PATRONES DE SOFWTARE
-
PRÁCTICA FINAL

Hub de Aplicaciones

Daniel Hernández Arcos – 02758969G
Juan Gómez Hernández – 50338597S
UAH 2020-2021 | G. Ing. Informática | Patrones de Software

Contenido

1. INTRODUCCIÓN

Se propone el desarrollo de una simulación en local de un Hub de aplicaciones que consistirá en una aplicación que administre tres aplicaciones.

- Consistirá en un usuario general con el que se accede al Hub (tendrá que ser un usuario de Gmail para su posible vinculación con la aplicación de Youtube).
- Este usuario está compartido por las tres aplicaciones administradas, de forma que, se pueda acceder a ellas con un usuario común a ambas aplicaciones, siendo el usuario de acceso al Hub.
- Estas tres aplicaciones independientes son las siguientes:
  - Reproductor de video online (Reproductor Youtube): consta de un buscador de videos, así como su selección y visualización. Utiliza la aplicación de YouTube para realizar estas acciones, sin embargo, la visualización se realizará dentro del Hub.
    Adicionalmente se podrán descargar vídeos y reproducirlos de forma local en otra pestaña del Hub.
  - Reproductor de video offline: Es un reproductor de vídeo local, se podrán descargar archivos .mp4 para su visualización.
  - Reproductor de música: Es un reproductor de música local, se podrá descargar música desde el reproductor de video, convirtiendo desde mp4 a mp3 para su reproducción en local.
- La aplicación se ejecutará en un ordenador personal. Será necesario el acceso a una red en el caso de querer acceder al reproductor online, sin embargo, no es necesario el acceso a internet para la reproducción de vídeos y música ya descargada, por lo que se podrán reproducir de manera offline.

Los elementos a tener en cuenta por la aplicación son los siguiente:

Usuarios

- En esta versión de la aplicación se contempla la utilización del sistema por dos tipos de usuarios, un administrador del Hub e invitados.
- Ambos usuarios tienen funcionalidades comunes, las cuales consisten en:
  - Podrá realizar búsquedas en youtube por una palabra clave o categoría, así como su posterior selección.
  - Visualizar vídeos y reproducir música tanto en youtube como aquellos archivos ya descargados.
  - Filtrado de vídeos y música descargada por título o id.
- El usuario administrador podrá realizar las siguientes funciones:
  - El inicio de sesión se realizará mediante un formulario donde se solicitará un usuario *"admin@gmail.com*" y clave "*xxxx*".
  - Podrá descargar vídeos y música desde el acceso a youtube.
  - Podrá eliminar y editar propiedades de los vídeos y música ya descargados.
- Los usuarios invitados:

    - Tendrán un nombre de usuario y una contraseña que no se corresponde a un correo de Gmail. Para su acceso a Youtube utilizarán el Gmail general de la aplicación
    - Tendrán permisos limitados, pudiendo únicamente acceder al reproductor de youtube.

Reproductor de video online

Para esto, el usuario administrador deberá de dar de alta, en la consola de developer de google la Youtube Data API v3. Debido a la complejidad de esto para un usuario normal, se utiliza un correo gmail general.

Esta tiene un límite de envío de datos diario, debido a esto, se utiliza una variable que limita el número de videos solicitados, siendo un máximo de 10.

A partir de aquí:

- Una vez se accede al reproductor de video online se verifican las credenciales con la API correspondiente a YouTube y posteriormente se procurarán las siguientes funcionalidades:
    - Un buscador por tema el cual ordenará por dating o por rate, dependiendo de la opción marcada por el usuario
    - El usuario elegirá el video correspondiente para su reproducción. Una vez realizada esta acción se accederá a una ventana de reproducción donde se podrá visualizar el video seleccionado.
    - En la ventana general habrá dos funcionalidades adicionales accesibles por el administrador, que serán: "*descargar como música" y "descargar como vídeo"*, que descargaron, una vez pulsados, el archivo correspondiente .mp4 y guardado posteriormente en el directorio local correspondiente accesible por la aplicación.
    - En el caso de "*descargar como música"* este archivo será convertido directamente a .mp3 una vez descargado, y guardado posteriormente en el directorio local correspondiente.
- Este reproductor es el propio ofrecido por Youtube, por lo que constará de todas las funcionalidades provistas por el propio reproductor oficial de Youtube.

Reproductor de video offline

- El acceso a este reproductor de video es automático, debido a la propia verificación del Hub.
- Una vez se accede al reproductor se mostrará una lista con los videos descargados.
- Habrá un botón por el cual se podrá filtrar el listado de videos por título o id.Una vez iniciada la búsqueda se mostrará un listado de aquellas que tengan un índice de coincidencia suficiente
- Estos videos mostrados se pueden reproducir con su selección, tras la cual se abrirá el reproductor correspondiente.

- En el listado de videos habrá un botón, para eliminar los vídeos que no se desean, los cuales serán eliminados del sistema (con opción de revertir en caso de equivocación).

Reproductor de música

- El acceso a este reproductor de música es automático, debido a la propia verificación del Hub.
- Una vez se accede al reproductor se mostrará una lista con los videos descargados.
- Habrá un botón por el cual se podrá filtrar el listado de videos por título o id. Una vez iniciada la búsqueda se mostrará un listado de aquellas que tengan un índice de coincidencia suficiente
- Estos videos mostrados se pueden reproducir con su selección, tras la cual se abrirá el reproductor correspondiente.
- En el listado de videos habrá un botón, para eliminar la música que no se desea, que será eliminada del sistema (con opción de revertir en caso de equivocación).
- Adicionalmente, dentro del propio reproductor de música se encontrará la opción de pasar canción hacia atrás y hacia adelante según la música descargada (estará con la opción cíclica por defecto, de forma que, aunque se acabe la lista, se continúe desde el extremo opuesto).
    - En esta opción se mostrará el título tanto de la canción actual, como de la posterior y la anterior.

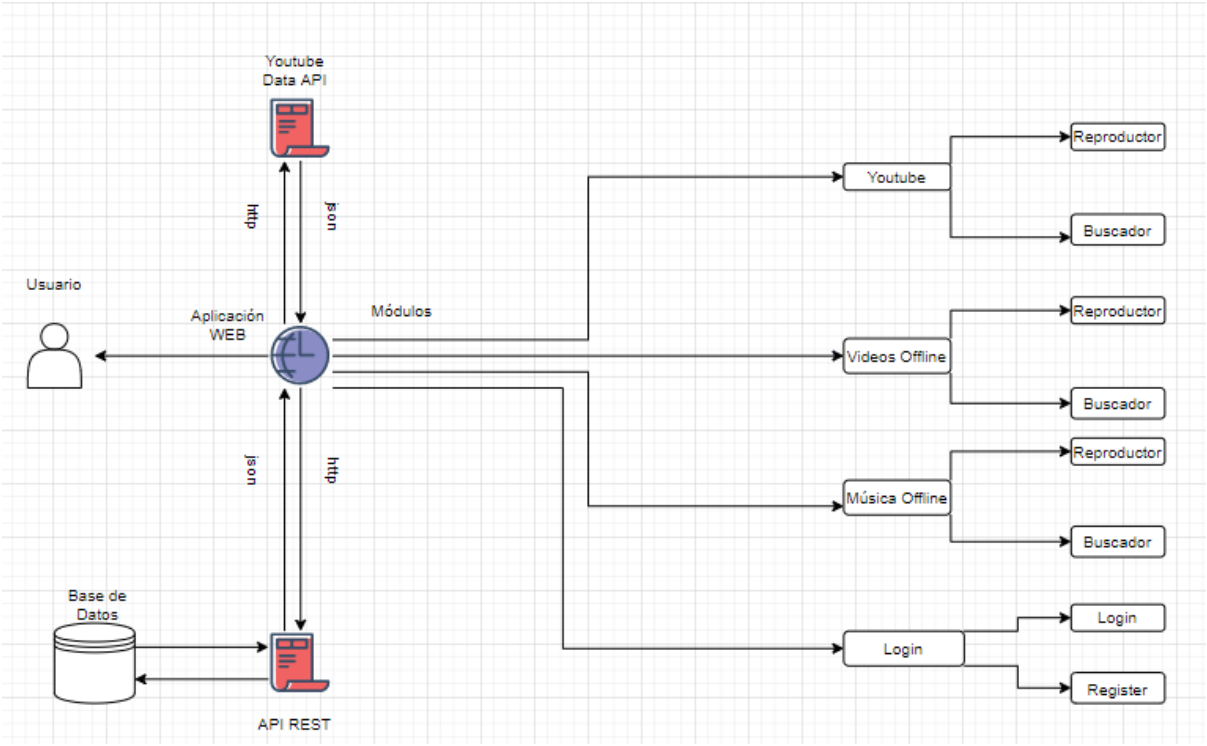
2. DESARROLLO DE LA APLICACIÓN

1.1. Aplicación Web.

Esto se desarrollará con el el framework de desarrollo Angular mediante la creación de los respectivos módulos: youtube, videos offline y música offline.
En estos módulos se desarrollan los patrones de diseño en archivos de typescript, los cuales se aplican en el respectivo componente, con la funcionalidad en un archivo .ts y su posterior visualización mediante la creación de archivos .css y .html ligados al funcionamiento del componente.
Para su correcto funcionamiento, además, se ha utilizado el localStorage para poder guardar elementos permanentes de la aplicación y el sessionStorage para guardar elementos con un ciclo de vida ligado a la sesión creada.
Esta aplicación se comunica mediante peticiones http a dos API REST, una creada por nosotros (aquella ligada con la base de datos) y la propia API de Youtube para poder acceder a los videos

- Estas comunicaciones se realizan mediante el patrón Observable, de forma que se pueda desarrollar de forma asíncrona, y se notifique al cliente una vez haya respondido la API correspondiente

Visualmente constará de un dashboard para la movilidad entre las diferentes apps del Hub. Este únicamente se mostrará en caso de estar logueado.
Permitirá el registro y login de usuarios (esta autenticación se realizará mediante el aprovisionamiento de un token generado en el back).

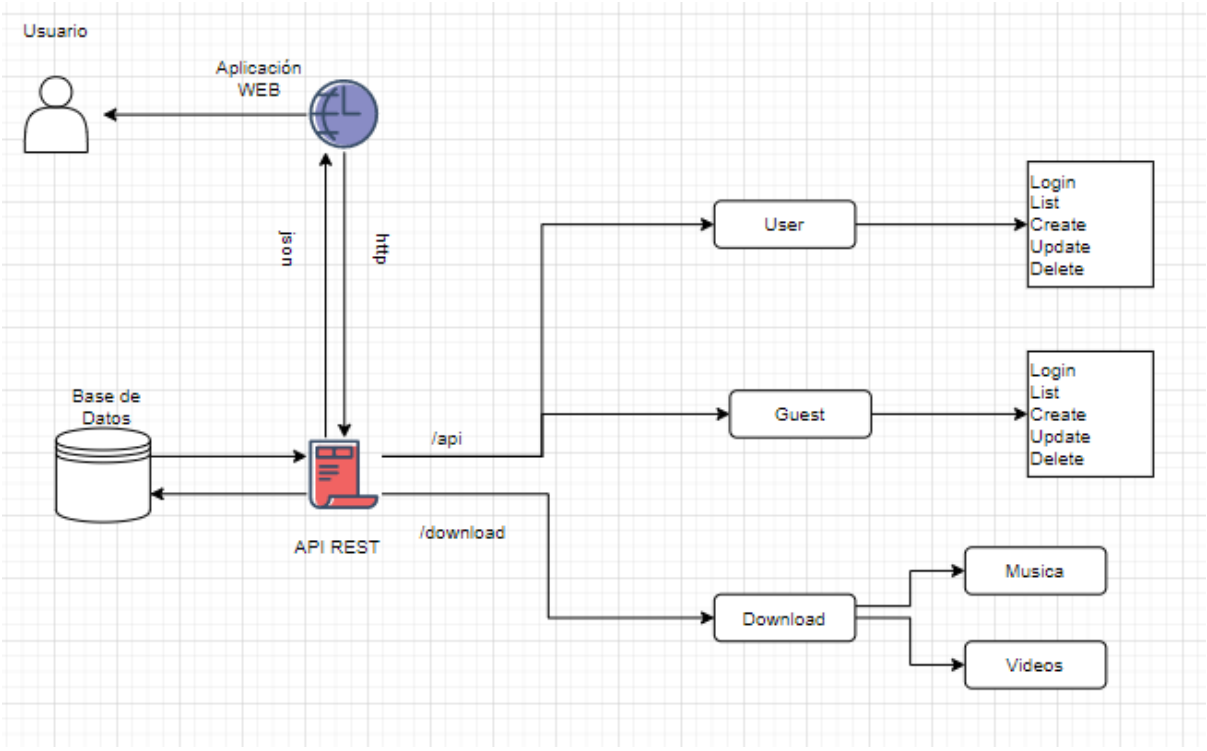


1.2. Rest API.

Se ha creado una API REST con Node.js qué estará escuchando una vez iniciada en el puerto 3000. Esta consta de dos partes:

- **Comunicación con la base de datos**: se han creado los controladores correspondientes a las dos tablas de la base de datos, una correspondiente a cada tipo de usuario.
  Para esta comunicación se utiliza el patrón de Object Pool, para lidiar con una serie predeterminada de conexiones a la base de datos.

- **Downloader**: en este caso hay dos rutas diferentes(siendo su base utl “/download”), una para descargar música y otra para descargar videos.
  Para ambas se ha utilizado el complemento de youtube-dl, servidor utilizado para poder descargar videos desde node (aunque hay videos, que, debido a las características de youtube, no comparten el enlace de descarga, por lo que no será posible esta acción (contemplado dentro de las excepciones posibles de la operación))

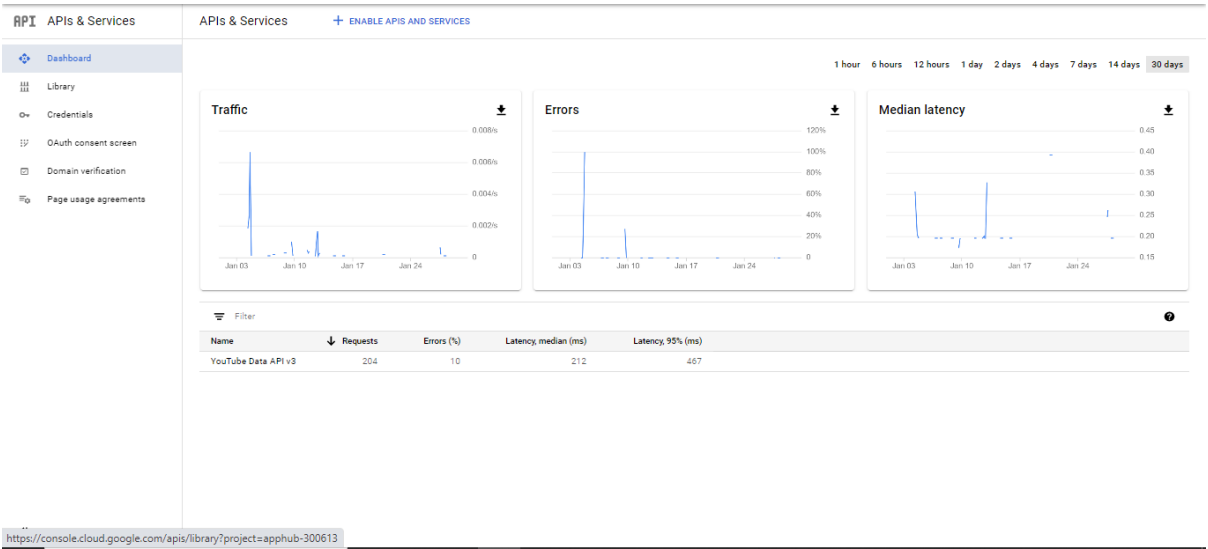


1.3. Base de Datos.

En este caso se ha implementado una base de datos sencilla en mysql con una tabla por tipo de usuario, tipo guest y tipo usuario admin para la posibilidad y control de permisos y/o rol dentro de la aplicación.
El script y montaje de la base de datos se realiza en el punto 7. Manual de Usuario

1.4. Youtube Data API.

Está API, para su uso, deberá ser activada desde la consola de developer de google. Como se mencionó con anterioridad, tiene un límite de envío de datos de forma diaria (por usuario de gmail).



