



INSTITUTO DE EDUCACIÓN SECUNDARIA CAMPANILLAS
TÉCNICO SUPERIOR DE DESARROLLO DE
APLICACIONES WEB

PROYECTO DAW

DESARROLLO DE UNA APLICACIÓN WEB DE GESTIÓN
MUSICAL

Realizado por
Juan Luis Almahano Leiva

Tutorizado por
Equipo Educativo 2DAW

DEPARTAMENTO DE FAMILIA PROFESIONAL DE
INFORMÁTICA Y COMUNICACIONES



MÁLAGA, 17 DE JUNIO DE 2021

Contenido

1. Presentación del proyecto.....	4
1.1 Introducción.....	4
1.2 Objetivos	4
1.3 Contexto	5
1.4 Planteamiento del problema	5
1.5 Principales tecnologías y herramientas empleadas	5
2. Descripción general	7
2.1 Perspectiva del producto	7
2.2 Funcionalidades del producto.....	7
2.3 Características del usuario.....	9
2.4 Restricciones, supuestos y dependencias	9
3. Requisitos específicos	9
3.1 Requerimientos funcionales	9
3.2 Definición de interfaces de usuario	17
3.2.1 Landing Page	18
3.2.2 Login.....	19
3.2.3 Register.....	20
3.2.4 Home/Inicio	21
3.2.5 Buscar	22
3.2.6 Lista de artistas.....	22
3.2.7 Añadir artista.....	23
3.2.8 Editar artista.....	23
3.2.9 Albums de artista.....	24
3.2.10 Detalle de un album.....	24
3.2.11 Añadir album	25
3.2.12 Editar album.....	25
3.2.13 Añadir canción	26
3.2.14 Editar canción	27
3.2.15 Playlist de canciones favoritas	27
3.2.16 Datos de perfil.....	28
3.3 Interface hardware	29
3.4 Interface software	29

3.5	Interface de comunicación	29
3.6	Estándares cumplidos.....	29
3.7	Seguridad.....	29
4.	Diseño de la base de datos.....	29
5.	Diseño	31
6.	Documentación del código de la aplicación.....	33
7.	Despliegue	33
7.1	Despliegue local	33
7.2	Despliegue remoto	33
8.	Conclusiones.....	34
9.	Bibliografía.....	35

1. Presentación del proyecto

1.1 Introducción

Este documento describe el proyecto realizado para el módulo Proyecto de Aplicaciones Web, del Ciclo Formativo de Grado Superior Desarrollo de Aplicaciones Web, en el Instituto de Educación Secundaria Campanillas. El proyecto realizado consiste en una aplicación web accesible desde cualquier navegador web, la cual se ha implementado usando las últimas tecnologías en el entorno de la programación web actual, y tiene por objetivo que se vean reflejados todos los conocimientos adquiridos durante la formación y realización de este grado.

1.2 Objetivos

El proyecto realizado se trata de una aplicación web que tiene como objetivo ser un entorno donde los aficionados a la música puedan escuchar canciones de sus artistas favoritos, similar a Spotify. Se pretende crear un espacio en el que los usuarios puedan buscar a sus artistas favoritos, ver información sobre ellos, ver información sobre sus albums y sobre las canciones incluidas en ellos, reproducir dichas canciones, y añadir cualquier canción a la lista de favoritos, así como eliminarla de dicha lista.

En cuanto a los usuarios que pueden participar dentro de esta aplicación encontraremos tres roles principales:

- Usuarios no registrados: únicamente pueden acceder a la Landing Page de la aplicación, donde se muestra información general de ésta y enlaces para la página de Login y la página de Registro.

- Usuarios registrados: pueden acceder a las mismas acciones que el usuario anterior, y además entrar en la propia aplicación, donde consultar los diferentes artistas, albums, canciones, reproducir canciones, añadir canciones a lista de favoritos, eliminar canciones de su lista de favoritos, hacer búsquedas de artistas. Por otro lado también pueden modificar sus datos de perfil.

- Usuario Administrador: puede realizar las mismas funciones que los dos usuarios anteriores, y además se encarga de añadir las canciones disponibles para escuchar, y la información para los diferentes artistas, albums y canciones. También puede tanto modificar como eliminar dicha información.

1.3 Contexto

El proyecto ha sido realizado con idea de hacer una aplicación web similar a Spotify, donde los aficionados a la música puedan escuchar canciones de sus artistas favoritos via streaming desde su navegador web, sin necesidad de instalar ninguna aplicación de escritorio.

Uno de los aspectos principales de esta aplicación es poder añadir canciones a la lista de favoritos personal de cada usuario.

1.4 Planteamiento del problema

El problema que se ha planteado es la creación de una aplicación web en donde los usuarios puedan escuchar música de forma dinámica, obtener información sobre los artistas, sus albums y sus canciones. Pudiendo añadir información descriptiva sobre cada artista, y sus respectivos albums, y añadir nuevas canciones.

Las funcionalidades, a grandes rasgos, serían las siguientes:

- Mostrar información general de los artistas.
- Mostrar información general de todos los albums.
- Mostrar información de los albums pertenecientes a un artista.
- Mostrar información general de las canciones de un album.
- Reproducir canciones.
- Añadir canciones a la lista de favoritos.
- Eliminar canciones de la lista de favoritos.
- Mostrar lista de canciones favoritas.
- Realizar búsquedas de artistas.
- Permitir al administrador añadir nuevos artistas.
- Permitir al administrador añadir nuevos albums.
- Permitir al administrador añadir nuevas canciones.
- Permitir al administrador subir imagen de album.
- Permitir al administrador subir imagen de artista.
- Permitir al administrador modificar artistas.
- Permitir al administrador modificar albums.
- Permitir al administrador eliminar artistas.
- Permitir al administrador eliminar albums.
- Permitir al administrador eliminar canciones.

1.5 Principales tecnologías y herramientas empleadas

En cuanto a las tecnologías y herramientas utilizadas para la realización de este proyecto, se han optado por el uso de tecnologías y herramientas que se están demandando mucho en el mercado actual de este sector.

Los lenguajes de programación que se han utilizado son TypeScript y JavaScript. Más específicamente, se ha decidido que para la parte backend de nuestra aplicación se implemente una API REST usando Node.js, que es un entorno de ejecución de JavaScript fuera del navegador, orientado a eventos asíncronos; Node.js está diseñado para crear aplicaciones fácilmente escalables. Para trabajar en Node.js se ha utilizado el framework Express.js; este framework permite la creación de un servidor web de forma cómoda y sencilla, está respaldado por una gran comunidad de desarrolladores, lo que garantiza la continua actualización de sus características principales; Express.js es un framework minimalista que permite crear aplicaciones web y APIs rápidamente, ayuda a los usuarios a configurar rutas para enviar y recibir solicitudes entre la vista Front y la Base de Datos actuando como un servidor HTTP y, así mismo, es compatible con muchos paquetes npm, por lo tanto tiene muchísimas posibilidades a la hora de trabajar con él.

Para la parte frontend de la aplicación se ha decidido utilizar el framework Angular en su versión 11. Angular es uno de los frameworks más demandados del sector actual, usa el lenguaje TypeScript y se utiliza para desarrollar aplicaciones SPA (single page application), de código abierto y mantenido por Google; tiene una gran comunidad de desarrolladores que lo soportan y hacen que trabajar con él sea muy cómodo. Utilizando Angular CLI (herramientas de línea de comandos que nos facilita el fabricante) se permite comenzar la creación de un proyecto de forma muy rápida con pocos pasos, y la modularidad de sus componentes permite crear elementos muy reutilizables lo que permite una programación más dinámica y escalable; por todo esto y por muchas más cualidades que tiene este framework, ha sido elegido para la creación de este proyecto.

Se ha utilizado también HTML junto a etiquetas enriquecidas proporcionadas por Angular, para dar formas a las vistas de la aplicación.

Para dar estilos a los diferentes elementos se ha utilizado CSS, que nos permite trabajar de forma mucho más dinámica y eficiente con nuestros estilos; se han implementado diferentes animaciones y efectos conseguidos gracias a las múltiples posibilidades que nos da CSS3.

Como apoyo para el diseño, se ha utilizado el framework Bootstrap para la creación de algunos elementos, como son los formularios, la barra de navegación y las alertas.

Como Base de datos se ha optado por MongoDB, que es libre y muy potente, y es además la base de datos que más he utilizado durante el periodo de Formación en Centros de Trabajo; es una base de datos no relacional que nos permite trabajar de forma muy intuitiva, creando elementos, modificándolos y eliminándolos sin ningún tipo de restricción.

Para el control de versiones se ha utilizado el software Git manejado mediante la consola de comandos Git Bash. El repositorio remoto está alojado en GitHub.

