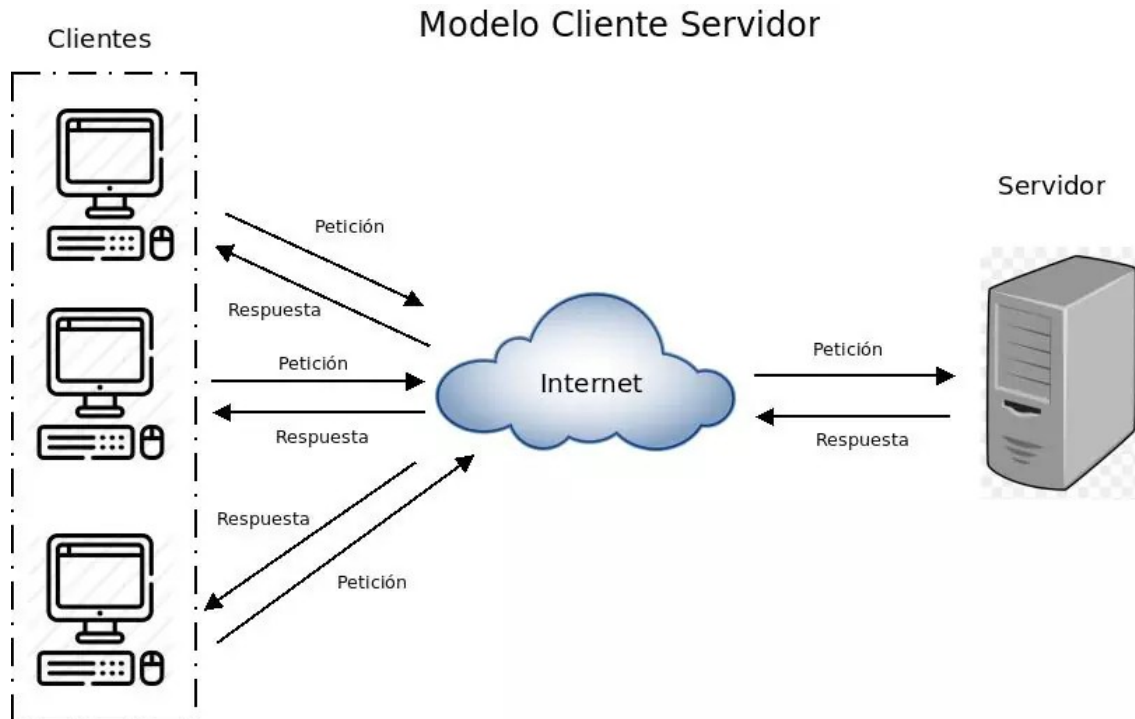
### **5.1 ARQUITECTURA DE LA APLICACIÓN**

La aplicación se basa en la arquitectura cliente/servidor, es un modelo de diseño de software en el que las tareas se reparten entre los proveedores de recursos o servicios y los clientes.

Esta arquitectura es utilizada en multitud de aplicaciones usadas diariamente, como son por ejemplo el correo electrónico, los servidores de impresión y las páginas web en general, dicho esto, procedo a explicar las diferencias entre el cliente y el servidor:

- **Cliente:** El concepto de cliente hace referencia a un demandante de servicios, este cliente puede ser un ordenador como también una aplicación de informática, la cual requiere información proveniente de la red para funcionar.
- **Servidor:** Un servidor hace referencia a un proveedor de servicios, este servidor a su vez puede ser un ordenador o una aplicación informática que envía información a los demás agentes de la red.

Adjunto un diagrama auxiliar con el fin de aclarar el funcionamiento de la arquitectura cliente/servidor.



*Figura 8: Diagrama arquitectura cliente/servidor.*

## 5.2 TECNOLOGÍAS USADAS

- **HTML:**

- **Uso:** Se ha utilizado para crear el contenido de texto y multimedia de la aplicación.
- **Descripción:** HTML significa lenguaje de marcado de hipertexto, y le permite al usuario crear y estructurar secciones, párrafos, encabezados, enlaces y elementos de cita en bloque (blockquotes) para páginas web y aplicaciones.