Para el acceso a esta API hay varias formas, la utilizada en nuestra aplicación será de tipo http, utilizando una petición con la url especificada para la comunicación:

- "https://youtube.googleapis.com/youtube/v3/search?part=snippet&maxResults=10"

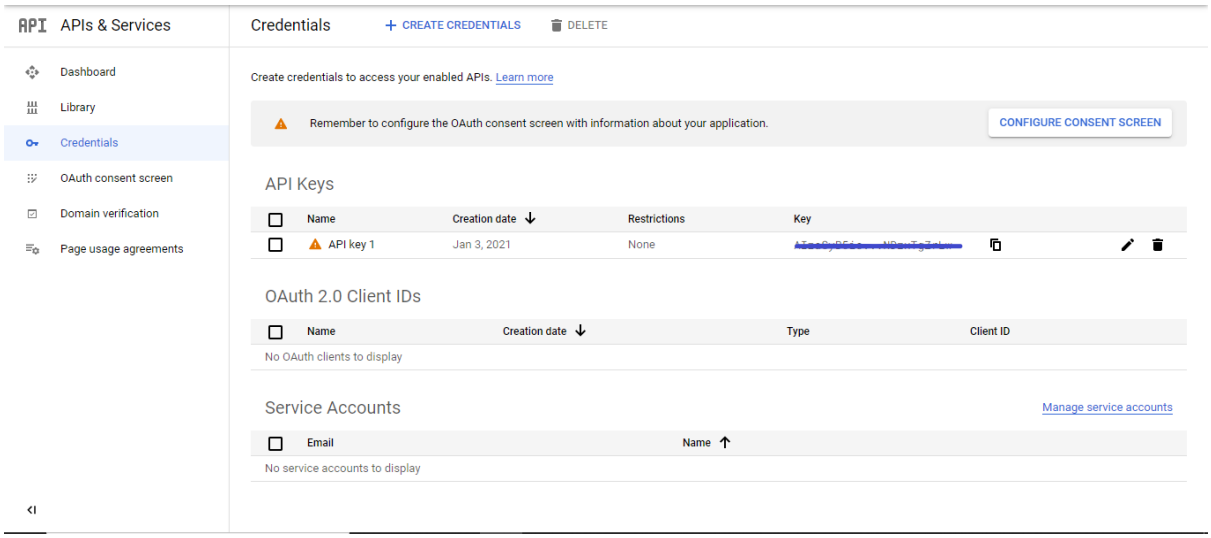
A esta url, se deberá añadir la búsqueda definida, por id o por título de video, para esto se extiende de la siguiente manera:

- Búsqueda por título (se añadirá a la url anterior): "&q="

Las ordenaciones se harán con los apartados siguientes:

- Rating (se añadirá a la url anterior): "&type=video&videoCaption=closedCaption&key="
- Date (se añadirá a la url anterior): "&order=date&key="

En este caso se especificará la key del usuario con la API, especificada en la consola de developer.



Por defecto, la API devuelve los resultados de los videos en formato .json, con las características propias de cada video:

- videoId
- videoUrl
- channelId
- channelUrl
- channelTitle
- title
- publishedAt
- description
- thumbnail

Esto se guardará en una lista para que, para que, una vez se seleccione un video, se utilice la información para su visualización, enviándola en formato .json con la siguiente url:

- 'https://www.youtube.com/watch?v='
    - Posteriormente se incluirá la información anterior del video seleccionado

3.PATRONES UTILIZADOS

En este caso se han implementado de manera propia 3 patrones de creación, 3 patrones estructurales y 4 patrones de comportamiento.

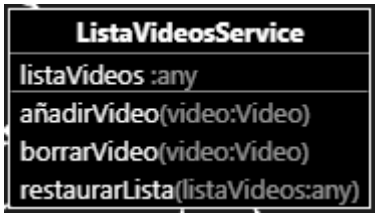
2.1. Patrones de Creación

En esta sección se presentarán los patrones de creación, los cuales corresponden a patrones de diseño de software que solucionan problemas de creación de instancias, y además ayudan a encapsular y abstraer las creaciones.

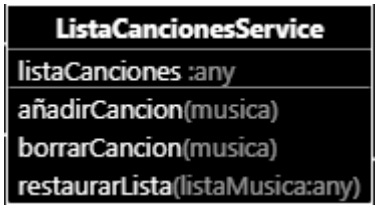
2.1.1. Singleton

Objetivo: Garantiza que una clase tenga una sola instancia y proporciona un punto de acceso global a ella. Todos los objetos que utilizan una instancia de esa clase usan la misma instancia. El resultado de aplicar este patrón es que solo puede existir al mismo tiempo una instancia de una clase, que será accedida a través de un punto de acceso.

ListaVideosService



ListaCancionesService

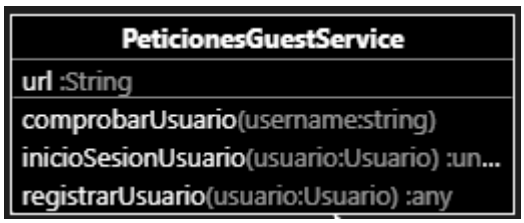


Aclaración: debido a la naturaleza de los servicios en Angular, estos se desarrollan aplicando el patrón singleton de forma que, se crea una única instancia al inicio y es accesible globalmente por todos los módulos o componentes de la aplicación.

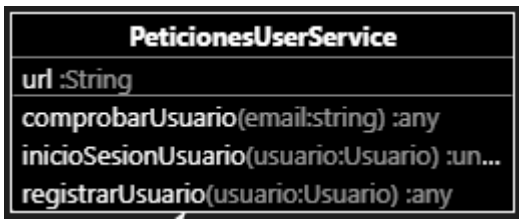
Uso: Gracias a la naturaleza de este patrón, se puede emplear para evitar los accesos a la base de datos cada vez que se quiera realizar una búsqueda, bien de un vídeo o de una canción. De modo que se almacenan en un único objeto singleton al que tengan acceso todos los elementos dentro de la aplicación. En nuestro caso, se utilizarán dos singleton para almacenar las listas de videos y canciones guardadas.

Adicionalmente se utilizan como singleton aquellos servicios de comunicación con el back, para evitar crear instancias innecesarias a lo largo de la ejecución
Además, se aplica a las fachadas relacionadas con los downloaders y las peticiones anteriormente mencionadas, a modo de evitar tener instancias innecesarias para la creación de objetos, disminuyendo los recursos necesarios y conservar la corrección del propio patrón singleton, evitando la posibilidad de que, al crear varias fachadas, se contemplen varias instancias de la conexión a la base de datos.

PeticionesGuestService



PeticionesUserService

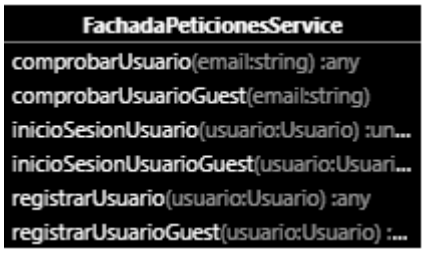


MusicDownloaderService

VideoDownloaderService

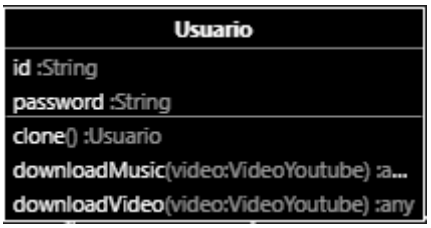
FachadaDescarga

FachadaPeticiones



2.1.2. Prototype

Objetivo: Especifica los tipos de objetos a crear por medio de una instancia prototípica, y crea nuevos objetos copiando dicho prototipo.
Este patrón se usa en los casos en los que crear una instancia de una clase sea un proceso muy complejo y requiere mucho tiempo. Lo que hace es crear una instancia original, y, cuando necesitemos otra, en lugar de volver a crearla, copiar esa original y modificarla.



Uso: la clase abstracta Usuario utiliza el método clone para devolver un nuevo objeto al usuario con los mismos valores, realizando una copia superficial del objeto.
De esta clase heredan el usuario administrador como el usuario guest.
De estos se crea un prototipo en el componente de registro, de manera que, se utilice el prototipo a la hora de crear nuevos usuarios, para tener un usuario tipo predeterminado, del que se modifiquen los valores introducidos pero sin modificar el original.

Esto se realiza de la siguiente manera:

- Definición de valores:

```
usuarioDefault = new Usuario("juangh99@gmail.com", "*****")
guestDefault = new Usuario("juangh99", "*****")
```

- Clone:

```
async register(){
  if(this.valor==1){
    this.usuarioRegistro = this.usuarioDefault.clone()
  }
  else{
    this.usuarioRegistro = this.guestDefault.clone()
  }
  this.usuarioRegistro.id = this.id
  this.usuarioRegistro.password = this.password
  var data = await this.factorialogin.crearEstrategia(this.valor).ejecutar(this.usuarioRegistro)
  if(data){
    alert("Se ha registrado el usuario correctamente, Inicie Sesión")
    this.router.navigateByUrl("Login")
  }
  else
    alert("No se ha podido registrar el usuario, vuelva a intentarlo")
}
```

2.1.3. Factory Method

Objetivo: Define una interfaz para crear un objeto pero deja que sean las subclases quienes decidan qué clase instanciar. Permite que una clase delegue en sus subclases la creación del objeto. Se utiliza este patrón cuando no se puede prever la clase de objetos que se debe instanciar, o una clase quiere que sean sus subclases quienes especifiquen los objetos que ésta crea.

FactoriaEstrategiasLogin

FactortiaEstrategiasRegister

Uso: Para entrar a la aplicación se deberá, bien iniciar sesión o crear un usuario nuevo. Dentro de cada una de las opciones se encuentra un factory method para la creación de un usuario mediante un prototype. Para el caso de Login se determinará si inicia sesión como usuario o como invitado, según los datos que introduzca. Para el caso del registro, tendrá un funcionamiento similar en lo que respecta a usuarios e invitados, salvo que en este caso servirá para determinar qué tipo de usuario se crea.

Adicionalmente, a la hora de buscar videos en el módulo de Youtube, se pueden escoger dos formas de ordenación, por rating y por date, por lo que se ha creado una factoría de estrategias, en la que, se crea y devuelve la estrategia definida de ordenación en tiempo de ejecución, siendo esta la escogida por el usuario.

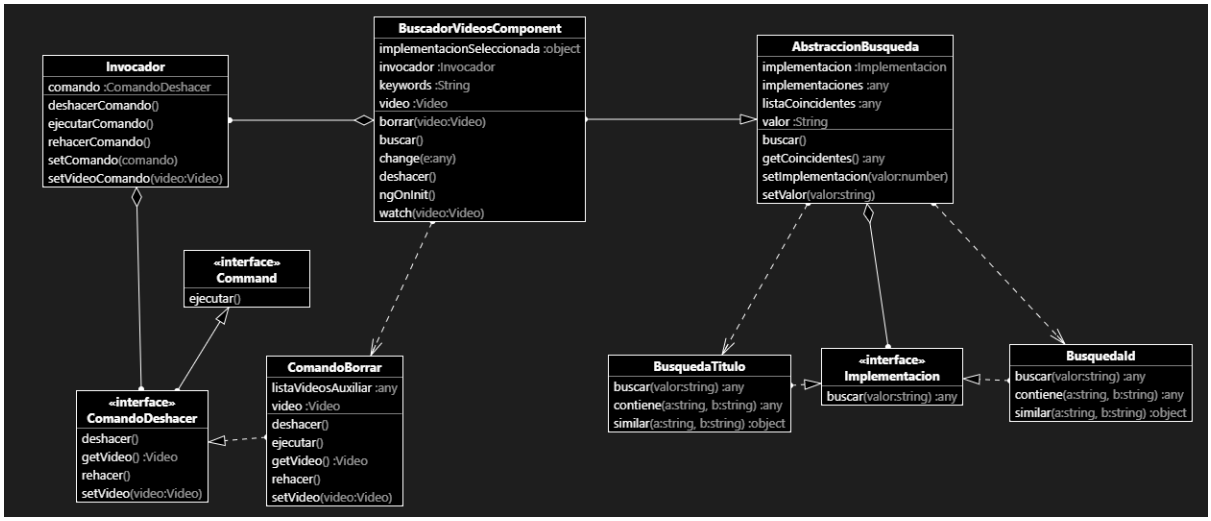
2.2. Patrones Estructurales

2.2.1. Bridge

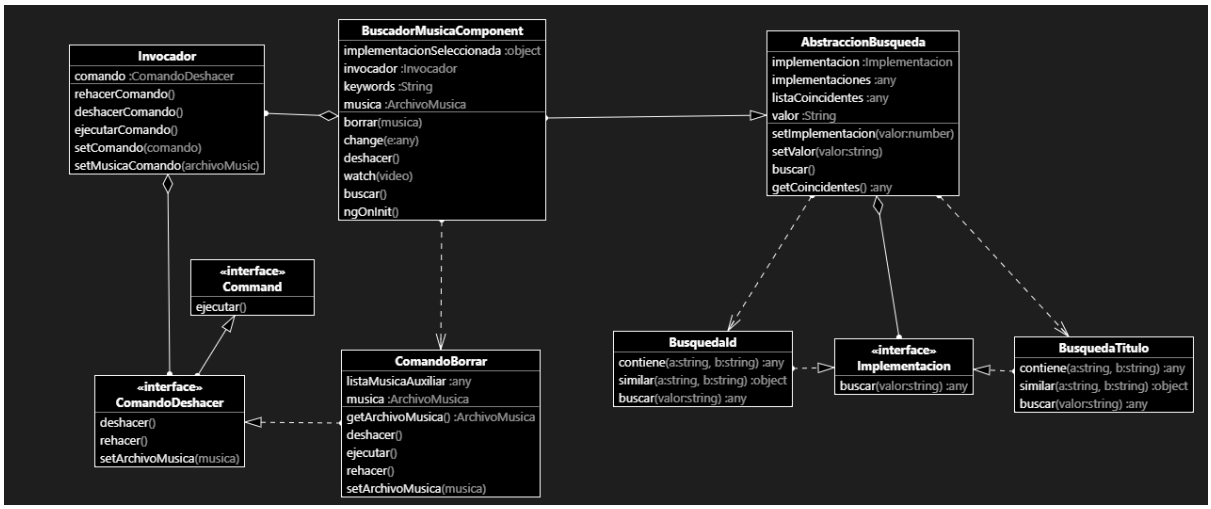
Objetivo: Desacoplar un componente complejo en dos jerarquías relacionadas: la abstracción funcional de su implementación interna. De esta forma ambas partes pueden variar de forma independiente y el cambio de cualquier aspecto del componente se hace de manera más fácil.