Como entorno de desarrollo, se ha utilizado Visual Studio Code, ya que es un entorno gratuito con muchísimas extensiones que ayudan en diferentes tareas, permitiendo

escribir código de una forma cómoda y veloz. Visual Studio Code nos permite además abrir una terminal dentro de él, que nos ayudará a crear los elementos y movernos dentro del proyecto de una forma muy rápida. Además es uno de los mas recomendados a la hora de trabajar con JavaScript o TypeScript ya que ofrece integración total con estos lenguajes, y autocompletado

Para el despliegue remoto se han utilizado tres servicios diferentes, uno para cada parte de la aplicación:

Para la parte del backend (API REST), se ha utilizado la plataforma Heroku: es una plataforma como servicio de computación en la nube, que soporta distintos lenguajes de programación y es propiedad de Salesforce.com, por lo tanto es muy estable, tiene opciones gratuitas y una gran cantidad de herramientas que permiten que el despliegue de las diferentes partes de la aplicación resulte muy intuitiva.

Para el almacenamiento remoto de la base de datos se ha decidido usar MongoDB Atlas, que es una plataforma para desarrolladores que permite la creación de clusters para el almacenamiento de bases de datos MongoDB con una interfaz elegante y cómoda, teniendo opciones de almacenamiento gratuito de gran capacidad y estable.

Finalmente, para desplegar la parte de frontend se ha usado GitHub Pages, la opción de GitHub para mostrar páginas estáticas del lado del servidor, ya que los ficheros de salida que genera Angular son simplemente HTML, CSS, JS, y no necesitan de ninguna solución que requiera servidor con interprete de lenguaje o base de datos.

2. Descripción general

2.1 Perspectiva del producto

La aplicación web desarrollada pretende ser un medio en el que los aficionados a la música puedan realizar búsquedas sobre artistas, visualizar su informacion sobre sus discos, y escuchar sus canciones, creando así un entorno completo donde poder gestionar todos estos elementos.

2.2 Funcionalidades del producto

Las funcionalidades que conforman la aplicación se pueden distinguir según el tipo de usuario que se encuentre en ella, y son:

Usuarios no registrados:

- Autenticación: cuando el usuario no registrado intente hacer una acción que dependa de su autenticación se le informará de que debe registrarse o loguearse para acceder a ella.
- Registro en la aplicación, mediante un formulario.

- Visionado de la Landing Page y de la diversa información que se encuentra en ella.

Usuarios registrados:

- Cerrar sesión: se podrá cerrar sesión mediante un botón definido en la barra de navegación, que una vez pulsado redirigirá a la Landing Page.
- Editar Perfil de usuario: cada usuario podrá visualizar o modificar sus datos personales.
- Buscar artista: mediante el boton buscar de la barra lateral se pueden buscar artistas por nombre o descripción.
- Mostrar todos los artistas: se mostraran todos los artistas a razon de cuatro por página, y se usaran botones de paginación para salta a la siguiente página.
- Entrar en un artista: se mostraran todos los albums de ese artista determinado.
- Mostrar todos los albums disponibles en la aplicación
- Entrar en un album: se mostraran todas las canciones de ese album
- Añadir a favoritos: dentro de la página de mostrar canciones de un album, se mostrará un boton para añadir esa canción a la lista de canciones favoritas de cada usuario.
- Ver lista de favoritos: se podrá ver la lista de canciones favoritas mediante un botón definido en la barra lateral.
- Eliminar canción de lista de favoritos: desde la lista de canciones favoritas se muestra un botón en cada canción para eliminarla de esta lista.
- Reproducir canción: cada canción tendrá un botón para permitir su reproducción por parte del reproductor de audio.

Usuario Administrador:

- Añadir un artista nuevo a la base de datos: mediante el cumplimiento de un formulario se podrán añadir nuevos artistas a la aplicación.
- Subir imagen de artista al servidor: mediante un boton de formulario se podrá seleccionar una imagen para asociarla a un artista determinado.
- Editar artista: se podrá editar la información de los artistas ya almacenados en la base de datos.
- Eliminar artista de la base de datos: se eliminan además los album de ese artista y sus canciones.
- Añadir un album nuevo a la base de datos: mediante el cumplimiento de un formulario se podrá añadir un nuevo album para un artista determinado.
- Subir imagen de album al servidor: mediante un boton de formulario se podrá seleccionar una imagen para asociarla a un album de un artista determinado.
- Editar album: se podrá editar la información de los albums ya almacenados en la base de datos.
- Eliminar album de la base de datos: se eliminan también sus canciones.
- Añadir canción: mediante un boton de formulario se podrán añadir canciones a un album determinado.

2.3 Características del usuario

Se pueden definir dos tipos de usuarios principales: Los no registrados y los registrados; dentro de los registrados podemos diferenciar dos subtipos: normales y administrador.

- Usuarios NO registrados: solamente tendrán acceso a la landing page de la aplicación y no a ninguna de sus funcionalidades.
- Usuarios Registrados: son los que tienen acceso a las funcionalidades más interesantes de la aplicación, diferenciando entre usuarios registrados generales y usuarios administradores:
 - Usuarios Normales: pueden usar la aplicación, pero sin tener acceso a la parte de administración del contenido.
 - Usuarios Administradores: pueden acceder además a las diferentes opciones para añadir y gestionar el contenido de la aplicación.

2.4 Restricciones, supuestos y dependencias

El origen natural de esta aplicación web hace que se requiera de un ordenador, teléfono o tablet con navegador, y de una conexión a internet para poder tener acceso a ella.

La aplicación ha sido desarrollada utilizando el navegador Google Chrome, y aunque se han ido haciendo pruebas en otros navegadores como Firefox, algunos aspectos estéticos de la aplicación pueden variar según el navegador en donde se esté visionando. La aplicación ha sido desarrollada bajo el sistema operativo Windows.

3. Requisitos específicos

3.1 Requerimientos funcionales

Mediante una serie de tablas se describen las diversas funciones que ofrece esta aplicación web, dispuesta según el tipo de usuario:

Usuario no registrado:

Apartado	Descripción
Título	Autenticación

Propósito	Acceder a apartados concretos de la app
Entrada	Email y contraseña
Proceso	Se comprueba la existencia del usuario
Salida	Permite el acceso a la aplicación o devuelve un error si no es válido.

Apartado	Descripción
Título	Registro
Propósito	Registrar al usuario dentro de la app
Entrada	Name, Surname, Email y contraseña
Proceso	Registrar al usuario en la app
Salida	Si los datos son correctos se registra al usuario en la app. Si no, se devuelve un error

Usuario registrado:

Apartado	Descripción
Título	Buscar artistas
Propósito	Busca artistas dentro de la app
Entrada	Nombre o descripción de artista
Proceso	Busca al artista dentro de la base de datos
Salida	Si encuentra algún artista que coincida se muestra en la vista de detalle

Apartado	Descripción
Título	Consultar datos de un usuario

Propósito	Ver información específica de un usuario concreto
Entrada	
Proceso	Se muestra la información referente a un usuario específico extraída de la base de datos mediante la id del usuario
Salida	Información referente al usuario: nombre, apellidos, email, imagen.

Apartado	Descripción
Título	Mostrar artistas
Propósito	Mostrar todos los artistas dentro de la app
Entrada	
Proceso	Busca todos los artistas dentro de la base de datos
Salida	Muestra todos los artistas de la base de datos

Apartado	Descripción
Título	Mostrar albums
Propósito	Mostrar todos los albums de un artista dentro de la app
Entrada	
Proceso	Busca todos los albums de un artista dentro de la base de datos
Salida	Muestra todos los albums de un artista

Apartado	Descripción
----------	-------------

Título	Mostrar canciones
Propósito	Mostrar todas las canciones de un album dentro de la app
Entrada	
Proceso	Busca todas las canciones de un album dentro de la base de datos
Salida	Muestra todas las canciones de un album

Apartado	Descripción
Título	Añadir a favoritos una canción
Propósito	Añadir a favoritos una canción concreta
Entrada	
Proceso	Se añade una relación entre el usuario y esa canción elegida en la base de datos
Salida	Muestra esa canción en el playlist del usuario

Apartado	Descripción
Título	Eliminar una canción de favoritos
Propósito	El usuario elimina una canción de su lista de favoritos
Entrada	
Proceso	Se elimina la relación entre el usuario y la canción en la base de datos
Salida	En el playlist del usuario ya no aparece la canción

Apartado	Descripción
----------	-------------

Título	Reproducir canción
Propósito	Escuchar la canción elegida en el reproductor de audio
Entrada	
Proceso	Recupera el nombre del fichero canción de la base de datos
Salida	Se reproduce la canción

Usuario administrador:

Apartado	Descripción
Título	Añadir nuevo artista
Propósito	Añadir nuevo artista a la aplicación
Entrada	Información relevante del artista
Proceso	Se añade la información del artista dentro de la base de datos
Salida	Se confirma que el artista se ha añadido correctamente mediante un mensaje, y si hay algún problema también se informará de ello

Apartado	Descripción
Título	Añadir nuevo album
Propósito	Añadir nuevo album a la aplicación
Entrada	Información relevante del album
Proceso	Se añade la información del album dentro de la base de datos
Salida	Se confirma que el album se ha añadido correctamente mediante un mensaje, y si hay algún problema también se informará de ello