HTML no es un lenguaje de programación, lo que significa que no tiene la capacidad de crear una funcionalidad dinámica.

- **CSS:**

- **Uso:** Se ha utilizado para el diseño y creación del layout.
- **Descripción:** CSS son las siglas en inglés para «hojas de estilo en cascada» (cascading style sheets). CSS funciona a la par con HTML, añadiéndole la posibilidad de elegir cómo mostrar la información ofrecida por el HTML, cambiando el color de los elementos, su posicionamiento, el tamaño o incluso añadiendo reglas activadas al pasar el ratón por encima de los elementos.

- **SASS:**

- **Uso:** Se ha utilizado para facilitar las tareas de diseño con CSS.



- **Descripción:** SASS proviene de Syntactically Awesome Style Sheets, o lo que es lo mismo, hojas de estilo increíbles sintácticamente, facilita el trabajo con las hojas de estilo CSS, permitiendo la creación de funciones que realicen ciertas operaciones matemáticas, creación de variables y reutilización de estilos ya creados.
- **JavaScript:**
  - **Uso:** Se ha utilizado para crear la lógica de la aplicación del lado de el cliente.
  - **Descripción:** JavaScript es un lenguaje de scripting<sup>10</sup> o programación que permite implementar características complejas en los sitios web, permite realizar lo que cualquier lenguaje de programación permite, aplicado a la web, por lo que abre cientos de posibilidades al desarrollador para crear las aplicaciones y funciones que necesite, de una forma rápida y sencilla.
- **JQuery:**
  - **Uso:** Se ha utilizado junto con JavaScript para crear la lógica de la aplicación, en su mayoría para realizar las llamadas Ajax al servidor.
  - **Descripción:** JQuery es una biblioteca de JavaScript que simplifica la forma de desarrollar aplicaciones web. JQuery es usado en multitud de sitios web, este permite manipular elementos del DOM<sup>3</sup>, cambiar el diseño especificado en las hojas de estilo CSS o realizar peticiones AJAX de forma rápida, concisa y sencilla.
- **AJAX:**
  - **Uso:** Se ha utilizado para realizar las llamadas asíncronas al servidor.
  - **Descripción:** AJAX significa JavaScript asíncrono y XML (Asynchronous JavaScript and XML). Es un conjunto de técnicas de desarrollo web que permiten que las aplicaciones web funcionen de forma asíncrona, procesando cualquier solicitud al servidor en segundo plano.
- **SQL:**
  - **Uso:** Ha sido utilizado para la creación de la base de datos.
  - **Descripción:** El lenguaje de consulta estructurado (SQL) es el lenguaje de base de datos más implementado y valioso para cualquier persona involucrada en la programación informática o que usa bases de datos para recopilar y organizar información.
- **MySQL:**
  - **Uso:** Ha sido utilizado para la gestión de la base de datos.
  - **Descripción:** MySQL es el sistema de gestión de bases de datos relacional más extendido en la actualidad al estar basada en código abierto. MySQL trabaja con bases de datos relacionales, al ser basada en código abierto es fácilmente accesible y cuenta con una gran comunidad actualizándolo.
- **PhpMyAdmin:**
  - **Uso:** Se ha utilizado para facilitar la gestión de la base de datos.