Uso: Este patrón está específicamente aplicado para desacoplar la parte del buscador (abstracciónBusqueda) de todas las opciones de búsqueda. Esto con el fin de poder facilitar los cambios, en caso para poder aplicar nuevas opciones de búsqueda dentro de las que ya hay presentes en la aplicación.
Se aplica en los dos componentes de buscador, tanto el de música como el de videos. Además, se crea una abstracción refinada (el componente) del buscador, a la cual, se le añade la funcionalidad propia del patrón de diseño command para poder borrar y deshacer el borrado de videos.

buscador-video



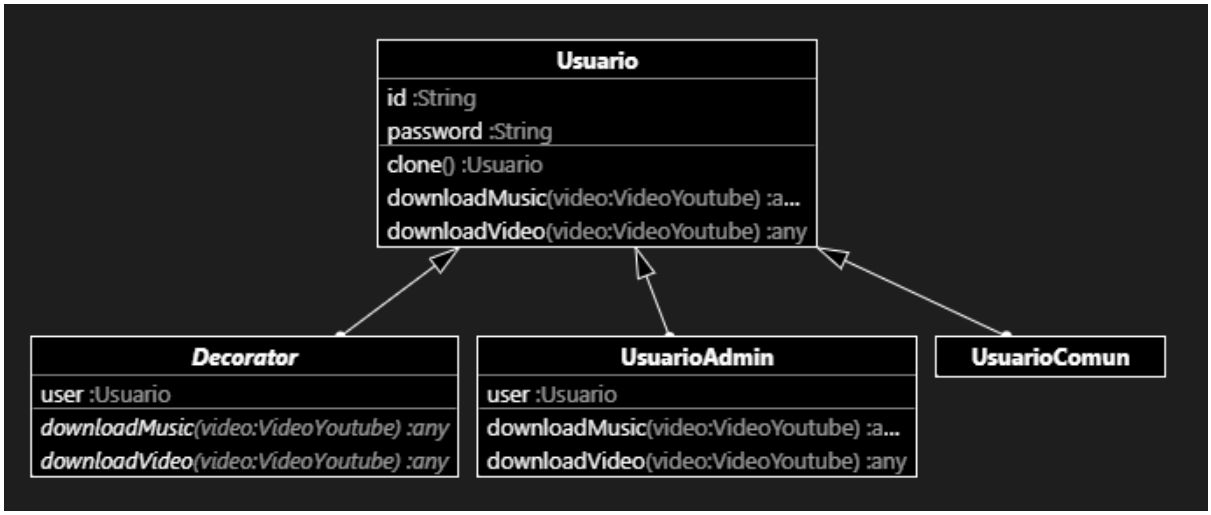
buscador-música



2.2.2. Decorator

Objetivo: Añadir nuevas responsabilidades dinámicamente a un objeto sin modificar su apariencia externa o su función, es una alternativa a crear demasiadas subclases por herencia. Se utiliza este patrón cuando se desea añadir funcionalidad a una clase sin las restricciones que implica la utilización de la herencia, o cuando se desea añadir dicha funcionalidad de forma dinámica en tiempo de ejecución, de manera transparente a los usuarios.

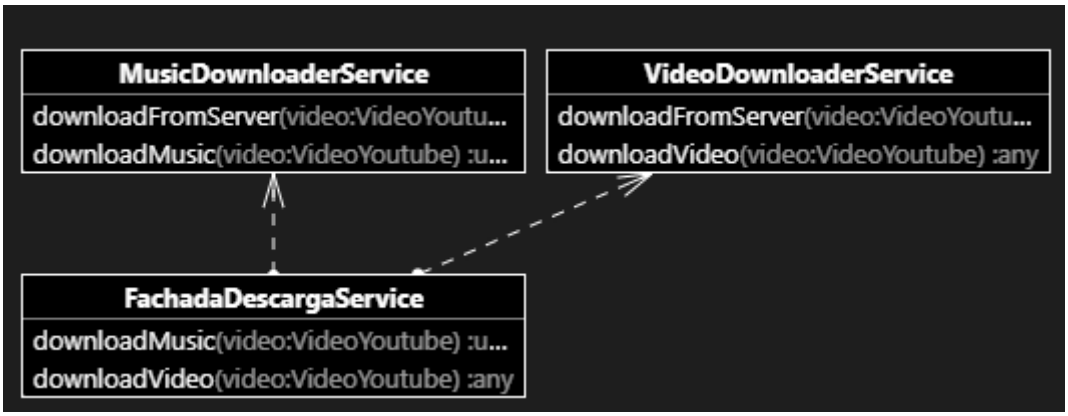
Uso: El decorador se utilizará para extender la funcionalidad de los usuarios únicamente en el caso de que no sean invitados. De forma predeterminada se parte de un usuario invitado, el cual no tiene funcionalidad de descarga, y se comprueba si el usuario presente es o no administrador, en caso afirmativo, este decorador añade la funcionalidad de descarga, permitiendo a los usuarios poder descargar música después de haber buscado un vídeo en youtube. Esto se aplica tanto para descargar música (formato mp3), como descargar videos (formato mp4).



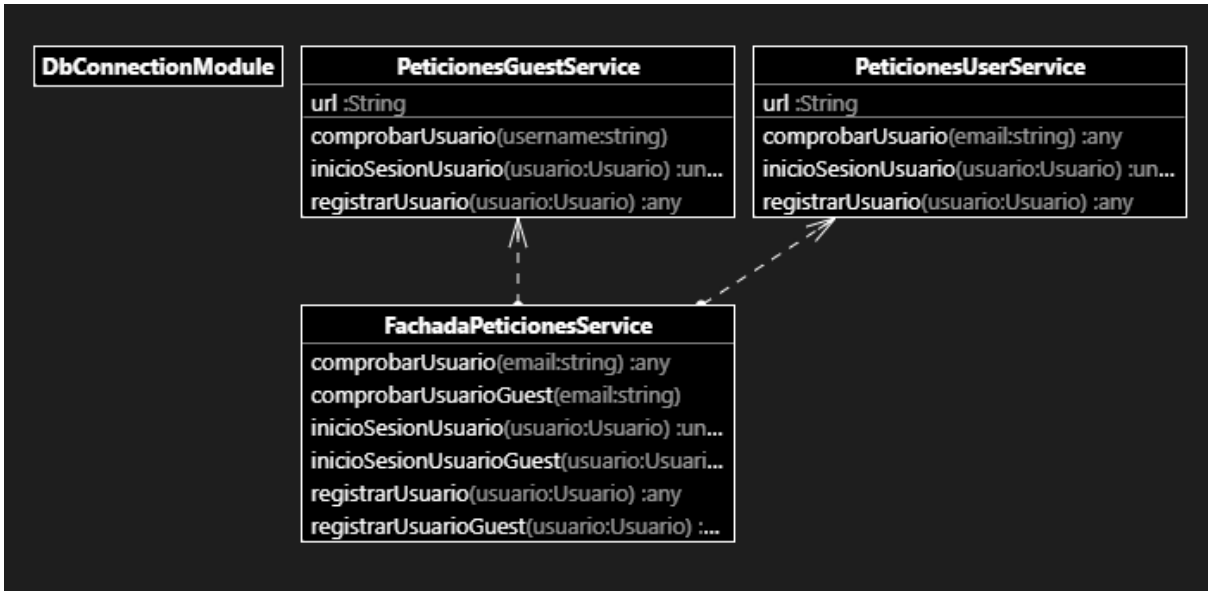
2.2.3. Facade

Objetivo: simplificar el acceso a un conjunto de subsistemas o un sistema complejo. Representa una única interfaz unificada, que envuelve el subsistema, y es responsable de colaborar con los componentes del subsistema. Este patrón ofrece un punto de acceso al resto de las clases, facilitando una interfaz sencilla para el acceso a subsistemas.

Uso: Para facilitar el acceso dentro de nuestra aplicación a los diferentes tipos de descarga, se implementa un patrón de fachada. Esto permite tener una mejor accesibilidad a la hora de acceder a cualquiera de las dos opciones. La clase encargada de conectar los distintos tipos de descarga será, como se puede ver, FachadaDescargaService.



Por otro lado, se encuentra aplicado este patrón a las peticiones a la base de datos. Estas peticiones se dividen entre peticiones de usuario y peticiones de invitado, pero quedan unificadas gracias al uso de este patrón para una mejor accesibilidad.

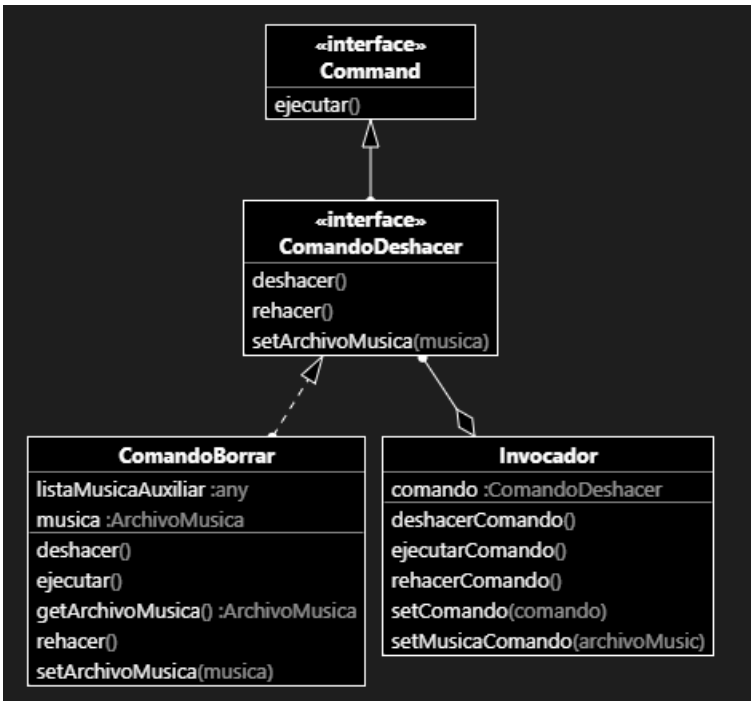


2.3. Patrones de Comportamiento

2.3.1. Command

Objetivo: encapsular un comando en un objeto. Este objeto contiene el comportamiento y los datos necesarios para una acción específica. Permite parametrizar a los clientes con diferentes peticiones, hacer cola o llevar un registro de las peticiones. Se utiliza este patrón cuando se desea desacoplar la fuente de una petición del objeto que la cumple.

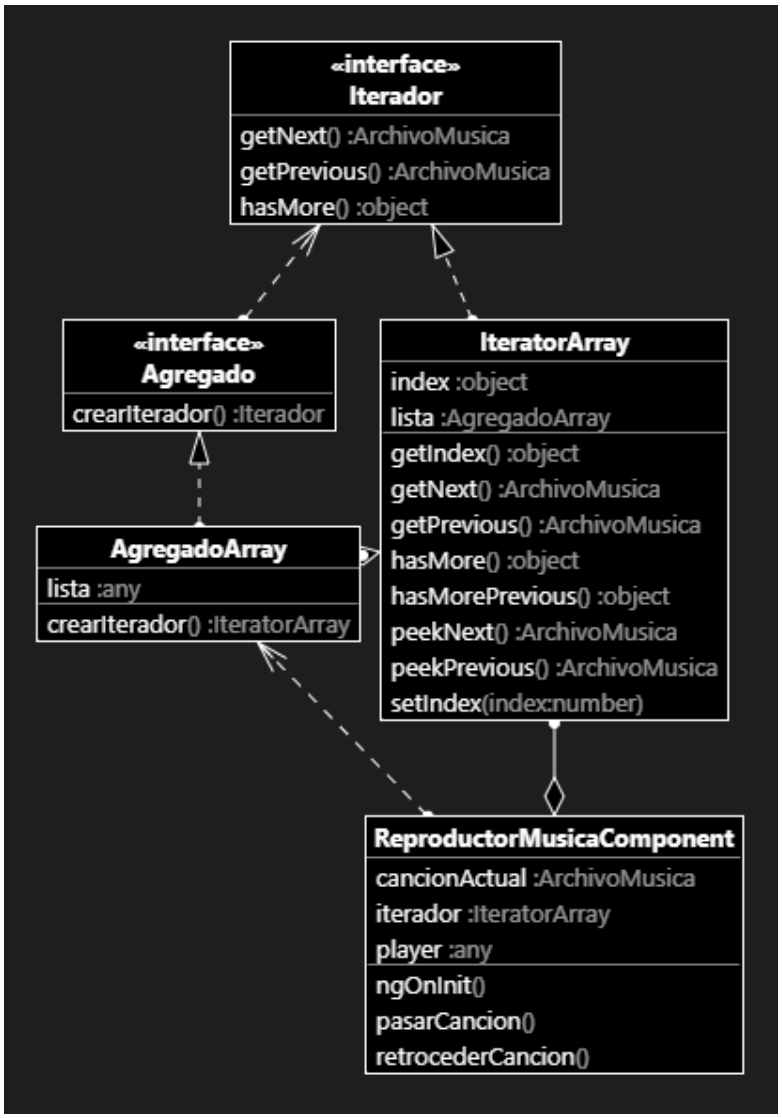
Uso: En nuestra aplicación, ya que se pueden realizar acciones para eliminar los vídeos o música descargada, se ha decidido implementar el patrón command para permitir deshacer acciones como la mencionada. Esto con el fin de evitar acciones delicadas, que de otro modo serían irreversibles.



2.3.2. Iterator

Objetivo: Proporcionar una forma coherente de acceder secuencialmente a los datos de una colección, independientemente del tipo de colección.

Uso: En nuestra aplicación se hace uso de un reproductor de música, por lo que para sacarle el máximo partido a su funcionamiento se aplica el patrón iterador implementado las funcionalidades básicas, así como la posibilidad de moverse hacia atrás y hacia delante en la lista de canciones realizando las comprobaciones propias de existencia de valores. Adicionalmente, permite el movimiento cíclico a través de la lista, de forma que, si se llega a un extremo de esta, se pueda seguir iniciando por el extremo contrario (es decir, si está en el final, se continua con la canción del principio de la lista).

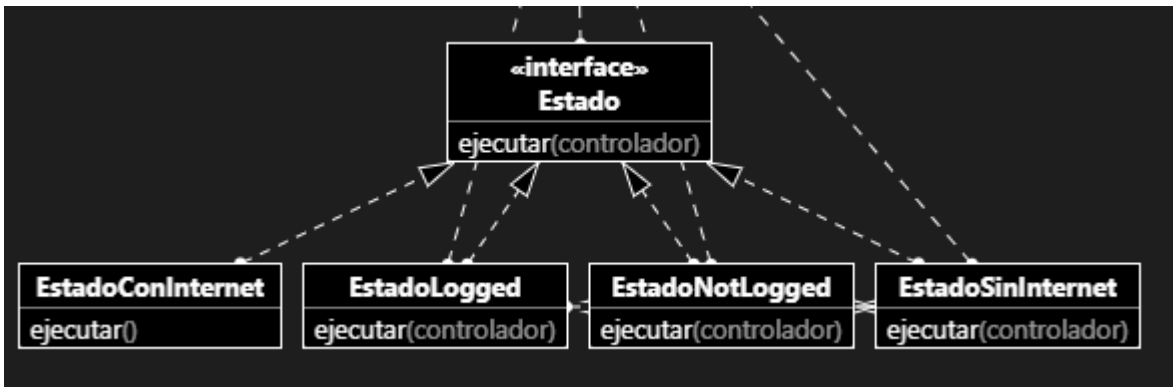


2.3.3. State

Objetivo: Permitir que un objeto se comporte de distinta forma dependiendo de su estado interno, como si cambiase la clase a la que pertenece. Permite cambiar fácilmente el comportamiento de un objeto en tiempo de ejecución.