Apartado	Descripción
Título	Añadir nueva canción
Propósito	Añadir nueva canción a la aplicación
Entrada	Información relevante de la canción
Proceso	Se añade la información de la canción dentro de la base de datos
Salida	Se confirma que la canción se ha añadido correctamente mediante un mensaje, y si hay algún problema también se informará de ello

Apartado	Descripción
Título	Añadir imagen de artista
Propósito	Añadir imagen de artista a la aplicación
Entrada	Imagen a añadir
Proceso	Se añade el nombre de la imagen dentro de la base de datos, y el fichero en el servidor
Salida	Se confirma que la imagen se ha añadido correctamente mediante un mensaje, y si hay algún problema también se informará de ello

Apartado	Descripción
Título	Añadir imagen de album
Propósito	Añadir imagen de album a la aplicación
Entrada	Imagen a añadir
Proceso	Se añade el nombre de la imagen dentro de la base de datos, y el fichero en el servidor

Salida	Se confirma que la imagen se ha añadido correctamente mediante un mensaje, y si hay algún problema también se informará de ello
--------	---------------------------------------------------------------------------------------------------------------------------------

Apartado	Descripción
Título	Añadir fichero de canción
Propósito	Añadir fichero de canción a la aplicación
Entrada	Fichero a añadir
Proceso	Se añade el nombre de la canción dentro de la base de datos, y el fichero en el servidor
Salida	Se confirma que el fichero se ha añadido correctamente mediante un mensaje, y si hay algún problema también se informará de ello

Apartado	Descripción
Título	Editar artista
Propósito	Editar artista de la aplicación
Entrada	Información relevante del artista
Proceso	Se modifica la información del artista dentro de la base de datos
Salida	Se confirma que el artista se ha modificado correctamente mediante un mensaje, y si hay algún problema también se informará de ello

Apartado	Descripción
Título	Editar album
Propósito	Editar album de la aplicación

Entrada	Información relevante del album
Proceso	Se modifica la información del album dentro de la base de datos
Salida	Se confirma que el album se ha modificado correctamente mediante un mensaje, y si hay algún problema también se informará de ello

Apartado	Descripción
Título	Editar canción
Propósito	Editar canción de la aplicación
Entrada	Información relevante de la canción
Proceso	Se modifica la información del artista dentro de la base de datos
Salida	Se confirma que el artista se ha modificado correctamente mediante un mensaje, y si hay algún problema también se informará de ello

Apartado	Descripción
Título	Eliminar artista
Propósito	Eliminar artista de la aplicación
Entrada	
Proceso	Se elige el artista que se quiere eliminar y se borrarán sus datos de la base de datos
Salida	Se confirma que el artista se ha eliminado correctamente, y este ya no se mostrará en la aplicación, ni sus albums ni canciones

Apartado	Descripción
Título	Eliminar album

Propósito	Eliminar album de la aplicación
Entrada	
Proceso	Se elige el album que se quiere eliminar y se borrarán sus datos de la base de datos
Salida	Se confirma que el album se ha eliminado correctamente, y este ya no se mostrará en la aplicación, ni sus canciones

Apartado	Descripción
Título	Eliminar canción
Propósito	Eliminar canción de la aplicación
Entrada	
Proceso	Se elige la canción que se quiere eliminar y se borrarán sus datos de la base de datos
Salida	Se confirma que la canción se ha eliminado correctamente, y esta ya no se mostrará en la aplicación

3.2 Definición de interfaces de usuario

Se trata de una aplicación web en el que todas las funciones se prestan a través de las diferentes páginas que la componen (en Angular se usa el concepto de componente, una clase con una plantilla asociada, en vez de página), a las cuales se podrán acceder a través de un navegador web. Se podrán encontrar diferentes botones para realizar ciertas funciones relevantes en la aplicación. La toma de datos de los usuarios en diferentes apartados se hará mediante formularios. Se mostrarán mensajes al realizar algunas de las acciones para confirmar que se han realizado con éxito o, por el contrario, han fracasado.

Se ha intentado realizar una interfaz llamativa e intuitiva en la que el usuario pueda tener una buena experiencia navegando en ella. La landing page tiene un diseño responsive que se adapta a ordenador, tablet y móvil.

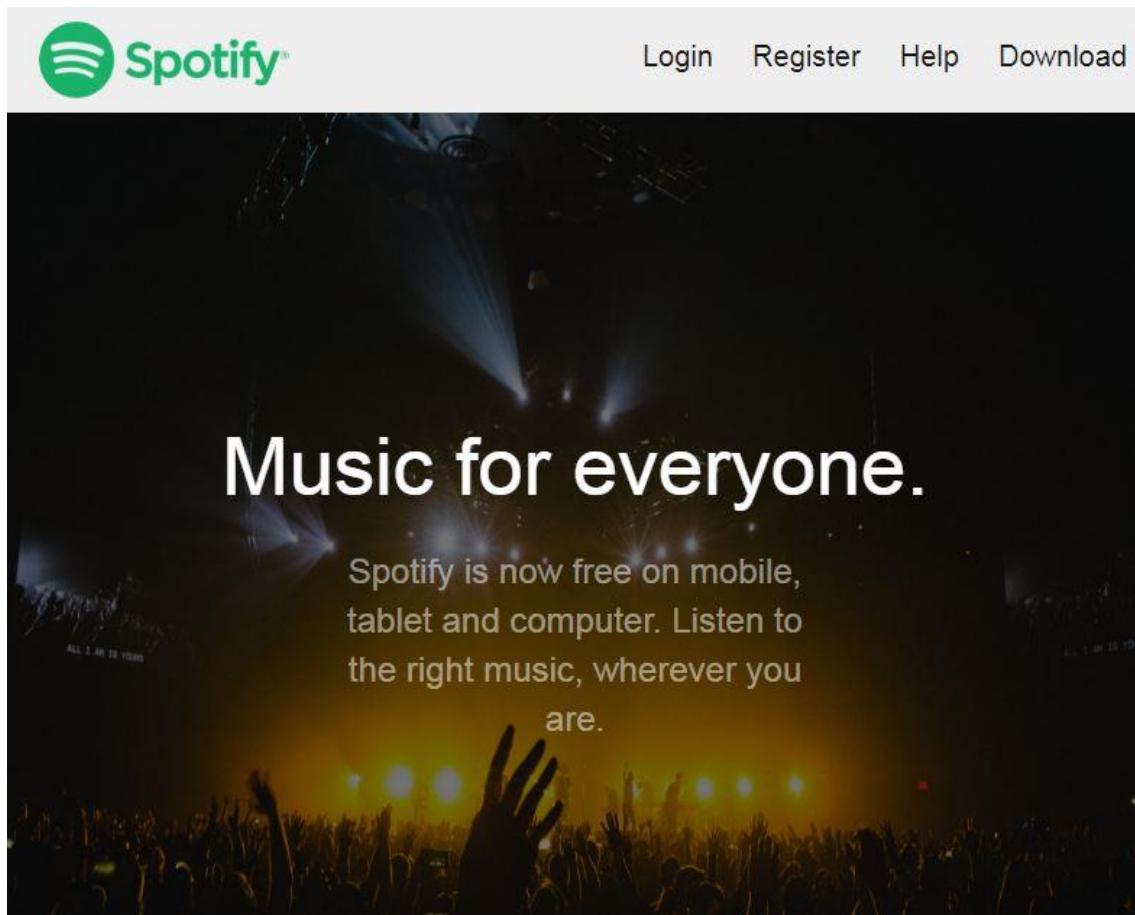
La aplicación se compone de 18 páginas/componentes diferentes, aunque algunos

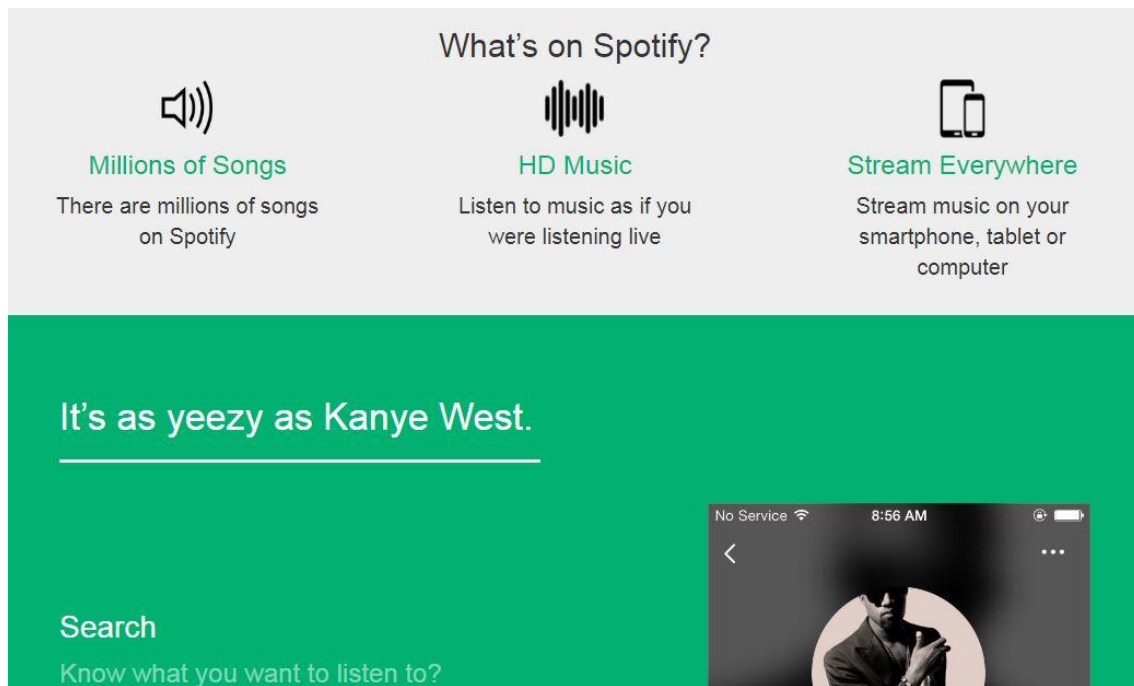
componentes carecen de vista/plantilla html asociada, ya que reutilizan la plantilla de otro componente.