- **Descripción:** es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web, utilizando un navegador web. Permite realizar todas las acciones que harías en tu MySQL habitual, de una forma más visual y sencilla, sin necesidad de conocer por completo el funcionamiento de MySQL.
- **PHP:**
  - **Uso:** Se ha utilizado como lenguaje de servidor para crear la lógica de la aplicación
  - **Descripción:** PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML. Se utiliza para general páginas web dinámicas, y especialmente, como lenguaje de servidor, comunicándose directamente con la base de datos con el fin de ofrecer una experiencia dinámica a los usuarios.
- **APACHE:**
  - **Uso:** Se ha utilizado como software de servidor
  - **Descripción:** Apache HTTP Server es un software de servidor web gratuito y de código abierto para plataformas Unix con el cual se ejecutan el 46% de los sitios web de todo el mundo. Es mantenido y desarrollado por la Apache Software Foundation.
- **Laragon:**
  - **Uso:** Se ha utilizado como suite para el desarrollo local de la aplicación.
  - **Descripción:** Laragon es una herramienta para equipos técnicos que permite crear diferentes entornos de desarrollo, facilitando el trabajo con las aplicaciones. Es normalmente comparado con XAMPP, pero tras la prueba de los dos sistemas, he preferido utilizar Laragon por su facilidad de uso y configuración.
- **Bootstrap:**
  - **Uso:** Se ha utilizado para la organización del layout y algunos estilos.
  - **Descripción:** Bootstrap es un framework CSS y Javascript diseñado para la creación de interfaces limpias y con un diseño responsivo. Además, ofrece un amplio abanico de herramientas y funciones, de manera que los usuarios pueden crear prácticamente cualquier tipo de sitio web haciendo uso de los mismos.
- **Github:**
  - **Uso:** Se ha utilizado como software de control de versiones de la aplicación.
  - **Descripción:** Github es un portal creado para alojar el código de las aplicaciones de cualquier desarrollador. La plataforma está creada para que los desarrolladores suban el código de sus aplicaciones y herramientas, y que como usuario no sólo puedas descargar la aplicación, sino también entrar a su perfil para leer sobre ella o colaborar con su desarrollo, Github utiliza el sistema de control de versiones creado por Linus Torvalds "Git".
- **Github Desktop:**

- **Uso:** Usado para facilitar el control de versiones con GitHub.
- **Descripción:** GitHub Desktop es una aplicación que te habilita para interactuar con GitHub utilizando una GUI en vez de la línea de comandos o de un buscador web.
- **Photoshop:**
  - **Uso:** Se ha utilizado para la creación y edición de elementos multimedia para la aplicación web y la documentación.
  - **Descripción:** Adobe Photoshop es un editor de fotografías desarrollado por Adobe Systems Incorporated. Usado principalmente para el retoque de fotografías y gráficos, considerada una de las mejores herramientas de su categoría, siendo utilizadas por profesionales de la imagen como diseñadores gráficos, fotógrafos, diseñadores web o ilustradores digitales, entre otros, el programa soporta toda clase de formatos de imagen.
- **Visual Paradigm:**
  - **Uso:** Se ha utilizado para la creación de diagramas en la fase de análisis para la documentación.
  - **Descripción:** Visual Paradigm Visual Paradigm es una herramienta de “Ingeniería de Software Asistida por Computación”. La aplicación ofrece multitud de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación.
- **Font Awesome:**
  - **Uso:** Se ha utilizado para la implementación de iconos vectoriales<sup>5</sup> en la aplicación.
  - **Descripción:** Es un framework de iconos vectoriales y estilos CSS. Este framework es utilizado para sustituir imágenes de iconos comunes por gráficos vectoriales convertidos en fuentes. Para ello utiliza una librería de más de 400 iconos transformadas en fuentes.

## **6. CONCLUSIONES**

A continuación, haré un resumen del trabajo realizado.

El primer paso fue decidir el tema del que iba a tratar el proyecto. Tras presentar al tutor la temática del mismo y recibir su conformidad, el siguiente paso fue definir los requisitos necesarios, qué tecnología usar, cómo estructurar el desarrollo de la aplicación y cuál sería su funcionalidad; según el tiempo que tenía para realizar dicho proyecto.

Después de decidir qué tipo de trabajo iba a realizar y con los recursos disponibles, comencé la fase de análisis con la que terminé de valorar los requisitos que iba a tener la aplicación. Así que, con la idea ya formada comencé el desarrollo de esta.