Uso: Se ha implementado el patrón state para determinar los múltiples estados en los que podrá estar la aplicación, siendo esto conectado o no a internet, y con sesión iniciada o no (son 4 en total). El estado de la aplicación afectará al resto de la funcionalidad de la misma puesto que:

- Sin internet no se podrán realizar búsquedas en youtube
- En el estado sin loggear no se puede realizar ninguna funcionalidad externa al login o al registro de usuarios
- El estado logged es el estado de funcionamiento completo, por el cual se pueden acceder a todas las funcionalidades propias del usuario en la aplicación

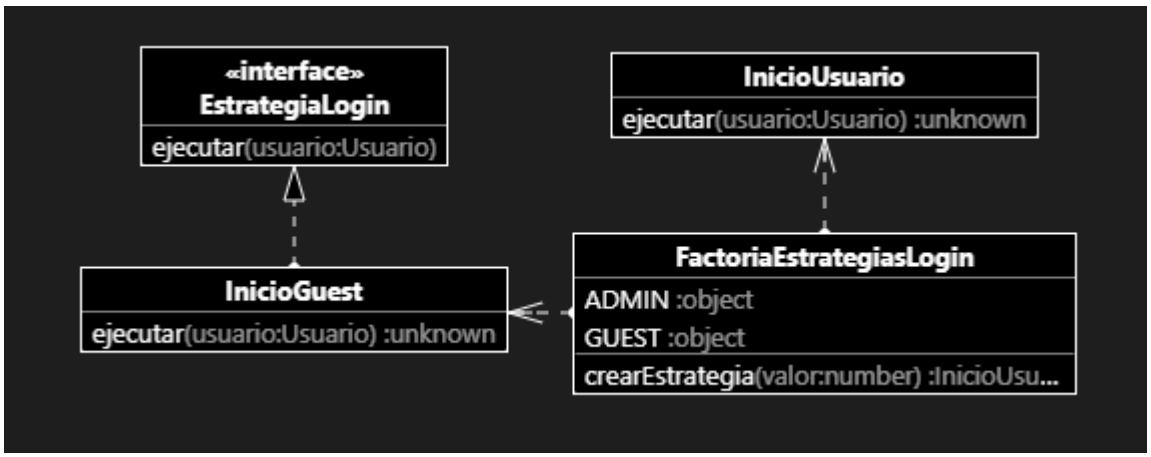


2.3.4. Strategy

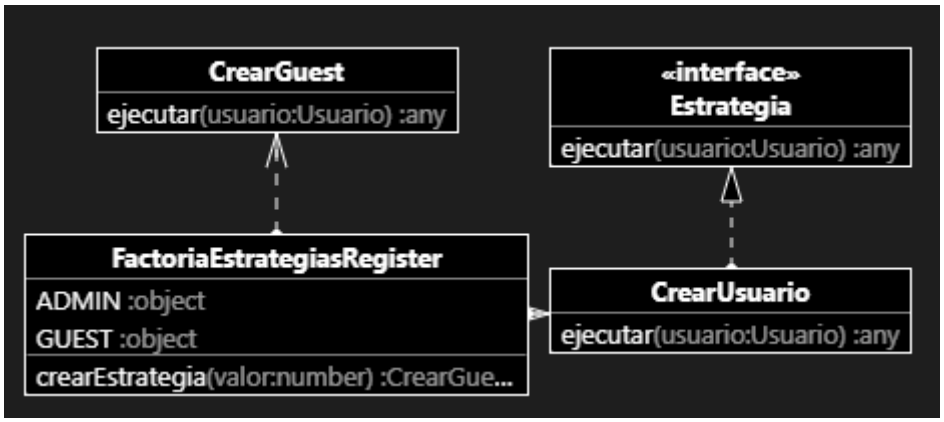
Objetivo: Definir un grupo de clases que representan un conjunto de posibles comportamientos. Estos comportamientos pueden ser fácilmente intercambiados en una aplicación, modificando la funcionalidad en cualquier instante

Uso: Dentro de nuestra aplicación se ha utilizado el patrón strategy para diversos casos. Para empezar se aplica tanto a login como register, en conjunto del factory method para el inicio de sesión y el registro de usuarios nuevos.
De esta forma se diferencian las diferentes posibilidades a la hora de hacer un registro, con la posibilidad de utilizar un usuario administrador o un usuario invitado.

EstrategiaLogin

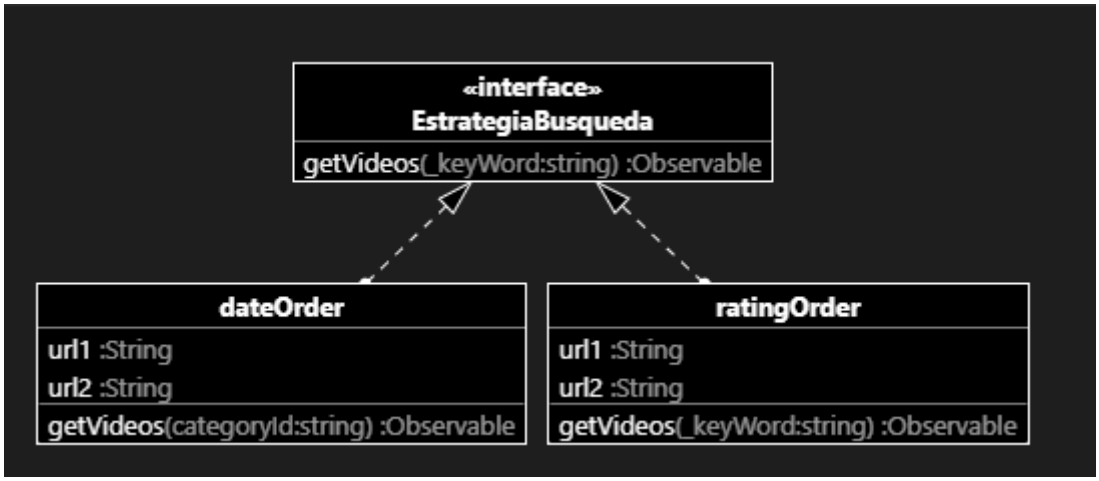


EstrategiasRegister



Por otro lado, se encuentran las estrategias utilizadas para las búsquedas de canciones, tanto para el buscador en youtube, como la lista de descargas.
Aquí se pueden diferenciar dos. Por un lado la búsqueda y ordenación dentro de youtube, llevada a cabo por fecha o por rating (puntuación dentro de youtube).
Cada una de estas opciones se marca en un selector dentro de la aplicación a la hora de realizar la búsqueda.

EstrategiaBusqueda



3.DIAGRAMAS DE CLASES

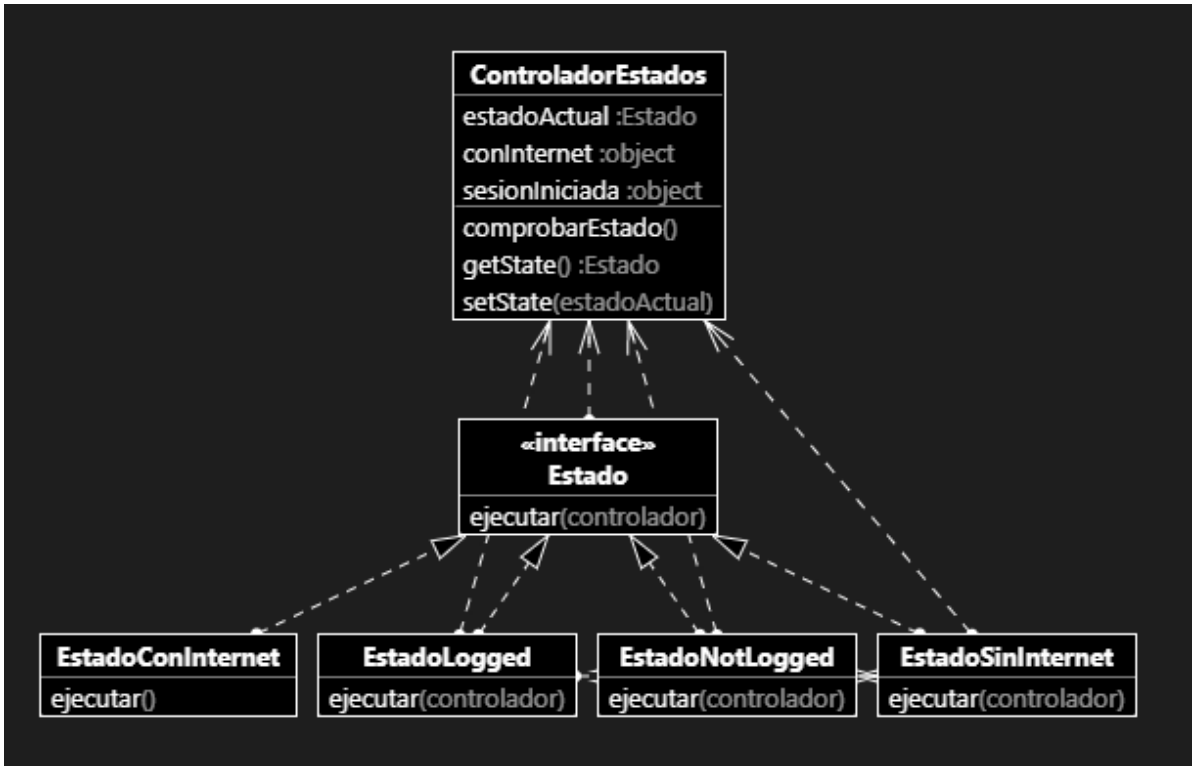
Para poder tener una mejor idea de las clases empleadas y las interacciones entre ellas, se procederá a explicar los distintos módulos en los que está dividida la aplicación, y cuales son

los diferentes componentes de los mismos, junto a un diagrama del módulo para mayor claridad.

3.1. appState

Este módulo consistirá de la clase ControladorEstados junto a las clases que conforman el patrón State. La función principal del módulo será la de tener un control sobre el estado o estados actuales de la aplicación, siendo estos si está conectado o no a Internet, o si el usuario ha iniciado o no sesión., con sus consecuentes características (por ejemplo, si no se esta logueado, no se podrá mover de las pantallas de inicio de sesión o registro).

Cada estado dentro de la aplicación extenderá la funcionalidad de la interfaz Estado, y únicamente ejecutará el controlador si procede.

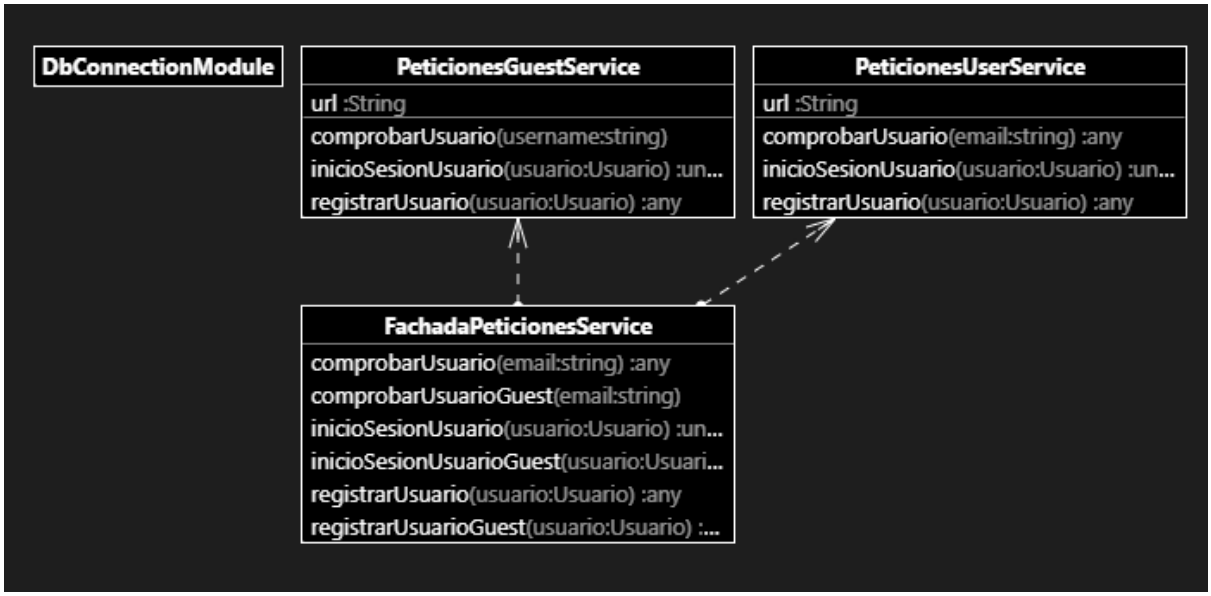


3.2. db-connection

Este módulo consistirá de las clases que interactúan con la base de datos al iniciar sesión o al crear un usuario nuevo, junto al patrón fachada que los unifica para una mejor accesibilidad.

Bien al iniciar sesión o al crear un usuario, se accederá a la clase que actuará de fachada y después, dependiendo del tipo de usuario, la acción será llevada a cabo por la clase de peticiones del invitado o del usuario.
Ambas clases constan de 3 funciones, relativas a comprobar el usuario, iniciar sesión de uno de los dos tipos, o de registrar un nuevo usuario, según la acción elegida en un principio.

- comprobarUsuario verificará si existe un usuario administrador o invitado con los datos introducidos.
- inicioSesionUsuario creará o no una sesión una vez se comprueban que los datos del usuario (username/email y contraseña) son correctos.
- registrarUsuario se encargará de crear un nuevo usuario dentro de la base de datos para poder después iniciar sesión desde la página de login.



En este caso, DbConnectionModule hace referencia al módulo propio de Angular, de forma que mantiene en las clases pertenecientes al módulo imports de librerías, así como su conexión con el componente general de la aplicación.

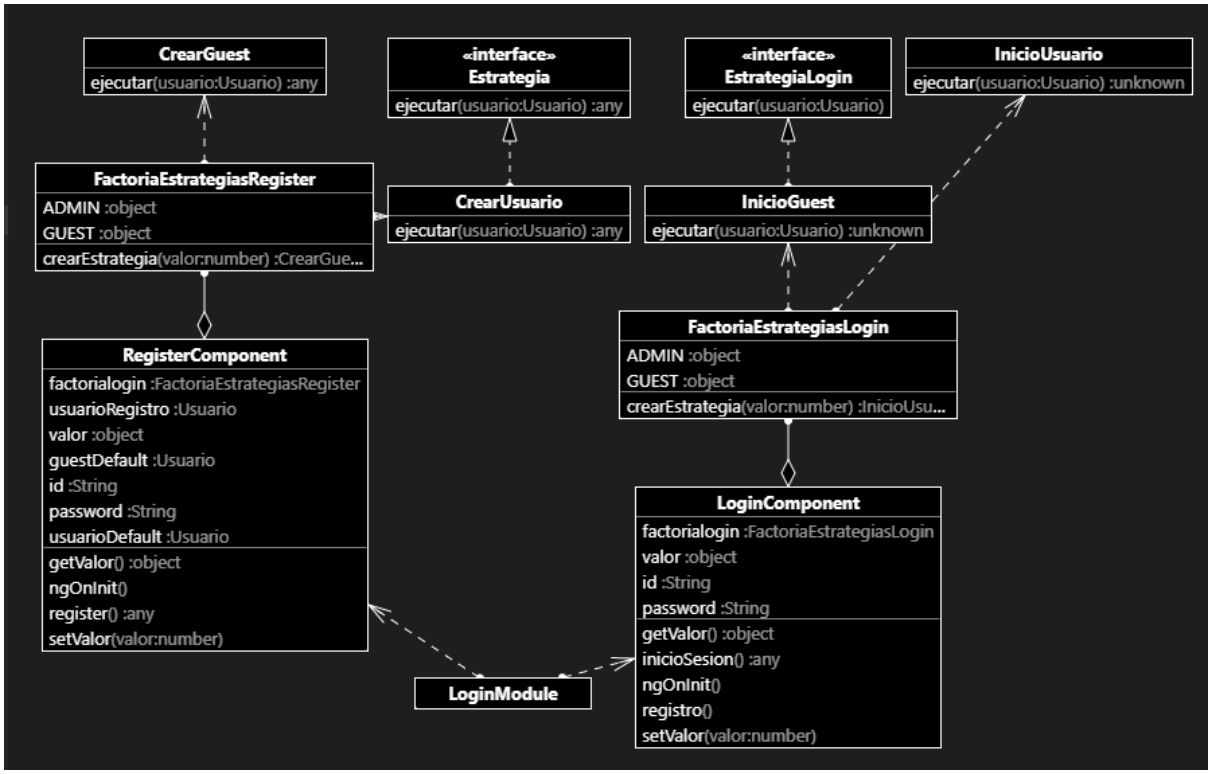
3.3. Login

El módulo login será el encargado de llevar la lógica relativa al login. Este módulo estará conectado a db-connection, pero no realizará ninguna acción que incumba a la base de datos (de eso se encarga db-connection).

Este módulo se compondrá de dos factory methods de estrategias, siendo uno de estos dedicado a los registros y el otro al inicio de sesión, representados por las clases FactoriaEstrategiasRegister y FactoriaEstrategiasLogin respectivamente.