A través de toda la aplicación será accesible una barra de navegación con el usuario logueado, una barra lateral en la que aparecen diferentes opciones que hagan más cómoda la interacción del usuario en la aplicación, y en la parte inferior un reproductor siempre fijo.

3.2.1 Landing Page

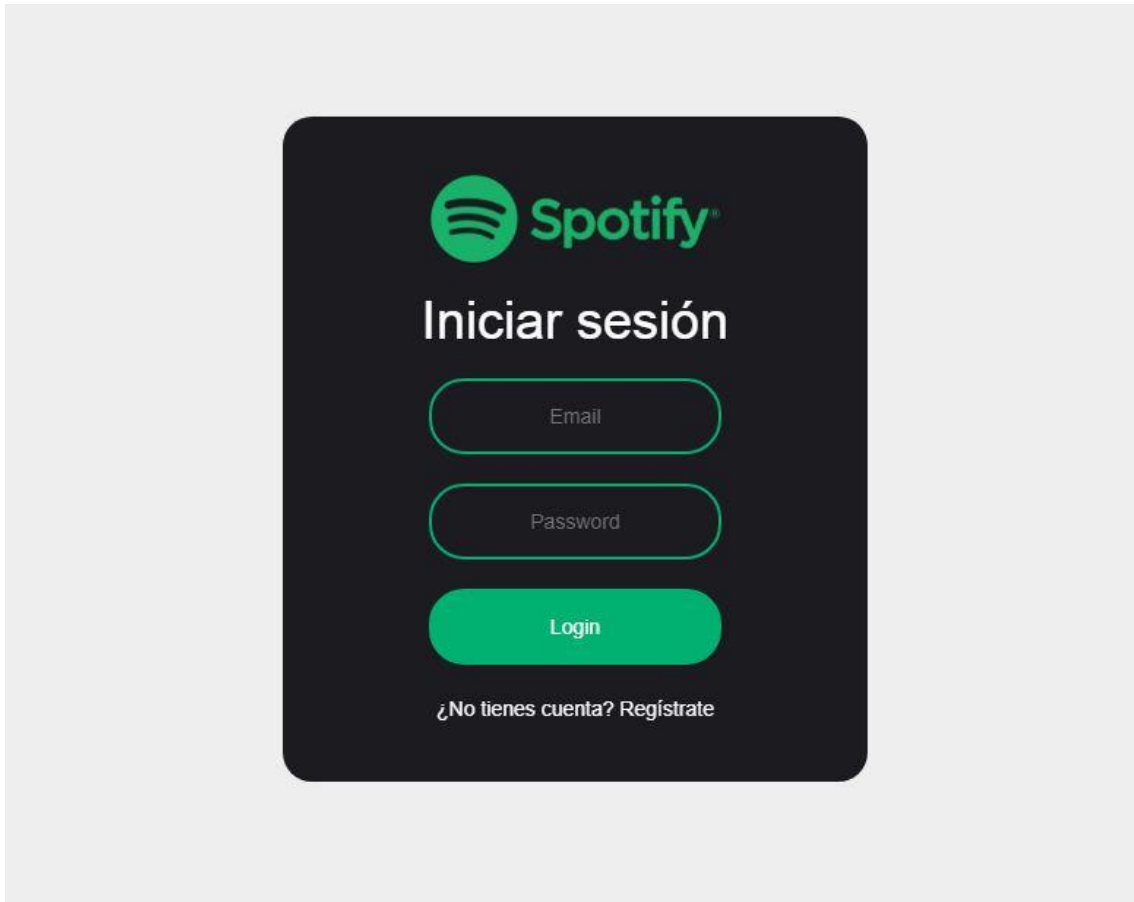
Es la pantalla por defecto de la aplicación, desde la cual se puede acceder a la pantalla de login y a la pantalla de registro de usuarios.





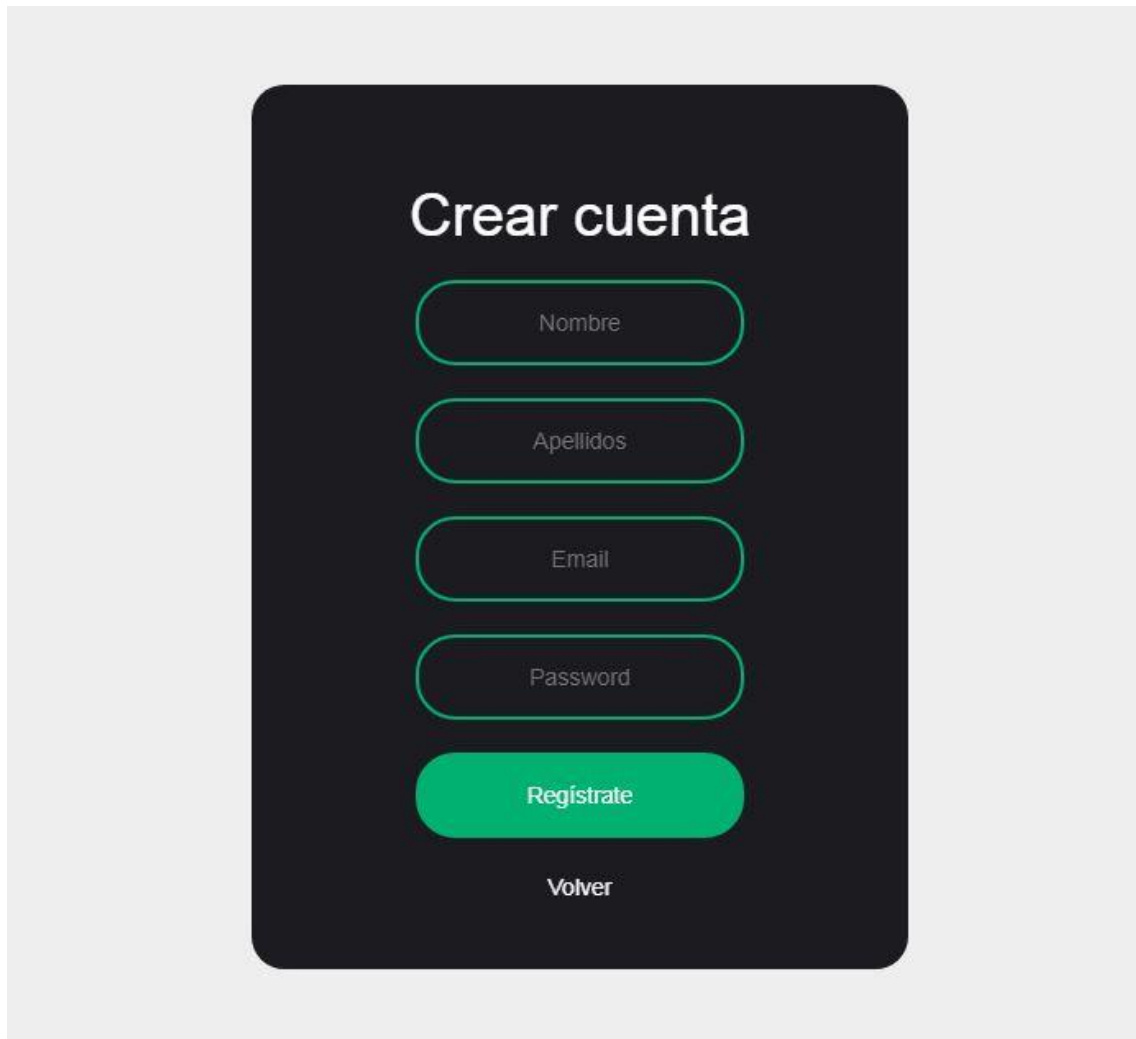
3.2.2 Login

Formulario de login de la aplicación.



3.2.3 Register

Formulario de registro de usuarios.