Cumpliendo los requisitos que me había propuesto en la fase de análisis, solucionando los problemas de la fase de desarrollo e implementando soluciones funcionales y eficientes.

Por último y para comprobar el correcto funcionamiento de todo el sistema, llevé a cabo las pruebas de la aplicación.

Puedo decir que estoy satisfecho con el resultado final y la funcionalidad de la aplicación, finalizándola en el tiempo previsto.

Cabe destacar que la realización del proyecto me ha servido para comprobar lo útiles que son los conocimientos adquiridos durante mis estudios del Ciclo Formativo de Grado Superior de Desarrollo de Aplicaciones Web y también me ha dado una visión más profesional del desarrollo de una aplicación web.

Ha sido un trabajo duro por el poco tiempo del que disponía para desarrollar la aplicación, pero eso me ha dado idea real del tiempo necesario para crear una aplicación profesional y preparada a las necesidades del mercado.

Quiero puntualizar que, por ser un prototipo, la funcionalidad de la aplicación web no es todo lo completa que hubiera deseado; muchas ampliaciones no han sido posibles dada la extensión del proyecto y el poco tiempo que he tenido para realizarlo, algunas de estas ideas las mencionaré a continuación.

A pesar de todo esto, estoy satisfecho con el trabajo realizado.

## 6.1 POSIBLES MEJORAS

A continuación, expongo una lista de mejoras que podrían implementarse sobre el prototipo:

- Apartado social:
  - Ofrecerle la posibilidad a los usuarios de la aplicación de poner una foto de perfil de usuario independientemente del formato de esta, poder cambiarla cuando quisiera.
  - Poder bloquear a otros usuarios el acceso a tus playlist, creando así una “blacklist”<sup>2</sup> de usuarios para tus listas.
  - Poder denunciar el uso incorrecto de otros usuarios.
  - Poder crear listas conjuntas permitiendo la colaboración de otros usuarios.
- Apartado funcional del reproductor:
  - Poder añadir otras fuentes, por ejemplo: Spotify, Mixcloud, Deezer, etc.
- Apartado de administración:
  - Crear un panel de administración que facilite al administrador el control sobre las playlist de los usuarios de la aplicación.

Pese a no haber podido implementar todas estas funciones, pienso que la aplicación es funcional y podría usarse diariamente de cara al público, siendo esta actualizada regularmente y añadiendo todas las funcionalidades mencionadas y aquellas que vayan surgiendo por necesidad o demanda de los usuarios.

## **7. BIBLIOGRAFÍA**

- Introducción a Laragon, Fernán García de Zúñiga , 7 de junio de 2019.  
<https://www.arsys.es/blog/programacion/introduccion-a-laragon/>
- Cliente servidor, Wikipedia. 7 de mayo de 2021  
<https://es.wikipedia.org/wiki/Cliente-servidor>
- Aprende lo qué es un diseño web y lo que hace un profesional de esta área. Redator Rock Content, 19 de junio de 2021.  
<https://rockcontent.com/es/blog/disenio-web/>
- Modelo cliente servidor. Andrés Schiaffarino en Tutoriales de Hosting, 12 de marzo de 2019.  
<https://blog.infranetworking.com/modelo-cliente-servidor/>
- HTTP API Guide. 8 de abril 2021  
<https://developers.soundcloud.com/docs/api/guide#playing>
- Login and Register Script In PHP PDO Whith MySQL. Hamid Shaikh, 15 de junio de 2019.  
<https://www.onlyxcodes.com/2019/04/login-and-register-script-in-php-pdo.html>
- Let users watch, find, and manage YouTube content. 8 de abril 2021  
[https://developers.google.com/youtube/?hl=en\\_US](https://developers.google.com/youtube/?hl=en_US)
- How to Create Image and File Upload in PHP with jQuery AJAX. Saquib Rizwan, 26 de marzo de 2019.  
<https://www.cloudways.com/blog/the-basics-of-file-upload-in-php/>
- CSS how to make scrollable list. 15 de marzo de 2014.  
<https://stackoverflow.com/questions/21998679/css-how-to-make-scrollable-list>