Una vez hayan pasado por db-connection, se le pasará la respuesta a una estas factorías dependiendo de la opción seleccionada, y se ejecutará la clase CrearUsuario o InicioUsuario respectivamente, que como bien se deduce, crearán un usuario nuevo o dará acceso al usuario.

Register y Login Component almacenarán los datos necesarios para, bien registrar un usuario nuevo con la información introducida, o para iniciar sesión con la información que ya había sido previamente almacenada en la base de datos.



Al igual que con anterioridad, LoginModule es un módulo propio de Angular y no afecta directamente a la funcionalidad.

3.4. Música-Offline

El módulo de Música-Offline a diferencia de los anteriores módulos tendrá muchos más componentes que lo conformen.

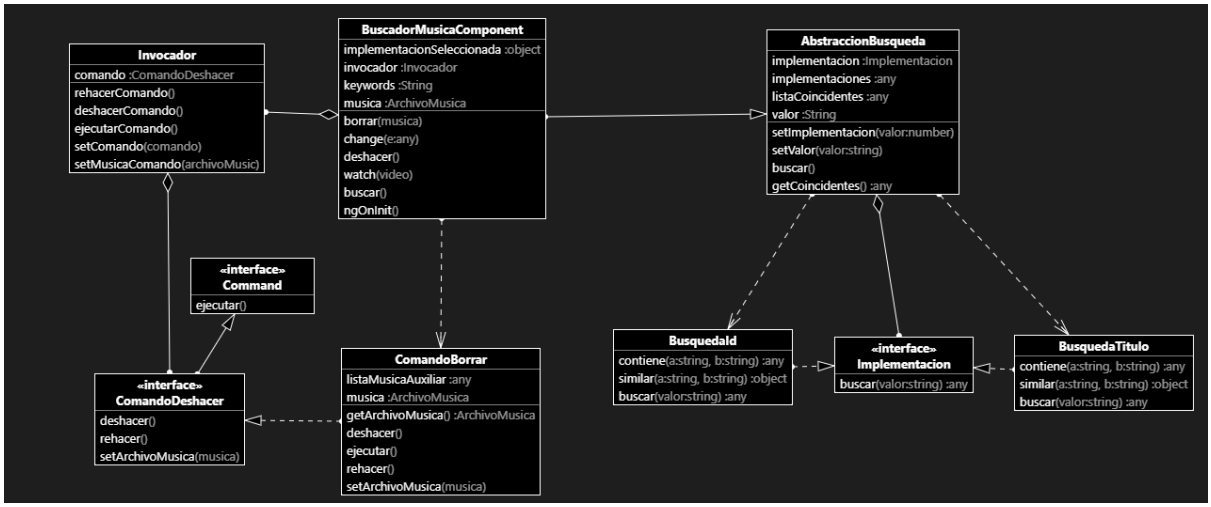
Para empezar, este módulo se divide en dos componentes principales, los cuales son el buscador de música y el reproductor de música.

- buscador-música: Dentro de este conjunto se encuentran las clases que conforman el patrón Bridge, siendo estas:

AbstracciónBusqueda se encargará de las distintas implementaciones de búsqueda que tendrá el patrón Bridge. Estas se dividirán en dos clases, BusquedaId y BusqeudaTitulo, las cuales heredan de la interfaz Implementación y BuscadorMusicaComponent que será la abstracción refinada del patrón.

En el otro lado del patrón se encuentran la implementación interna, junto a las clases referentes a la acción de deshacer, gestionadas en el patrón de diseño command que se implementarán en la abstracción refinada.Estas son: Invocador (Para llamar a los comandos), ComandoDeshacer, ComandoBorrar y Command.

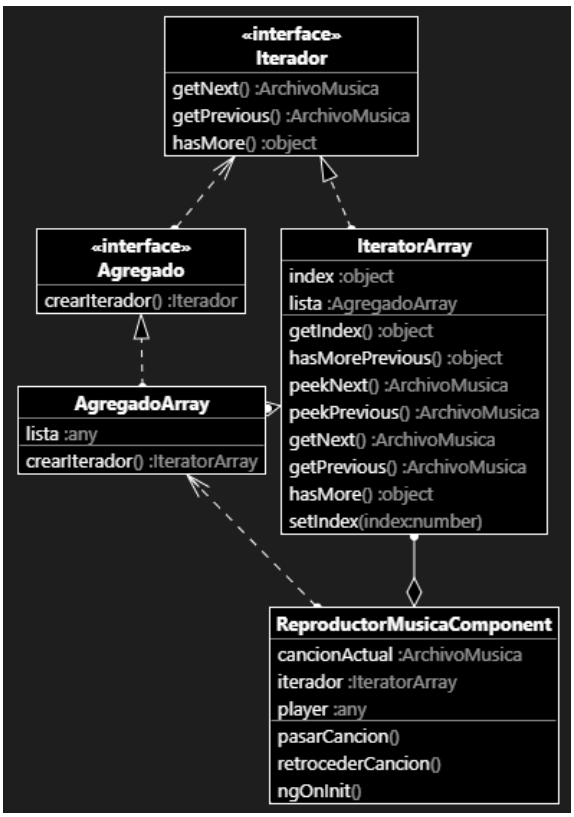
Para los comandos, ComandoDeshacer se encargará de deshacer la acción de borrar un elemento. Mientras que por el contrario, ComandoBorrar se encargará de eliminar un archivo seleccionado definitivamente.



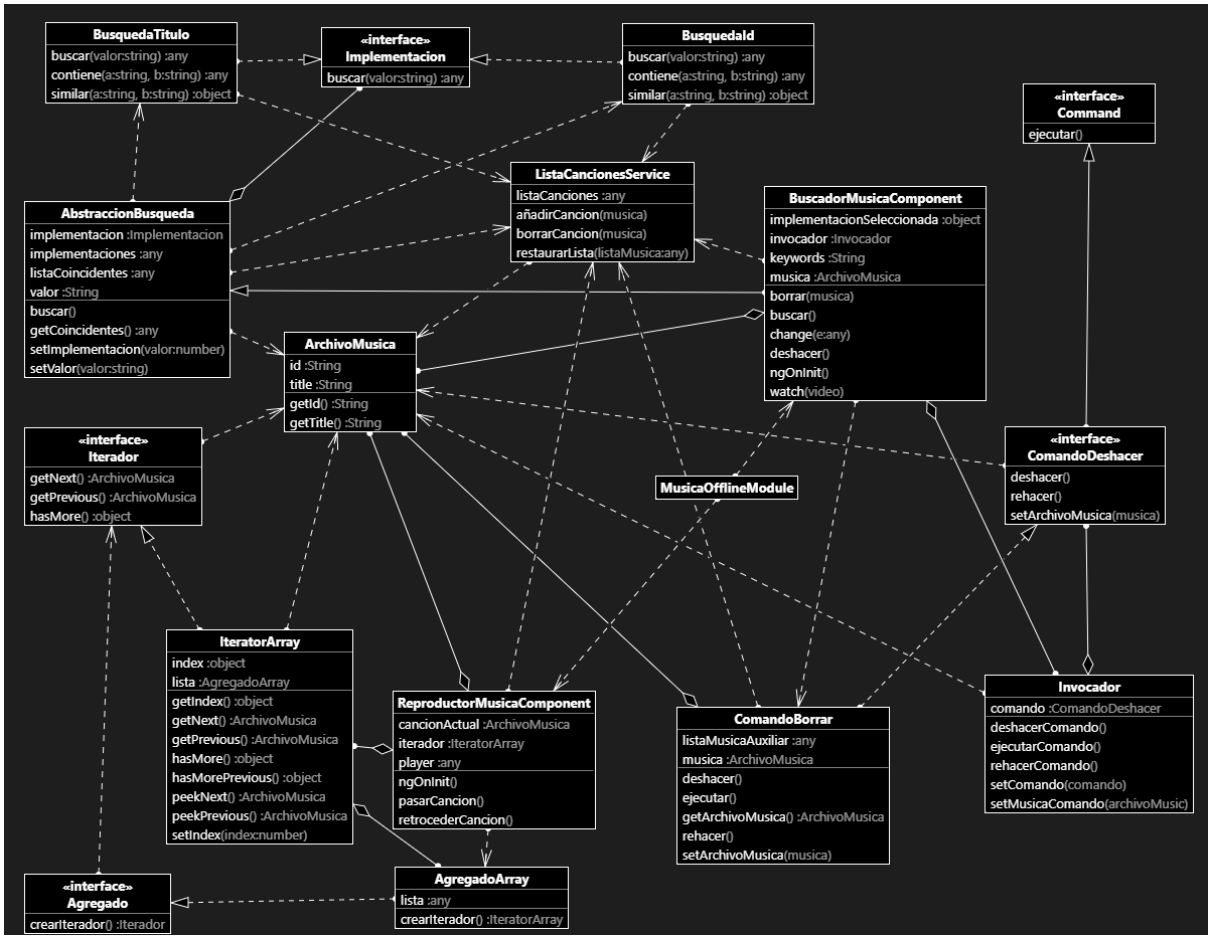
- reproductor-música: El reproductor de música se compondrá del componente ReproductorMusicaComponent, donde se encontrará la canción actual, junto a un iterador para desplazarse por la lista de canciones.

  El iterador se compondrá de una clase IteratorArray que extenderá la interfaz Iterator, junto a la clase AgregadoArray, que extiende a la interfaz Agregado.

  Dentro de AgragadoArray se almacenarán las canciones, las cuales serán recorridas dentro de IteratorArray (haciendo referencia al patrón iterador desarrollado en el apartado de patrones)



Finalmente, se muestra un diagrama de clases con las conexiones de todos los componentes involucrados en la creación del módulo de música off-line

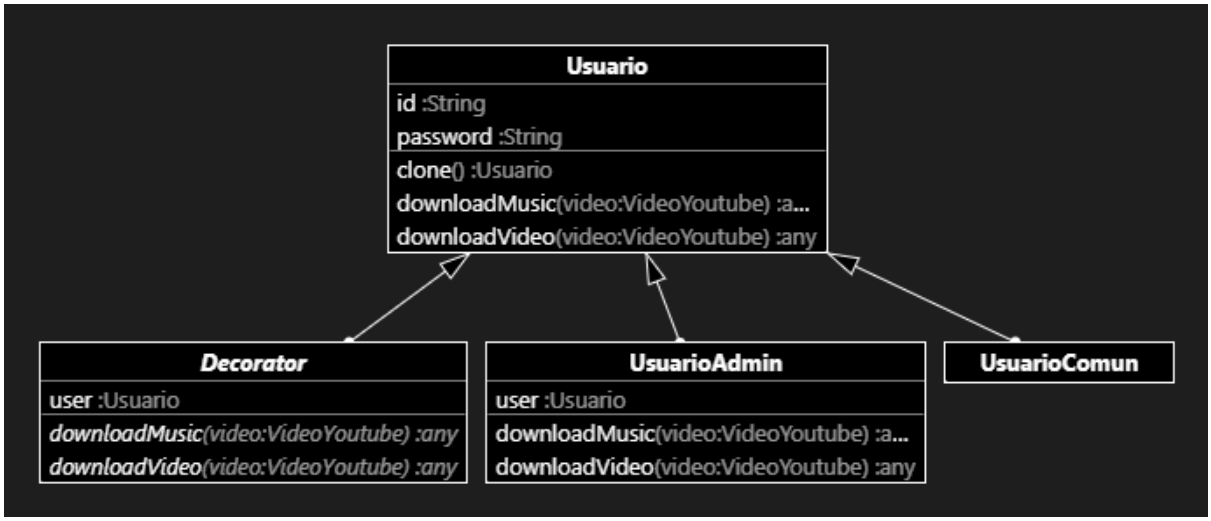


3.5. Usuario

Para el módulo de usuario se puede apreciar que se compone de un patrón decorador. Esto tiene el fin de ampliar la funcionalidad de la clase UsuarioComun, convirtiéndolo en un UsuarioAdmin.

Para empezar se encuentra la clase Usuario, de la que extenderán la funcionalidad el resto de clases. Por un lado el UsuarioComun, que no añade ninguna funcionalidad a la clase Usuario. Por otro lado la clase UsuarioAdmin, la cual se beneficia de la clase Decorator para ampliar su funcionalidad.

Dentro del UsuarioAdmin, además de tener los mismos atributos que la clase Usuario, puede hacer uso de los métodos downloadMusic y downloadVideo, los cuales permiten, evidentemente, descargar música y videos respectivamente.



3.6. Videos-Offline

El módulo de Vídeos-Offline es muy similar, por no decir que estructuralmente igual, al módulo Música-Offline, con la diferencia de que el reproductor no contiene una lista de videos iterable, además de que, utiliza el objeto Singleton de listaVideos, así como clases relacionados con videos, los cuales, posteriormente, se utilizarán para la reproducción de archivos .mp4.

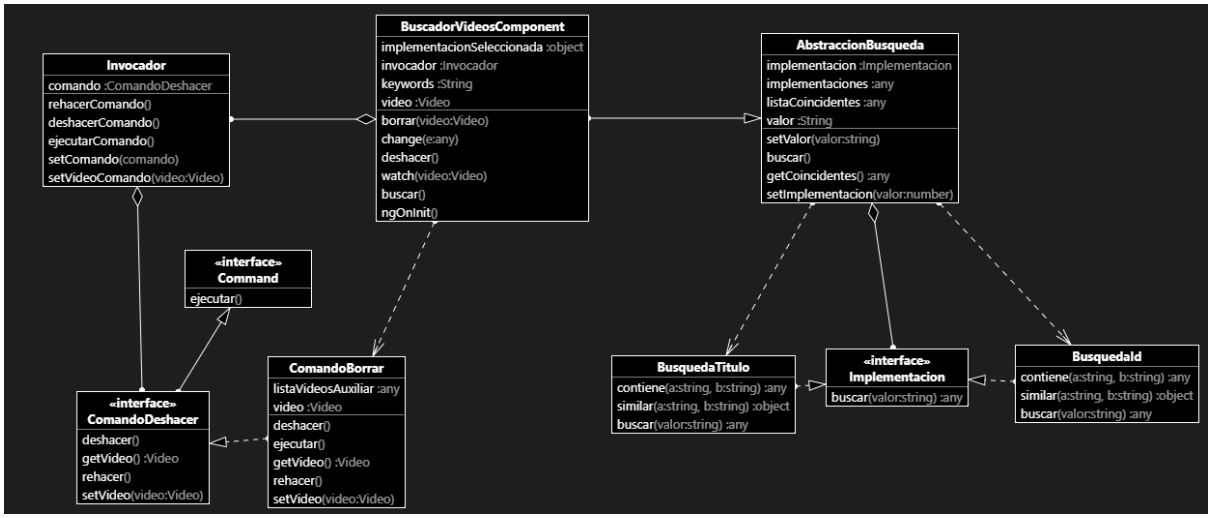
Para empezar, este módulo se divide en dos componentes principales, los cuales son el buscador de música y el reproductor de música.

- buscador-videos: Dentro de este conjunto se encuentran las clases que conforman el patrón Bridge, siendo estas:

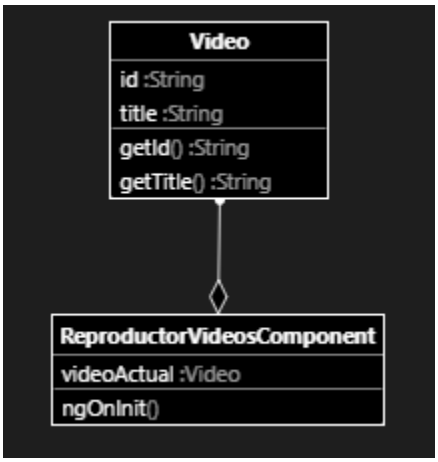
AbstracciónBusqueda se encargará de las distintas implementaciones de búsqueda que tendrá el patrón Bridge. Estas se dividirán en dos clases, BusquedaId y BusqeudaTitulo, las cuales heredan de la interfaz Implementación, BuscadorVideosComponent que será la abstracción refinada que contendrá la funcionalidad del patrón command.