The image shows a mobile application screen for creating a new account. The screen has a dark blue background with rounded corners. At the top, the title 'Crear cuenta' is displayed in white. Below the title, there are four input fields, each with a light blue border and a light blue placeholder text: 'Nombre', 'Apellidos', 'Email', and 'Password'. Below these fields is a prominent red button with the text 'Regístrate' in white. At the bottom of the form, there is a link labeled 'Volver' in white text.

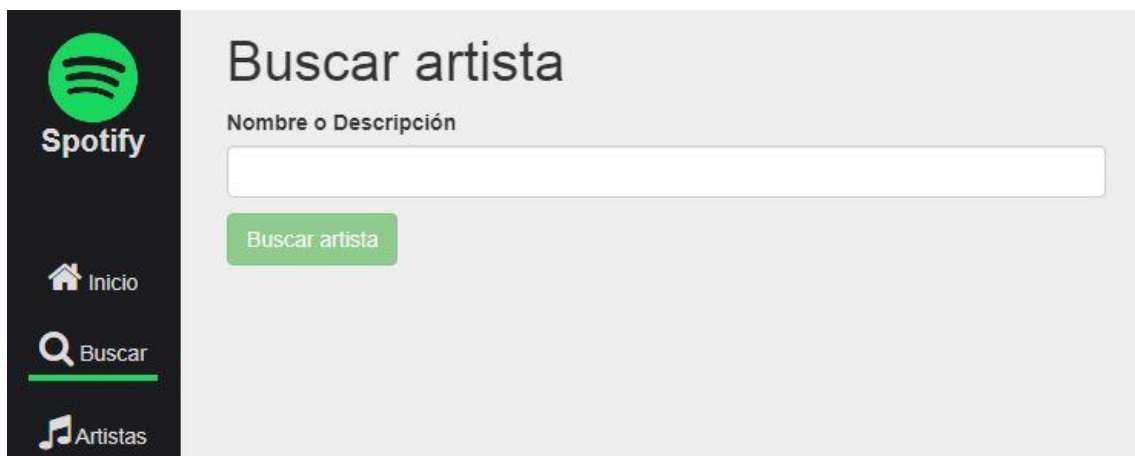
3.2.4 Home/Inicio

Pantalla por defecto una vez logueado el usuario para usar la aplicación. Se puede apreciar una barra lateral vertical con las diferentes opciones; una barra de navegación en la parte superior con opciones para el usuario; una parte central que muestra información personal y que irá cambiando de forma reactiva; y una parte inferior con el reproductor de audio y la carátula del album.



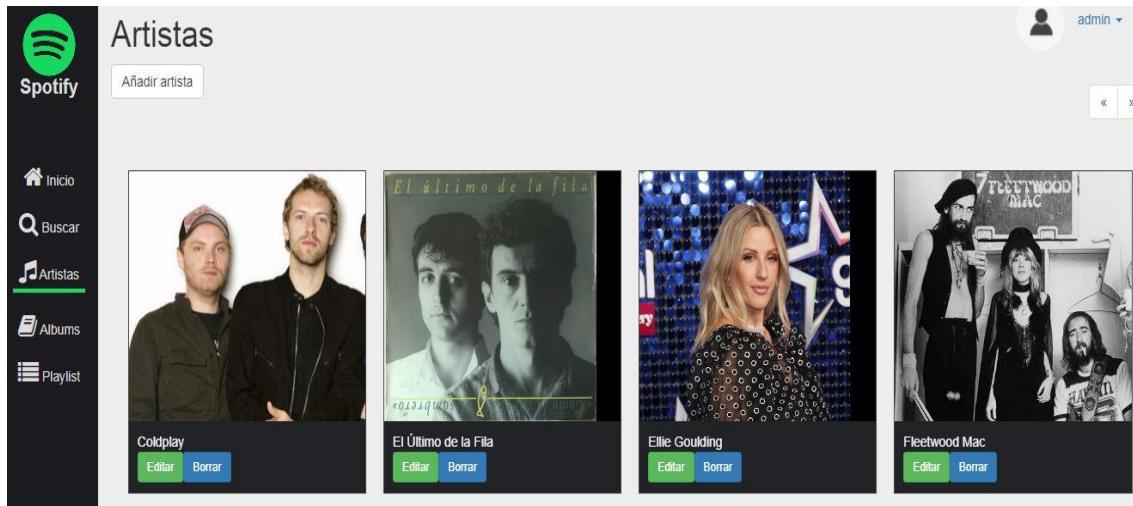
3.2.5 Buscar

Incluye un campo para buscar artistas por nombre o por descripción.



3.2.6 Lista de artistas

Muestra la lista completa de artistas, de forma paginada.



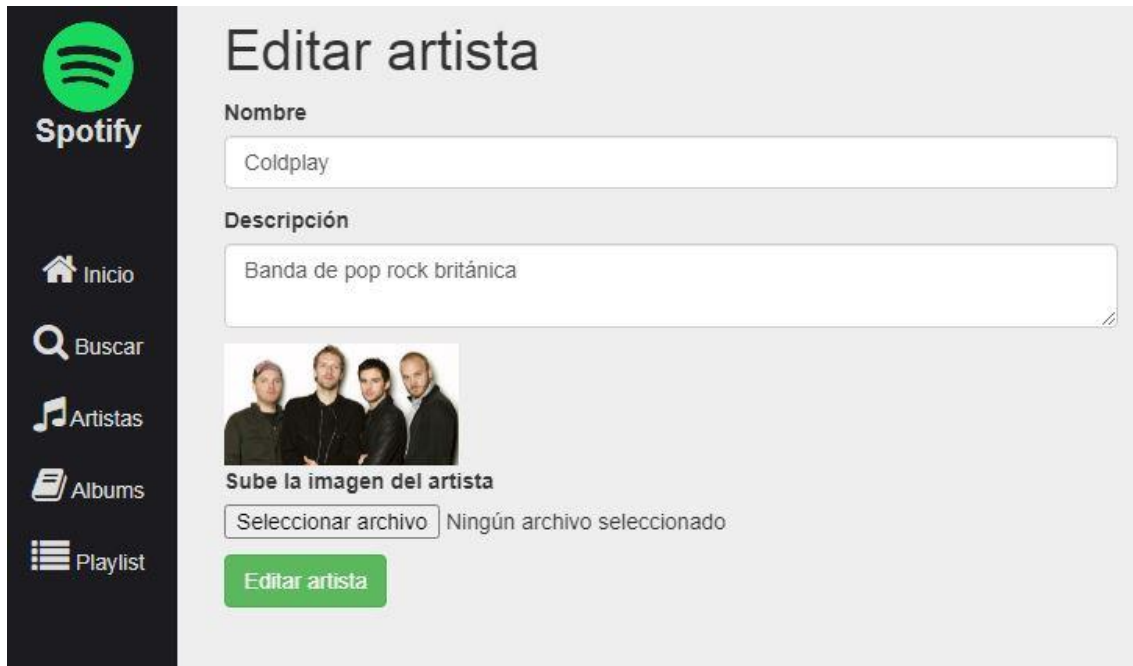
3.2.7 Añadir artista

Formulario para crear artista.

A screenshot of a web application interface for creating a new artist. The header shows the Spotify logo and the title 'Crear nuevo artista'. Below the header are two input fields: 'Nombre' and 'Descripción'. Below the 'Descripción' field is a green button labeled 'Crear nuevo artista'. The left sidebar is identical to the previous screenshot, showing navigation links and the Spotify logo.

3.2.8 Editar artista

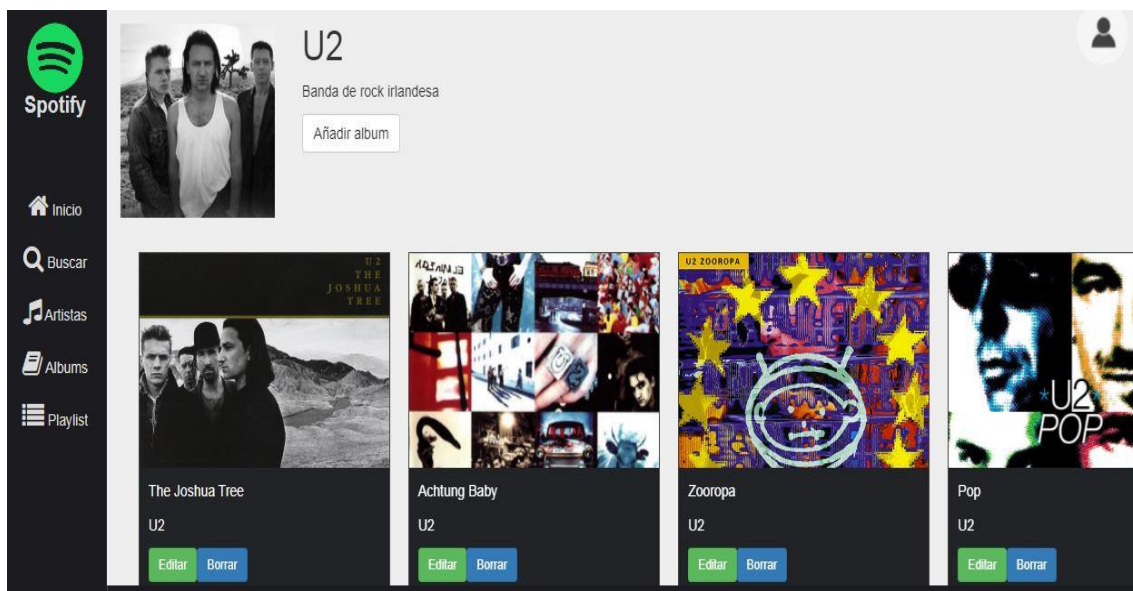
Formulario para editar artista. Permite seleccionar un archivo de imagen para asociarla al artista.