- YouTube Player API Reference for iframe Embeds. 14 de abril de 2021  
[https://developers.google.com/youtube/iframe\\_api\\_reference#GettingStarted](https://developers.google.com/youtube/iframe_api_reference#GettingStarted)
- Youtube. Wikipedia. 5 de mayo de 2021  
<https://es.wikipedia.org/wiki/YouTube>
- Widget API. 14 de abril de 2021  
<https://developers.soundcloud.com/docs/api/html5-widget#parameters>
- ¿Qué es Soundcloud? 5 de mayo de 2021  
<https://help.soundcloud.com/hc/es/articles/115003570488--Qu%C3%A9-es-SoundCloud->
- HTML5 Widget API. Alexander Kovalev, 22 de marzo de 2012.  
<https://developers.soundcloud.com/blog/html5-widget-api>
- Documentation. What is Laragon? Última actualización 1 de marzo de 2019.  
<https://laragon.org/docs/index.html>
- Introduction. Get started with Bootstrap. 21 de abril de 2021  
<https://getbootstrap.com/docs/4.1/getting-started/introduction/>
- jQuery API. 29 de abril de 2021  
<https://api.jquery.com/>
- MULTIPLATFORM SCALABLE VISION . 2014  
<https://www.statum.biz/statum/type1/59/arquitectura-de-apia>

## ANEXO – A

1. **API:** Interfaz de programación de aplicaciones.
2. **Blacklist:** En el caso de este documento, hablamos de “blacklist” haciendo referencia a listas de usuarios bloqueados por otros usuarios por no hacer un uso de la aplicación correcto o porque su contenido no es de su interés.
3. **DOM:** El DOM es la estructura de objetos que genera el navegador cuando se carga un documento y se puede alterar mediante Javascript para cambiar dinámicamente los contenidos y aspecto de la página.
4. **Fuente:** En este documento llamamos “Fuente” al servicio que nos ofrece las pistas de audio. Ejemplo de estas fuentes son: Youtube, Soundcloud.
5. **Icono vectorial:** Son imágenes comunes (JPG, PNG, etc.) que no están formadas por mapas de bits, sino que están estructuradas a partir de atributos matemáticos de forma, de posición, etc.; y por lo tanto se pueden cambiar de tamaño sin perder calidad.
6. **Layout:** La palabra layout hace referencia a la manera en que están distribuidos los elementos y las formas dentro del diseño de la aplicación. En general, el producto inicial de una página o sitio web es su layout, plantilla o diseño.
7. **Nick:** Del inglés nickname: significa “alias” o “seudónimo” para identificar a una persona de modo alternativo a su nombre propio.
8. **Pista:** Son las canciones o videos que se suben a la playlist.
9. **Playlist:** Lista de reproducción compuesta por pistas elegidas por el usuario.
10. **Script:** Es un programa sencillo, creado con la finalidad de facilitar tareas simples al programador.
11. **Soundcloud:** SoundCloud es la plataforma de audio abierta más grande del mundo, impulsada por una comunidad conectada de creadores, oyentes y comisarios en el pulso de lo nuevo, en la actualidad y en el futuro de la cultura.
12. **Youtube:** YouTube es un sitio web de origen estadounidense dedicado a compartir videos. Se le puede considerar un servicio de video en streaming, y permite a sus usuarios subir contenido de todo tipo sin límites.

## ANEXO – B

En este anexo agrego algunas imágenes de la aplicación web en funcionamiento, en ellas se puede observar el diseño final de la misma, además de algunas de sus funcionalidades en desempeño.