En el otro lado del patrón se encuentran la implementación interna, junto a las clases referentes a la acción de deshacer, gestionadas en el patrón de diseño command. Estas clases serán: Invocador (Para llamar a los comandos), ComandoDeshacer, ComandoBorrar y Command.

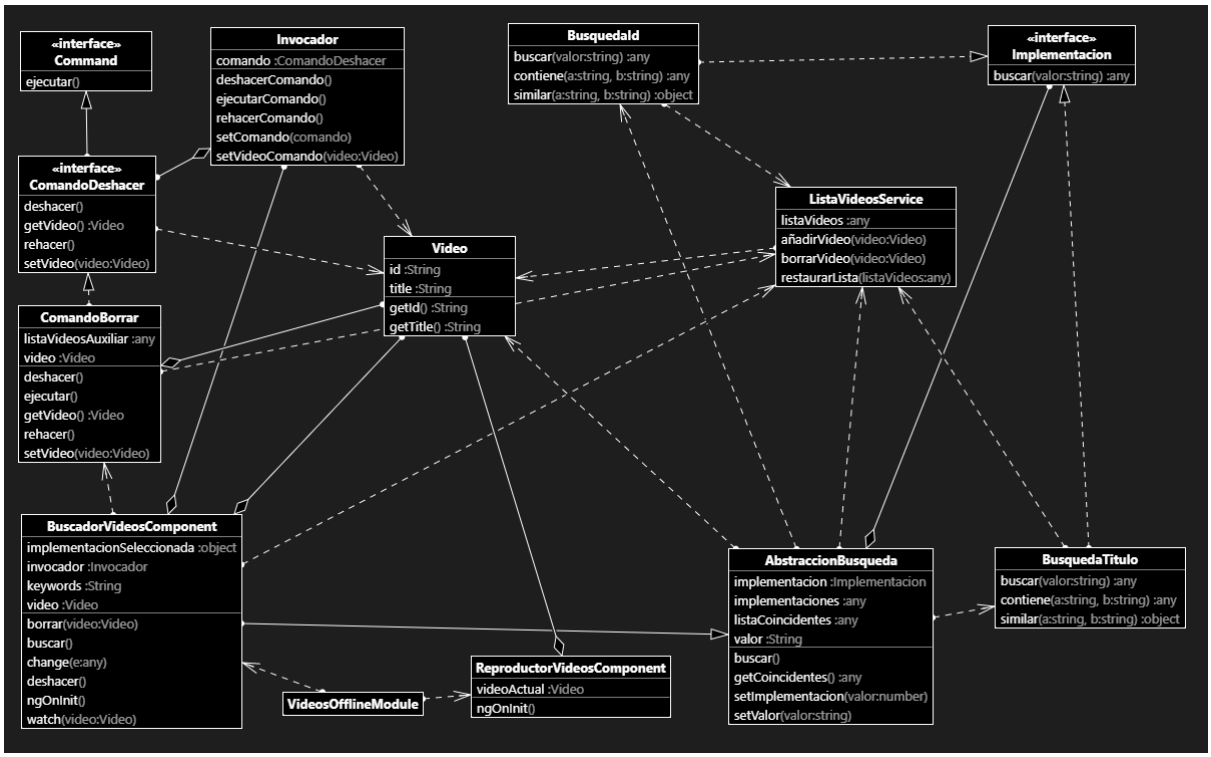
Para los comandos, ComnadoDeshacer se encargará de deshacer la acción de borrar un elemento. Mientras que por el contrario, ComandoBorrar se encargará de eliminar un archivo seleccionado definitivamente.



- reproductor-videos: El reproductor de videos, a diferencia del de música, sólo contiene la información del video actual a reproducir, lo cual, podrá ser visualizado por el usuario, en el propio reproductor de videos presentado por el componente.



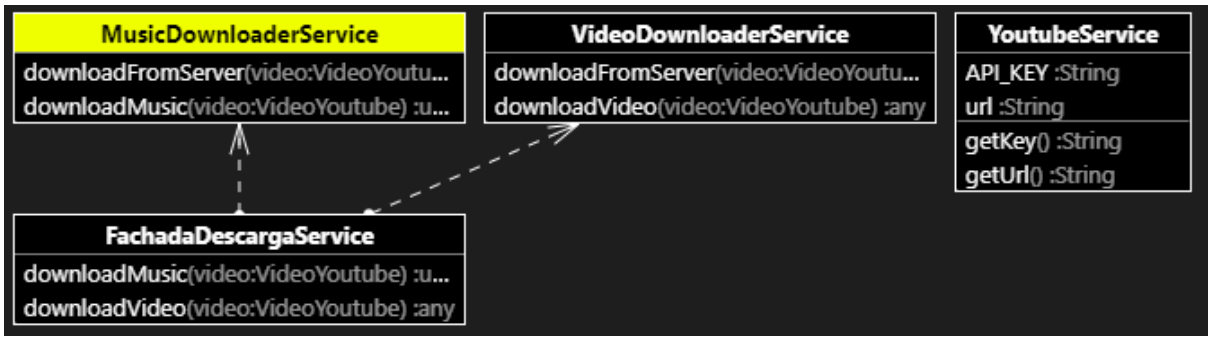
Finalmente, se muestra un diagrama de clases con las conexiones de todos los componentes involucrados en la creación del módulo de música off-line.



3.7. Youtube

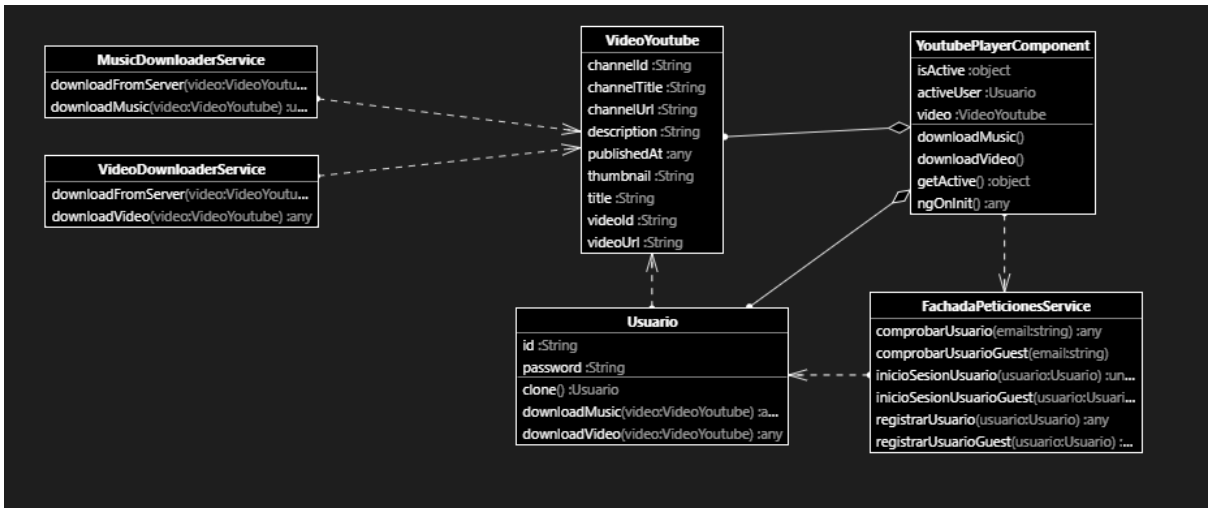
Para el módulo de youtube lo dividimos en servicios, youtube-player y youtube-searcher.

Dentro de los servicios se encuentra un patrón de fachada para la descarga multimedia, junto a las clases específicas de los servicios para la descarga de vídeos y de música. Ambas clases de servicio tienen las mismas funciones, siendo estas downloadFromServer y downloadMusic, si es el servicio de Musica, y downloadVideo, si es el servicio de Video.



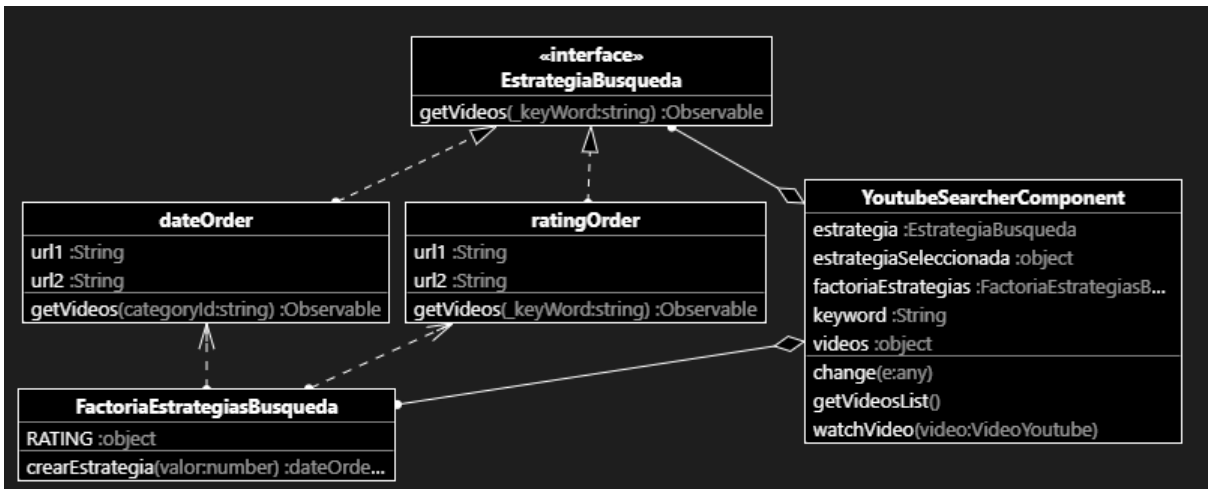
Por otro lado está el conjunto de youtube-player, el cual contiene un único componente con el nombre de YoutubePlayerComponent, que contendrá al usuario con sesión iniciada, el video que se reproducirá, y un campo para ver si está activo. Además de esto, tendrá las funciones principales para descargar el video que se esté visualizando como música (mp3) o descargarlo como vídeo (mp4), las cuales serán válidas únicamente si el usuario es de

tipo administrador, para lo que se recurrirá a la fachada de peticiones, de forma que compruebe si es o no administrado.

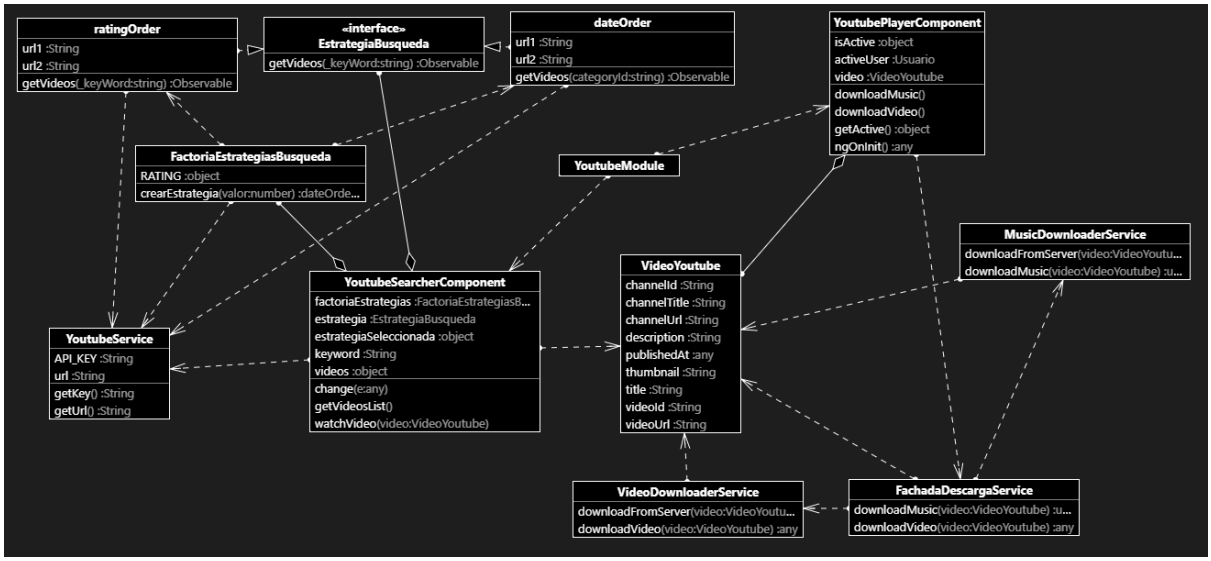


Por último, tenemos el youtube-searcher, el cual contiene la clase YoutubeSearcherComponent, encargada de gestionar el tipo de búsqueda según el input introducido en el campo. Además, se encuentran contenidas también un factory method para la creación de estrategias, junto a las estrategias que se aplicarán a la búsqueda.

Las clases dateOrder y ratingOrder serán las dos estrategias de búsqueda a implementar, las cuales extienden la funcionalidad de la interfaz EstrategiaBusqueda. Esto será escogido por el usuario en tiempo de ejecución mediante un menú desplegable.



Finalmente, se muestra el conjunto total del modelo, con todas las conexiones entre clases (de forma interna al módulo), en este diagrama.



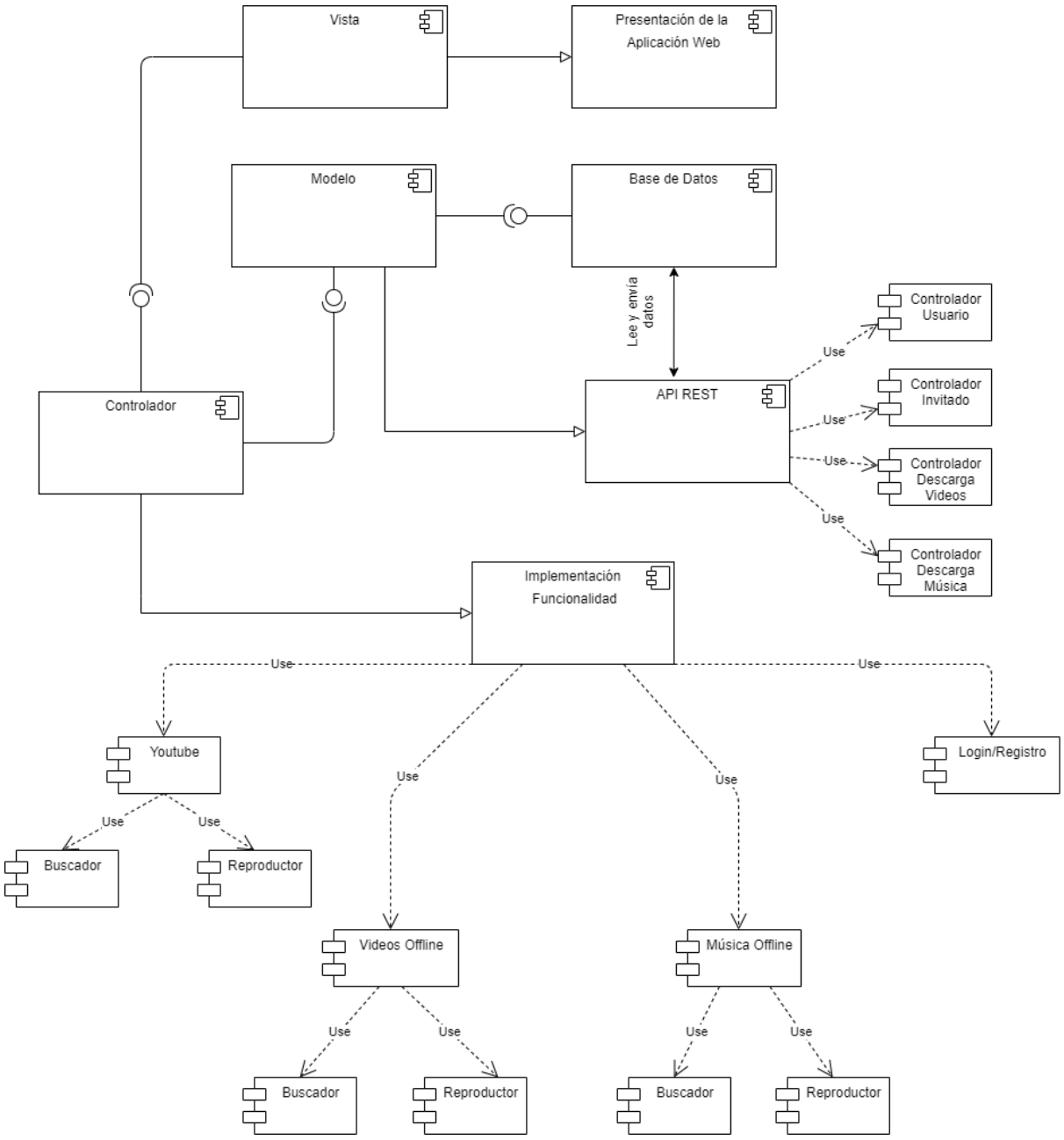
4.DIAGRAMA DE COMPONENTES

El modelo desarrollado anteriormente está basado en el patrón de diseño estructural MVC, por ello dispondrá de tres componentes.