The screenshot shows the Spotify 'Editar artista' (Edit artist) interface. On the left is a dark sidebar with the Spotify logo and navigation links: Inicio, Buscar, Artistas, Albums, and Playlist. The main area has a light gray background. At the top, the title 'Editar artista' is displayed. Below it, there are two text input fields: 'Nombre' (Name) with 'Coldplay' entered, and 'Descripción' (Description) with 'Banda de pop rock británica' entered. Below the description is a placeholder image of four men. Underneath the image is the text 'Sube la imagen del artista' (Upload the artist's image), followed by a button 'Seleccionar archivo' (Select file) and the text 'Ningún archivo seleccionado' (No file selected). At the bottom of the form is a green button labeled 'Editar artista' (Edit artist).

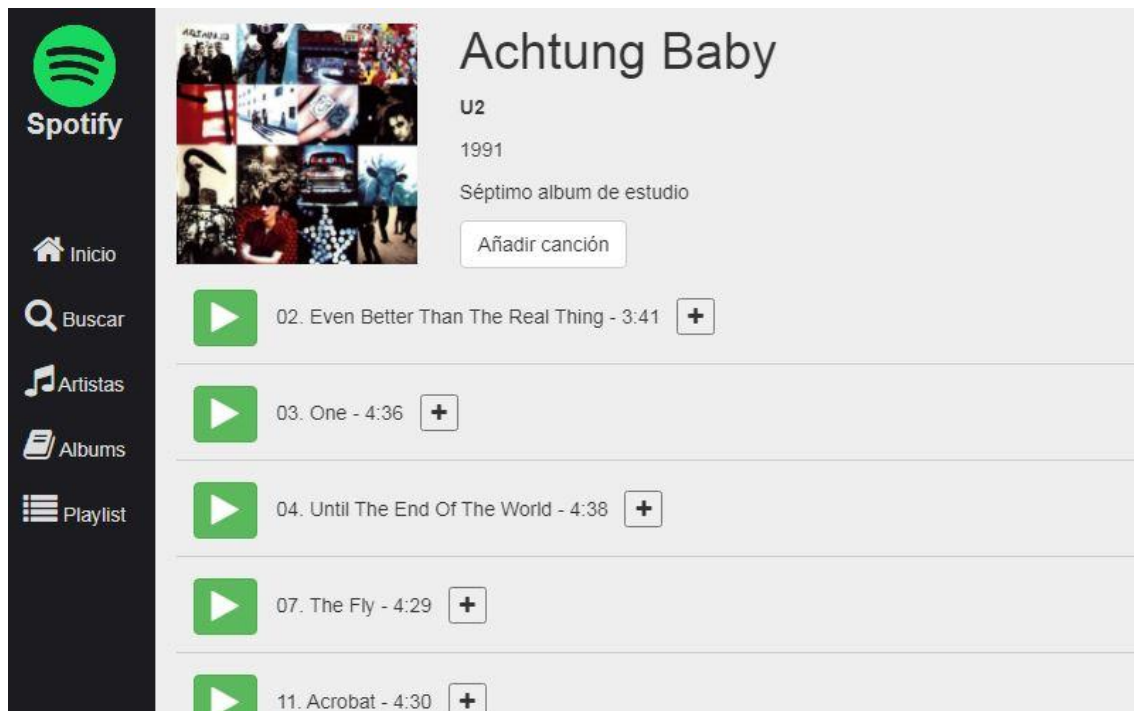
3.2.9 Albums de artista

Muestra todos los albums de un artista en particular, junto con la información de ese artista.



3.2.10 Detalle de un album

Muestra la información de un album junto con todas sus canciones.



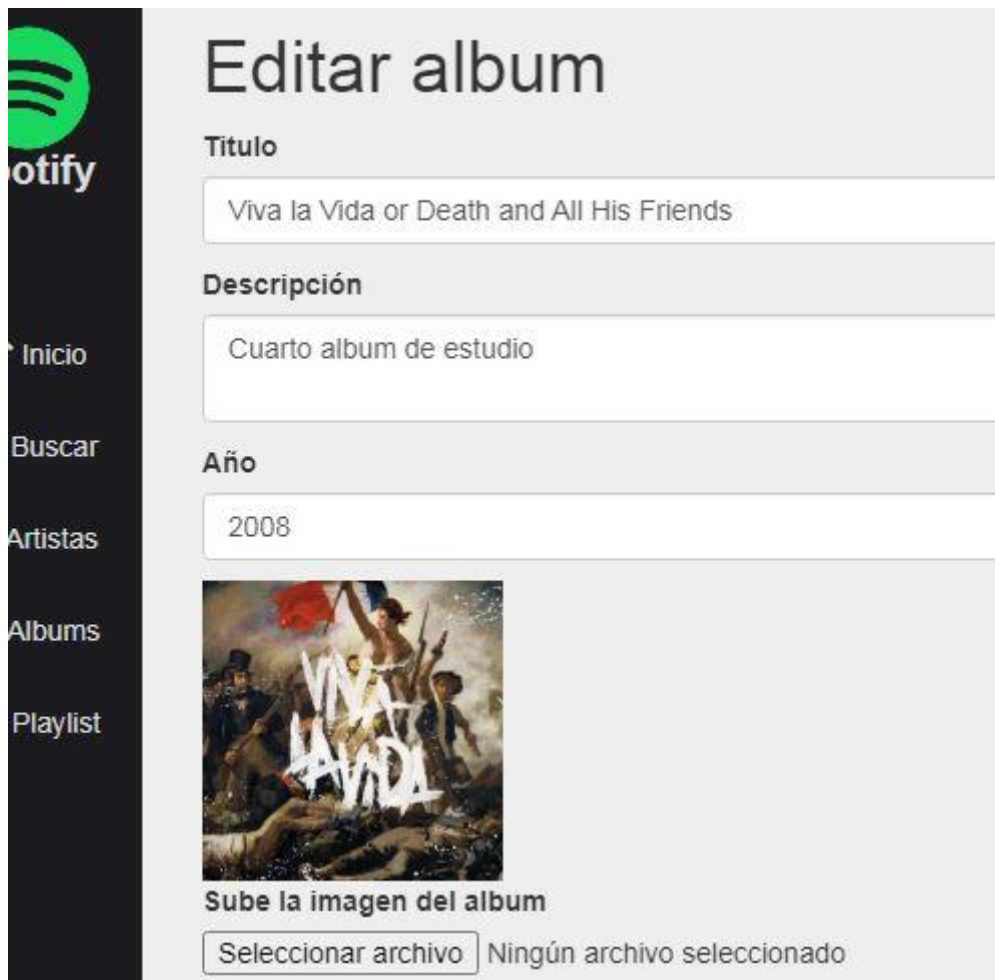
3.2.11 Añadir album

Formulario para crear nuevo album.

A screenshot of the Spotify web interface showing the 'Crear nuevo album' form. The left sidebar is identical to the previous image. The main content area has the title 'Crear nuevo album'. Below it are three input fields: 'Título', 'Descripción', and 'Año'. The 'Año' field contains the value '2021'. At the bottom of the form is a green button labeled 'Crear nuevo album'.

3.2.12 Editar album

Formulario para editar album. Permite seleccionar una imagen para asociarla con ese album.



Editar album

Título


Viva la Vida or Death and All His Friends

Descripción

Cuarto album de estudio

Año

2008

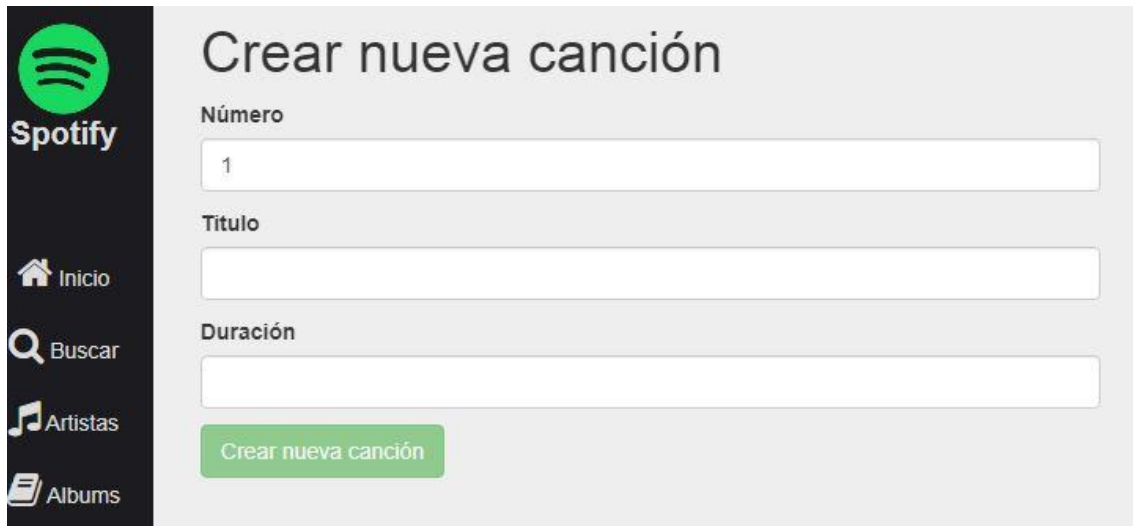


Sube la imagen del album

Seleccionar archivo Ningún archivo seleccionado

3.2.13 Añadir canción

Formulario para crear canción y añadirla a un album.