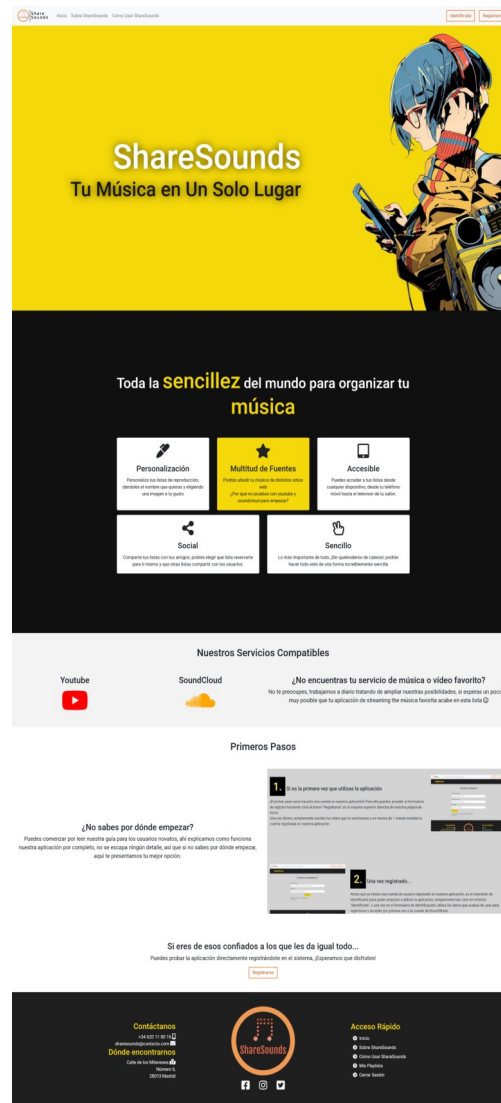


Figura 9: Página de inicio de la aplicación web.



The screenshot shows the 'Identificarse' (Login) page of the ShareSounds website. At the top, there is a navigation bar with the ShareSounds logo, links for 'Inicio', 'Sobre ShareSounds', and 'Cómo Usar ShareSounds', and buttons for 'Identificarse' and 'Registrarse'. The main heading is 'Identificarse'. Below it, the title 'Formulario de Identificación' is centered. The form contains two input fields: 'Usuario o Correo' with a placeholder 'Por favor escriba su usuario o correo electrónico' and 'Contraseña' with a placeholder 'Inserte su contraseña de acceso'. A yellow 'Identificarse' button is positioned to the right of the password field. Below the fields, there is a link: '¿No tienes una cuenta en ShareSounds? [Regístrate Aquí](#)'. The footer section is dark and contains contact information on the left, the ShareSounds logo in the center, and a 'Acceso Rápido' (Quick Access) menu on the right. The contact info includes a phone number, email, and address. The quick access menu lists: Inicio, Sobre ShareSounds, Cómo Usar ShareSounds, Mis Playlists, and Cerrar Sesión.

Figura 10: Formulario de identificación.