- **Modelo**: Es la representación de la información con la cual el sistema opera, por lo tanto gestiona todos los accesos a dicha información, tanto consultas como actualizaciones, implementando también los privilegios de acceso que se hayan descrito.
- **Vista**: Presenta el 'modelo' (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario), por tanto requiere de dicho 'modelo' la información que debe representar como salida.
- **Controlador:** capa que controla todo lo que puede realizar nuestra aplicación. Responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

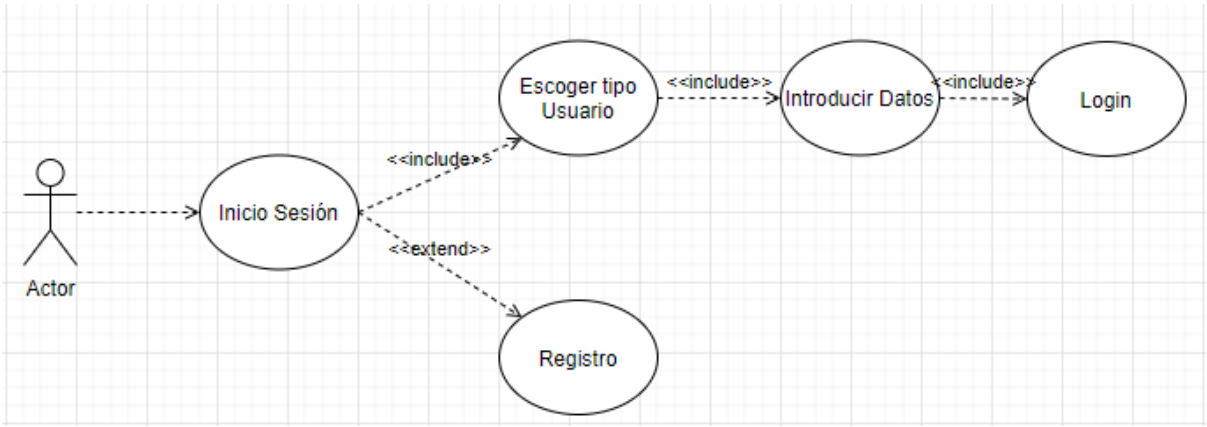
Implementación: Según las definiciones anteriores, se ha dividido la aplicación en los tres niveles definidos.

- Modelo: al ser el componente directamente conectado con los datos, estará compuesto por la base de datos y la API REST que se encargará de realizar las consultas sobre la base de datos (tanto para devolver información, insertarla o modificarla)
- Vista: al ser el componente de visualización por el usuario, está representado por el front-end de la aplicación web, haciendo referencia por lo tanto, la parte de los componentes Angular desarrollados en html5 y css .
- Controlador: al ser la base de la funcionalidad, constará de las clases de funcionalidad interna de la aplicación desarrolladas en Angular. Constará de varios módulos (haciendo referencia a cada una de las aplicaciones) con sus respectivas clases (para el reproductor, buscador…)

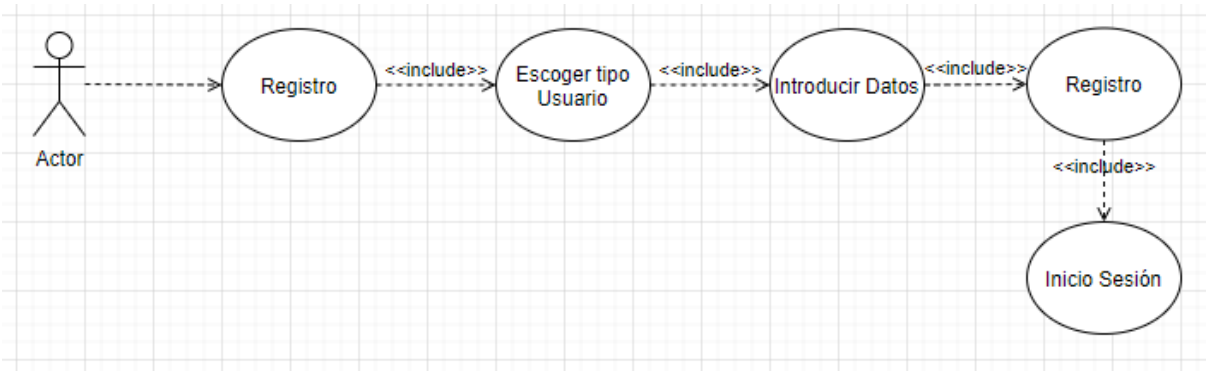

Vista
Presentación de la Aplicación Web
Modelo
Base de Datos
Lee y envía datos
Controlador
API REST
Controlador Usuario
Controlador Invitado
Controlador Descarga Videos
Controlador Descarga Música
Use
Implementación Funcionalidad
Youtube
Buscador
Reproductor
Videos Offline
Música Offline
Login/Registro