The screenshot shows the Spotify web interface for creating a new song. On the left is a dark sidebar with the Spotify logo and navigation links: Inicio, Buscar, Artistas, and Albums. The main content area is light gray and titled 'Crear nueva canción'. It contains three input fields: 'Número' (containing '1'), 'Título' (empty), and 'Duración' (empty). Below these fields is a green button labeled 'Crear nueva canción'.

3.2.14 Editar canción

Formulario para editar datos de una canción y seleccionar el fichero correspondiente.



The screenshot shows the Spotify web interface for editing a song. The sidebar is identical to the previous form. The main content area is titled 'Editar canción'. It contains three input fields: 'Número' (containing '02'), 'Título' (containing 'Even Better Than The Real Thing'), and 'Duración' (containing '3:41'). Below these fields is a media player control bar showing a play button, a progress bar at '0:00 / 3:41', a volume icon, and a settings icon. Below the media player is a section titled 'Sube el fichero de audio' with a button labeled 'Seleccionar archivo' and the text 'Ningún archivo seleccionado'. At the bottom is a green button labeled 'Editar canción'.

3.2.15 Playlist de canciones favoritas

Muestra la lista de canciones favoritas del usuario actualmente logueado.



3.2.16 Datos de perfil

Muestra los datos del usuario.

A screenshot of the 'Actualizar mis datos' (Update my data) form in the same Spotify-like interface. The sidebar is identical. The main content area has a light gray background with the title 'Actualizar mis datos' at the top. Below the title are four form sections: 1. 'Nombre:' with a text input field containing 'admin'. 2. 'Apellidos:' with a text input field containing 'admin'. 3. 'Correo electrónico:' with a text input field containing 'admin@admin.com'. 4. 'Sube tu foto:' which includes a small square placeholder icon with a person silhouette, a 'Seleccionar archivo' button, and the text 'Ningún archivo seleccionado'. At the bottom of the form is a blue button labeled 'Actualizar mis datos'.

3.3 Interface hardware

Puesto que se trata de una aplicación web, su visualización será posible a través de cualquier sistema operativo.

3.4 Interface software

La aplicación funciona sobre cualquier móvil, tablet u ordenador que disponga de un navegador web y de una conexión a internet.

3.5 Interface de comunicación

Las comunicaciones dentro de la aplicación se realizarán mediante el uso del protocolo HTTP mediante la conexión TCP/IP.

3.6 Estándares cumplidos

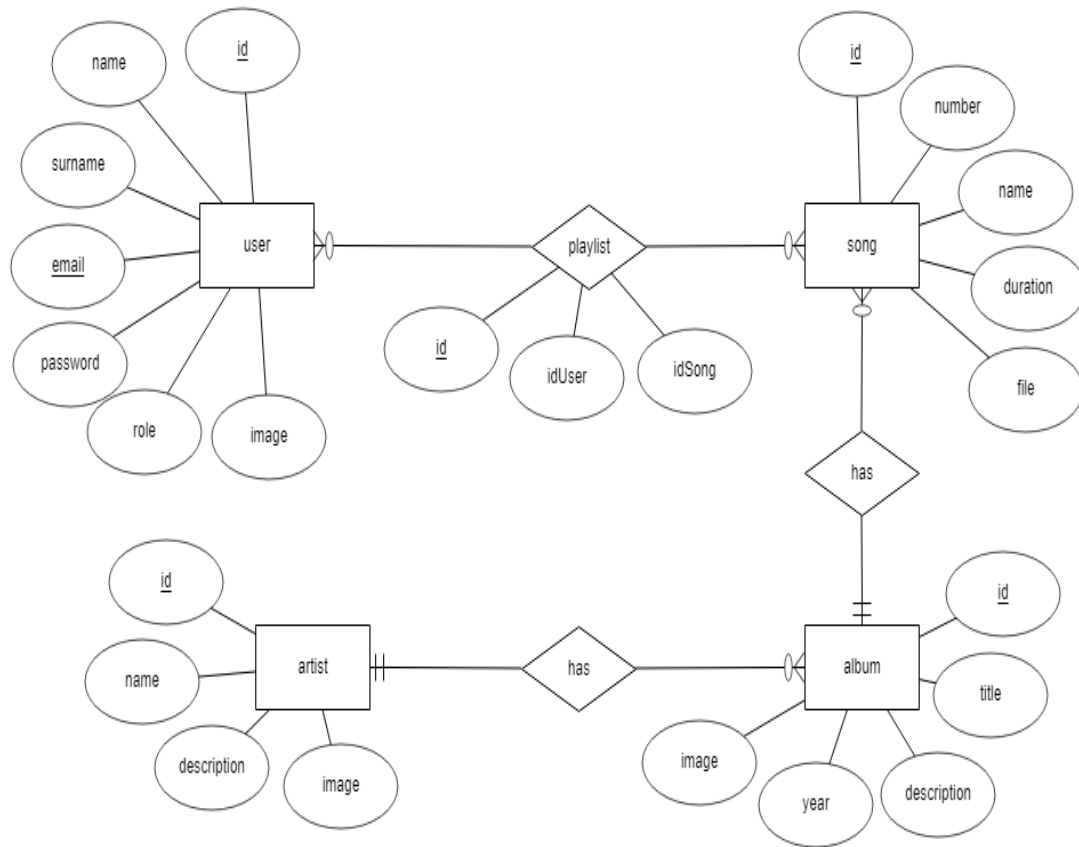
A la hora de realizar la aplicación web se han seguido los estándares generales de todo sitio web con un acceso seguro, creando diferentes perfiles que se controlan mediante el sistema de autenticación a nivel de backend, y que mediante la obtención de un token se permite el acceso o no a ciertas zonas de la aplicación (como hacer peticiones al API REST). A nivel de frontend se ha implementado guards de Angular, los cuales protegen las rutas de usuarios no identificados.

3.7 Seguridad

El tema de la seguridad se ha intentado hacer de la forma más óptima posible, controlando los permisos que poseen los usuarios dentro de la aplicación tanto en la parte del front como en el back: para poder interactuar con la aplicación será necesario el registro y login del usuario vía email y contraseña. Se hace control de sesiones (mediante el localStorage del navegador), generación de tokens y nunca se pasa a la parte frontal información sensible del usuario, como puede ser su contraseña.

4. Diseño de la base de datos

El diagrama entidad-relación de este proyecto sería el siguiente:



Para la creación de la base de datos se ha optado MongoDB, un sistema de bases de datos NoSQL, orientado a documentos (en lugar de guardar los datos en tablas, guarda estructuras de datos BSON, similar a JSON) con un esquema dinámico, haciendo que la integración de los datos en la aplicación sea más fácil y rápida.

Para ayudar en la tarea de realizar los modelos de esquemas de las colecciones, se ha utilizado la librería Mongoose, un ODM (Object Data Modeling) que nos facilita la tarea de interactuar con MongoDB:

A continuación se muestran los esquemas de las cinco colecciones:

```
var UserSchema = Schema({
  name: String,
  surname: String,
  email: String,
  password: String,
  role: String,
  image: String
});
```

```
var ArtistSchema = Schema({
  name: String,
  description: String,
  image: String
});
```

```
var AlbumSchema = Schema({
  title: String,
  description: String,
  year: Number,
  image: String,
  artist: {type: Schema.ObjectId, ref: 'Artist'}
});
```