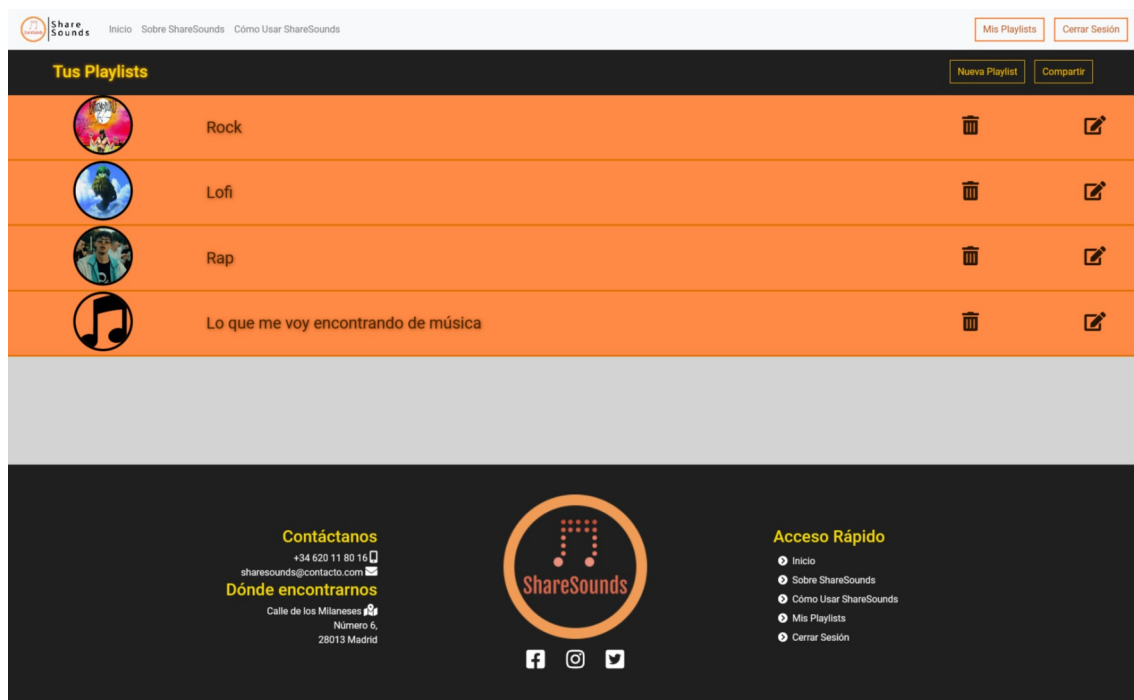



Figura 11: Lista de Playlist.


### Buscar Playlists

Nombre de la Playlist

Nombre del usuario creador



**Rock**  
Autor: Teker



**Lo que me voy encontrando de música**  
Autor: Teker

CerrarBuscar

Figura 12: Búsqueda de listas

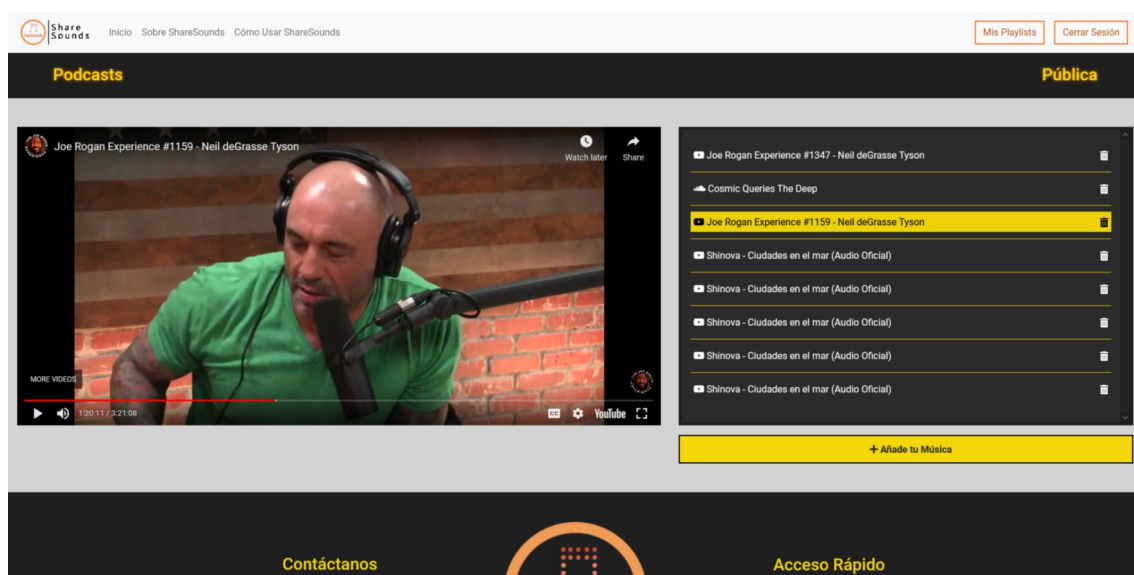


Figura 13: Reproductor funcionando.

**ANEXO – C**

Si se quiere ver la aplicación aquí inserto un enlace donde se puede visualizar la guía de la misma.

<http://sharesounds.test/es/howTo.php>