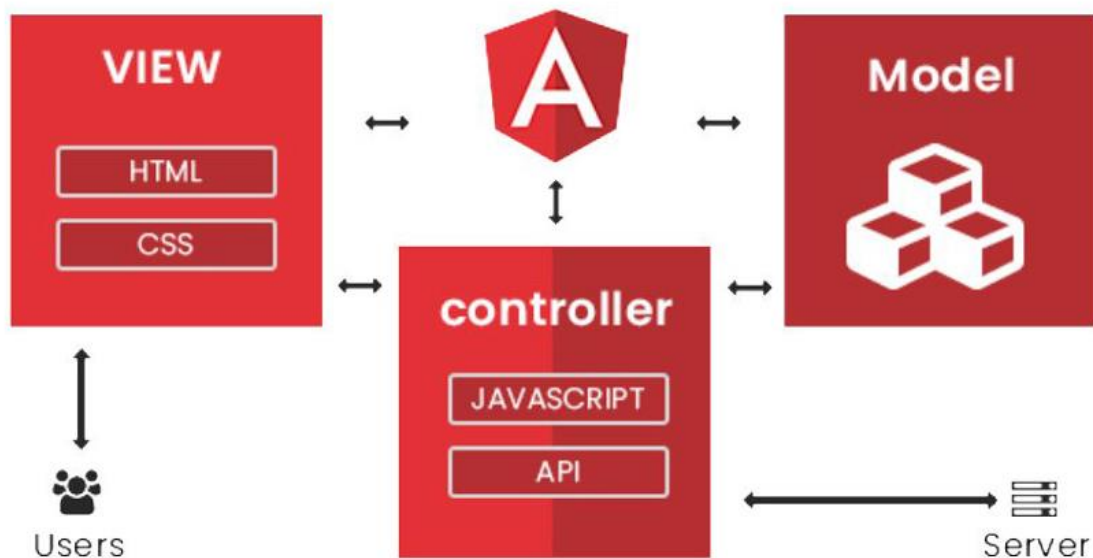
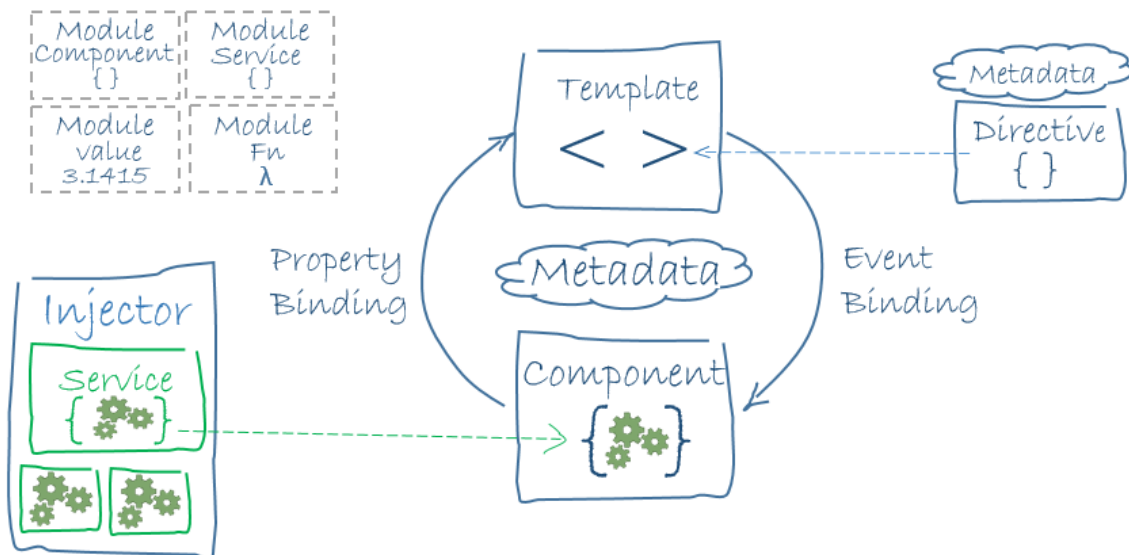
```
var SongSchema = Schema({
  number: String,
  name: String,
  duration: String,
  file: String,
  album: {type: Schema.ObjectId, ref: 'Album'}
});
```

```
var PlaylistSchema = Schema({
  user: { type: Schema.ObjectId, ref: 'User'},
  song: { type: Schema.ObjectId, ref: 'Song'}
});
```

5. Diseño

Para el diseño del backend se ha usado el patrón MVC (Model View Controller) con la distinción de que al ser un API REST no se usan vistas, en vez de ello se devuelven objetos en formato JSON.

El diseño del frontend también está basado en el patrón MVC, pero en este caso se encuentra integrado en el propio framework y en la manera de trabajar de Angular.



Aquí se puede ver un ejemplo de cómo se puede entender el patrón MVC desde el framework de Angular.

La vista representa la parte visual de la aplicación: es la parte donde el usuario va a interactuar, también conocida como la interfaz de usuario, y se compone por la información que se envía al cliente y los mecanismos de interacción con él.

El modelo contiene una representación de los datos que maneja el sistema, su lógica y mecanismos.

El controlador actúa como intermediario entre el Modelo y la vista, gestiona el flujo de la información entre ellos y hará las transformaciones necesarias para que se adapten

a las necesidades de cada uno.

6. Documentación del código de la aplicación

Para la documentación del código en la parte de backend se ha usado la herramienta JSDOC, la cual permite generar documentación automática de manera agíl, siempre que el código esté bien comentado, para lo cual se ha usado la extension de Visual Studio Code Document This.

7. Despliegue

7.1 Despliegue local

Para el despliegue local de nuestra aplicación solo es necesario tener instalado Node.js en nuestro equipo: Node.js es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome.

Para poder trabajar localmente con nuestro proyecto Angular basta con ir a la terminal y utilizar el comando “npm i” para descargarse todos los node_modules (npm folders) necesarios para el funcionamiento de la aplicación, y seguidamente usar el comando “ng serve” para compilar la aplicación: esto logrará que de forma automática, y según hemos configurado, se despliegue la aplicación localmente. Una vez compilado, basta con ir a nuestro a la dirección <http://localhost:4000> y podremos ver nuestra aplicación Angular desplegada localmente.

Para desplegar la parte de Node.js es necesario también ejecutar el comando “npm i”, para así instalar todas las dependencias de node_modules; hemos configurado su puerto de salida para que aparezca en la dirección <http://localhost:3000>. Al utilizar el comando “node nombreDeLaApp.js”, se compilará de forma automática y se mostrará en el puerto configurado; en este caso, se ha configurado el archivo package.json para que con solo usar el comando “npm run start” o “npm run dev” se realice la compilación.

Para la base de datos se necesita tener instalado MongoDB y lanzar el servicio mongod.exe.

7.2 Despliegue remoto

Para el despliegue remoto he optado por tres soluciones diferentes, una pr cada parte de la aplicación:

-MongoDB Atlas: plataforma que me permite crear clusters para alojar bases de datos para MongoDB. Las cuentas gratuitas permiten hasta 512MB. El proceso de creación no es complicado, pero hay que tener en cuenta varios aspectos: permitir el acceso desde cualquier IP, activar los permisos de lectura y escritura, y añadir un nuevo usuario y clave.

Una vez creado el cluster, se nos proporciona una cadena de conexión para que nuestras aplicaciones tengan acceso a él.

-Heroku: plataforma como servicio de computación en la nube que soporta aplicaciones desarrolladas en Node.js. Para desplegar se necesita git, heroku cli (herramientas de línea de comandos) y una cuenta (la gratuita permite hasta 5 aplicaciones con un límite de 500MB).

Antes de realizar el despliegue, debemos revisar ciertos aspectos:

- en el código fuente hay que usar la variable de entorno `process.env.PORT` para definir el puerto de escucha del servidor.
- tener fichero `Procfile`, el cual indica a heroku cual es el script de `package.json` que debe usar
- indicar en `package.json` la versión del motor de node.js
- tener el fichero `.gitignore` y evitar subir la carpeta `node_modules`

Para desplegar la aplicación debe tener algún commit realizado, y a continuación:

- loguearnos en la consola (heroku login)
- vincular repositorio local con el remoto (heroku create)
- desplegar (git push heroku main)

Una vez terminado el proceso nos proporciona una url para acceder, que debemos usar desde el frontend.

-Github Pages: para la parte de frontend, Angular genera ficheros de salida con el comando (`ng build --prod`). Al ser simplemente ficheros html, css y js se pueden alojar en un repositorio de github activando en la opción github pages, la cual indica que se podrá acceder como un servidor de páginas estáticas.

8. Conclusiones

La realización de este proyecto ha sido un logro alcanzado tras haber podido combinar en él todos los conocimientos adquiridos durante estos dos años de formación no solo en el instituto sino en la formación DUAL y en la FCT. Aunque en algunos momentos me he sentido desorientado, he conseguido resolver problemas, ayudándome sobre todo de la lectura de documentación y de las diferentes webs donde se puede encontrar el apoyo de la comunidad de desarrolladores.

9. Bibliografia

- El gran libro de angular: Miquel Boada Oriols
- Libro: Creando API con Node.js, express y MongoDB : Gustavo Morales
- Web oficial de angular: <https://angular.io/>
- Web oficial de bootstrap: <https://getbootstrap.com>

- Canal de youtube Fazt:
<https://www.youtube.com/channel/UCX9NJ471o7Wie1DQe94RVlg>
- Canal de youtube Dominicode:
https://www.youtube.com/channel/UC3QuZuJr2_EOUak8bWUd74A
- Canal de youtube Carlos Azaustre:
<https://www.youtube.com/c/CarlosAzaustre/playlists>
- Canal de youtube Informática DP:
https://www.youtube.com/channel/UCLa5WaffYWAaJY09_l863yw