5.DIAGRAMAS DE CASOS DE USO

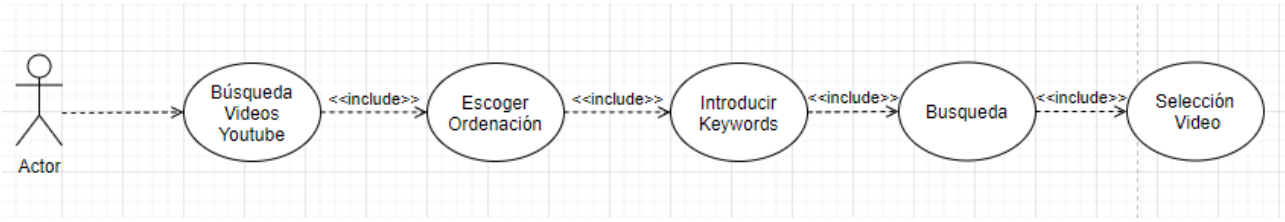
5.1 Inicio de sesión



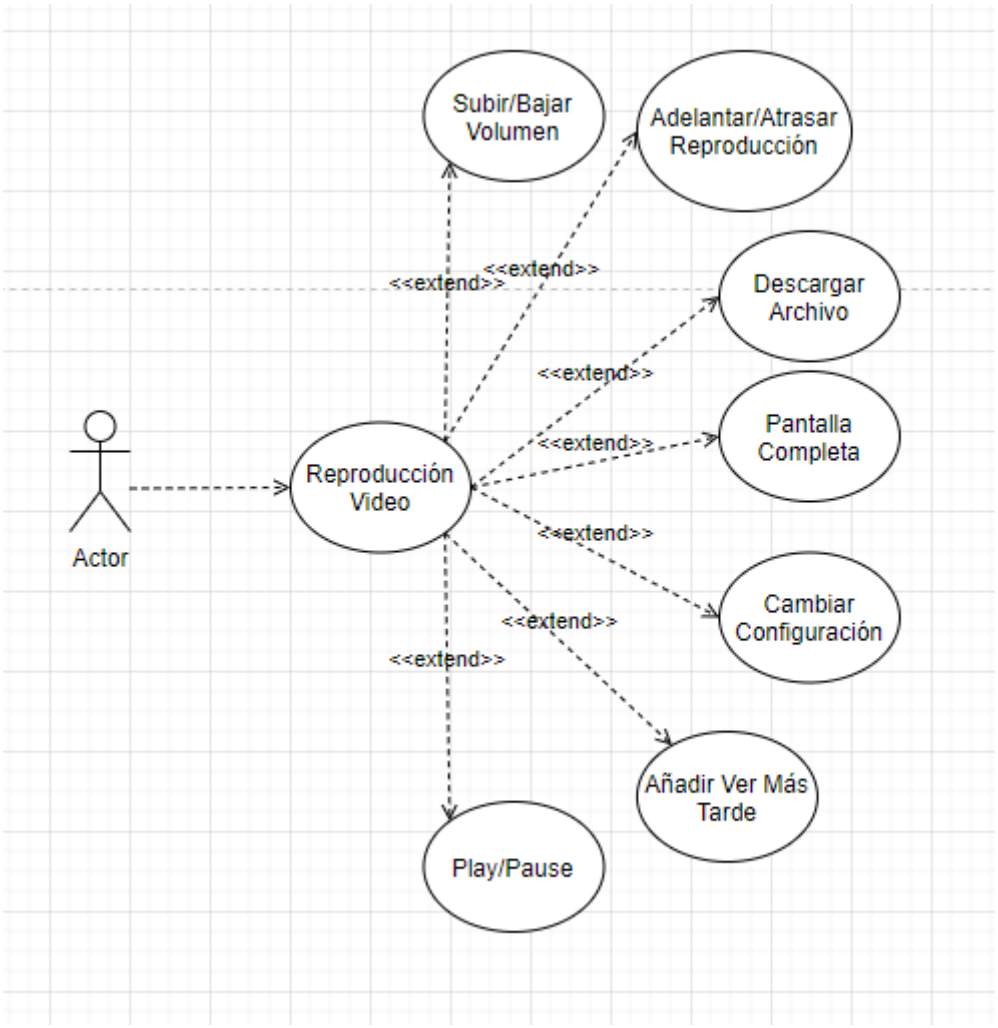
5.2 Registro de usuario



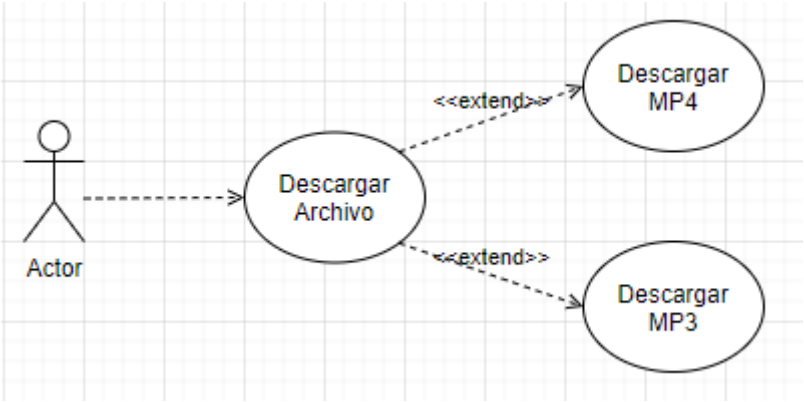
5.3 Búsqueda de videos



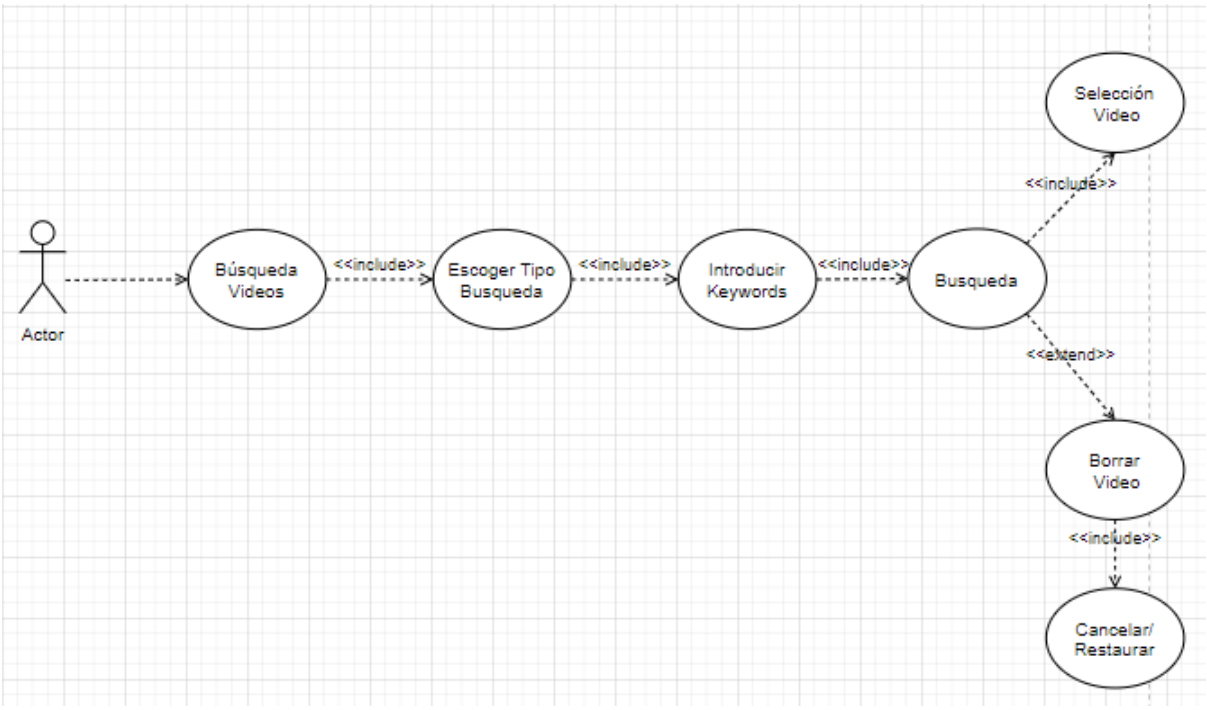
5.4 Reproducción de video en Youtube



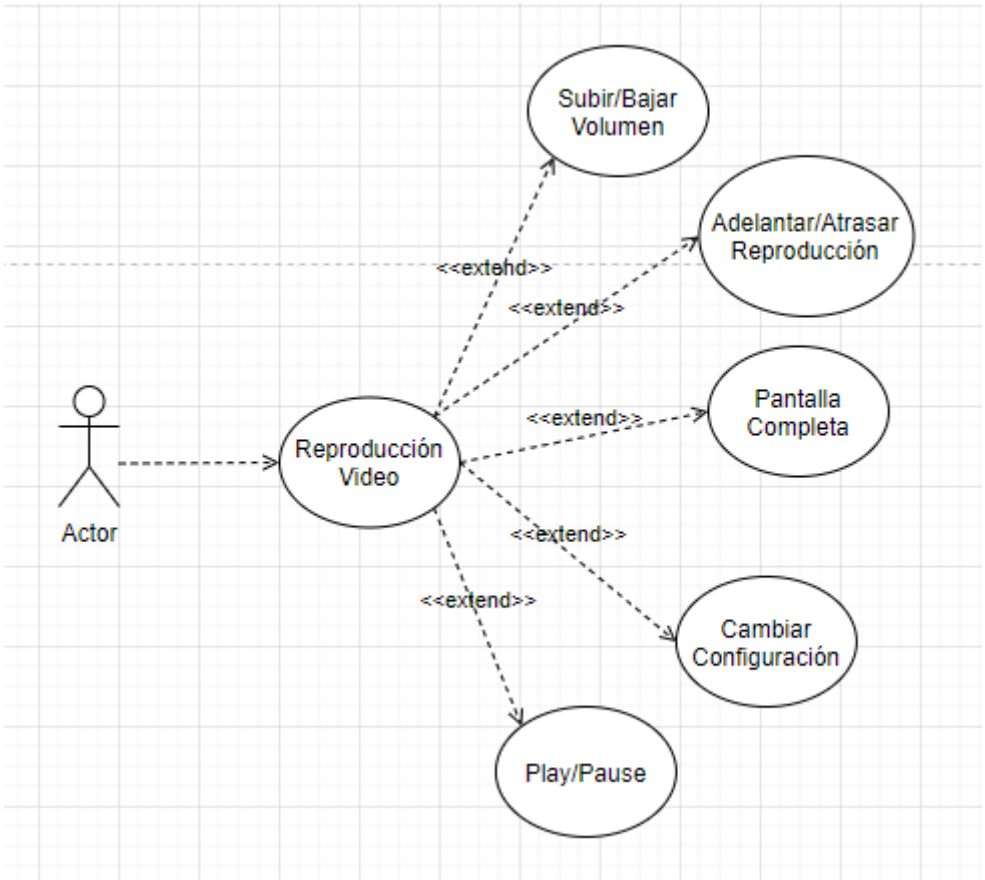
5.5 Descarga de archivo



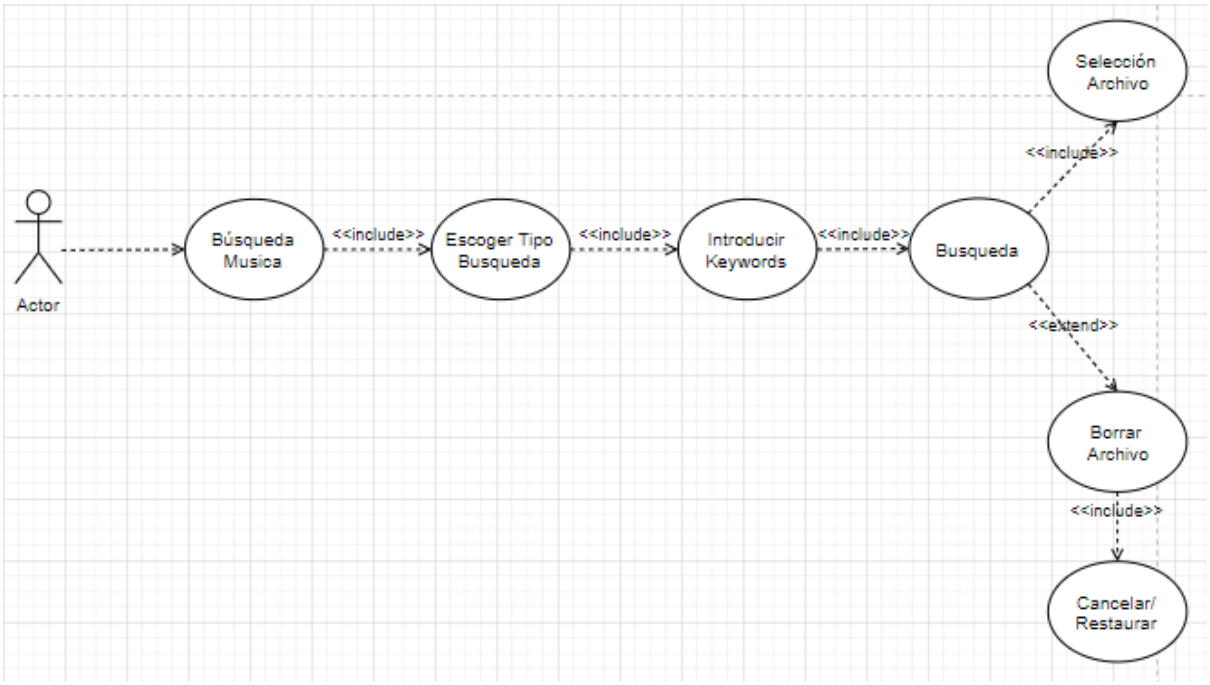
5.6 Búsqueda de video offline



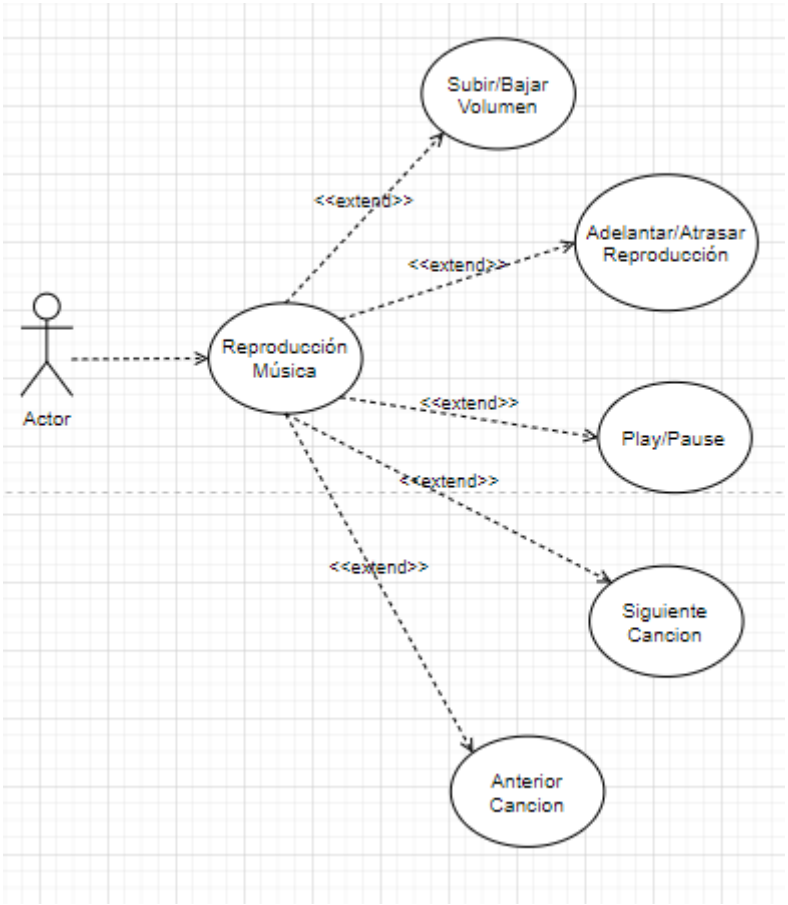
5.7 Reproducción de video offline



5.8 Búsqueda de música offline

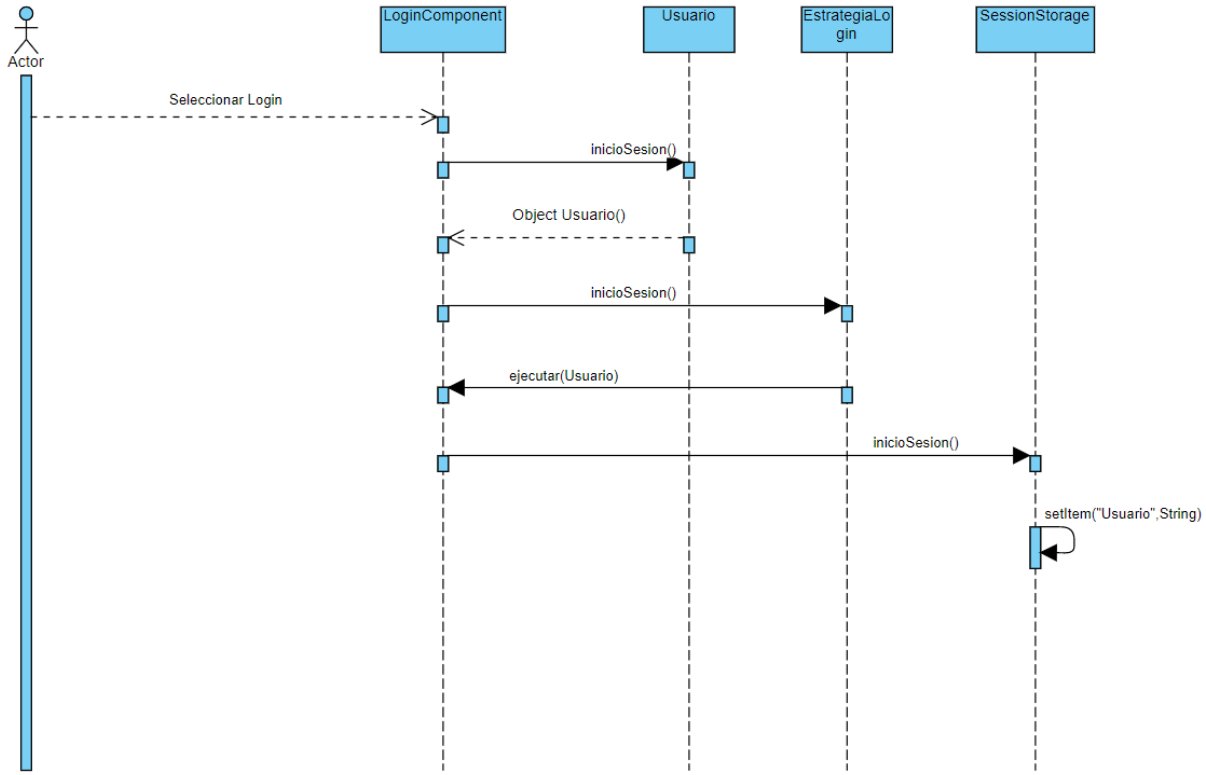


5.9 Reproducción de música offline

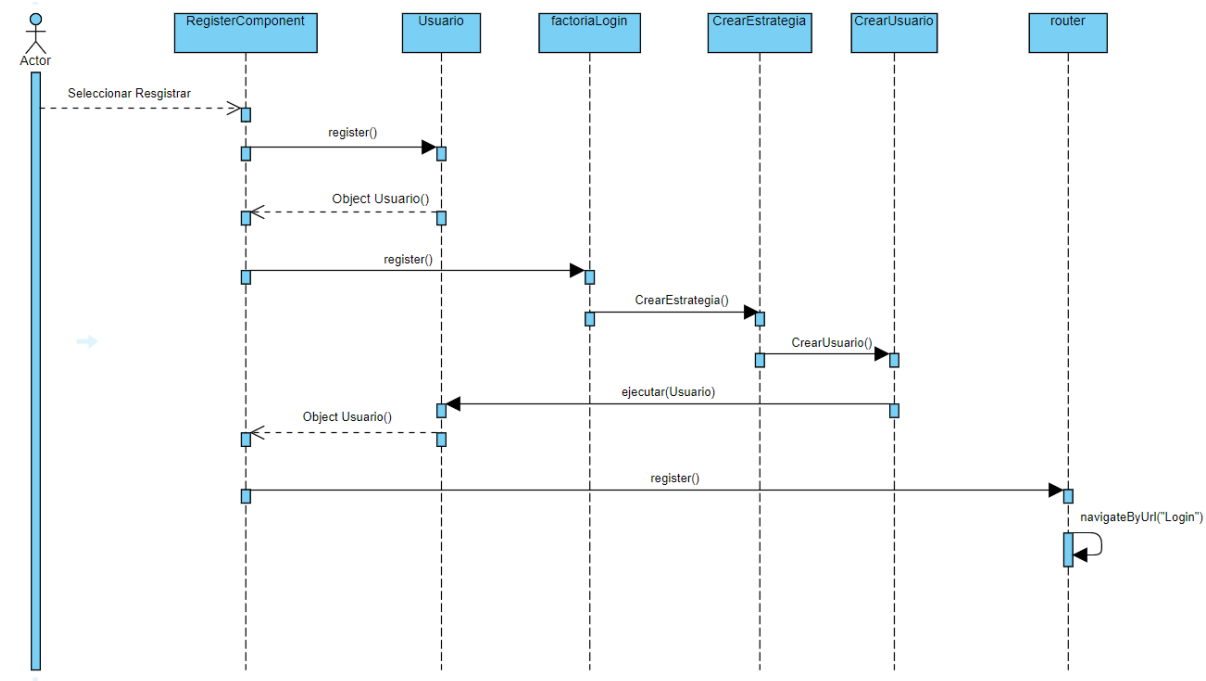


6.DIAGRAMAS DE SECUENCIA

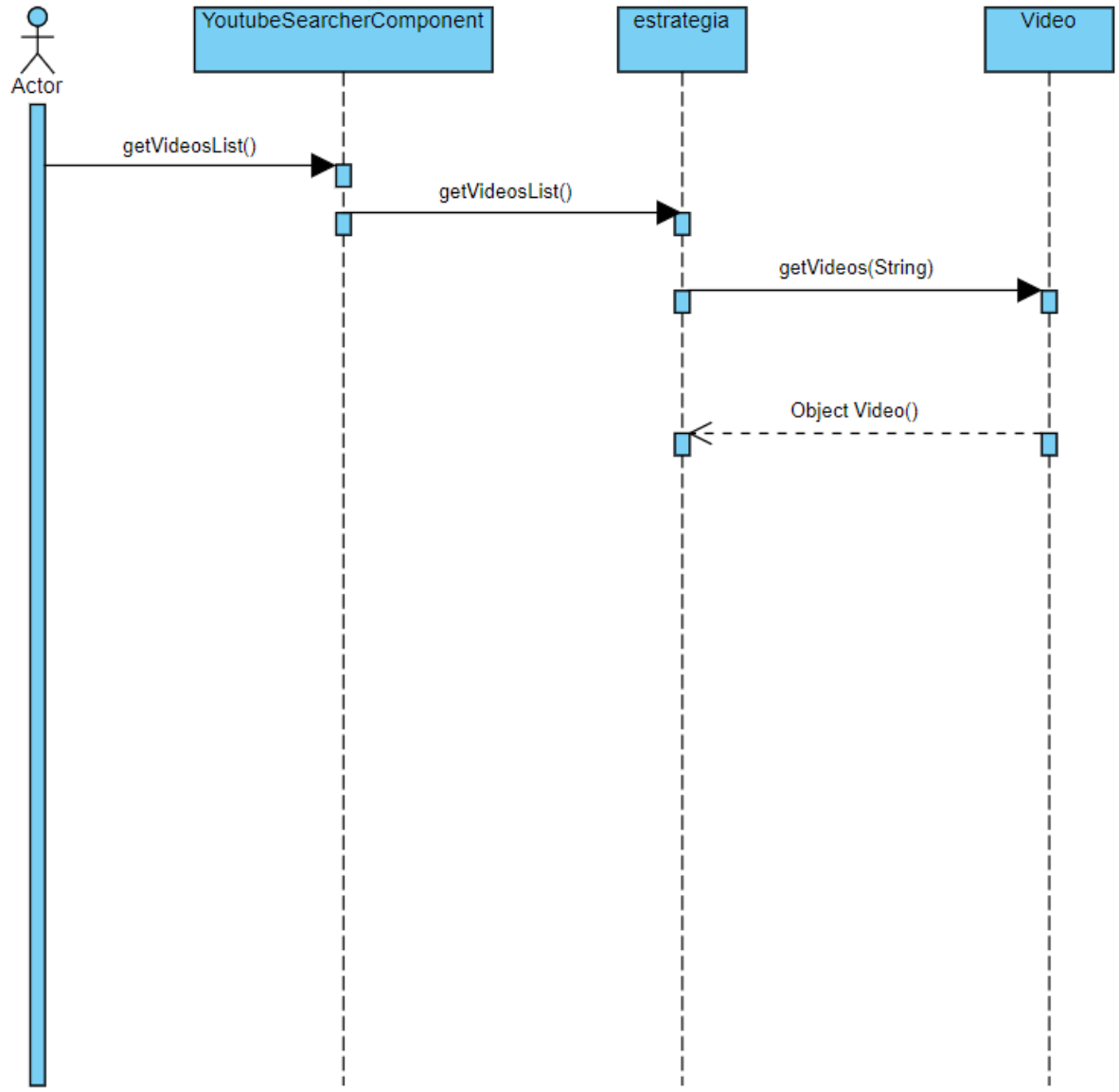
6.1 Inicio de sesión



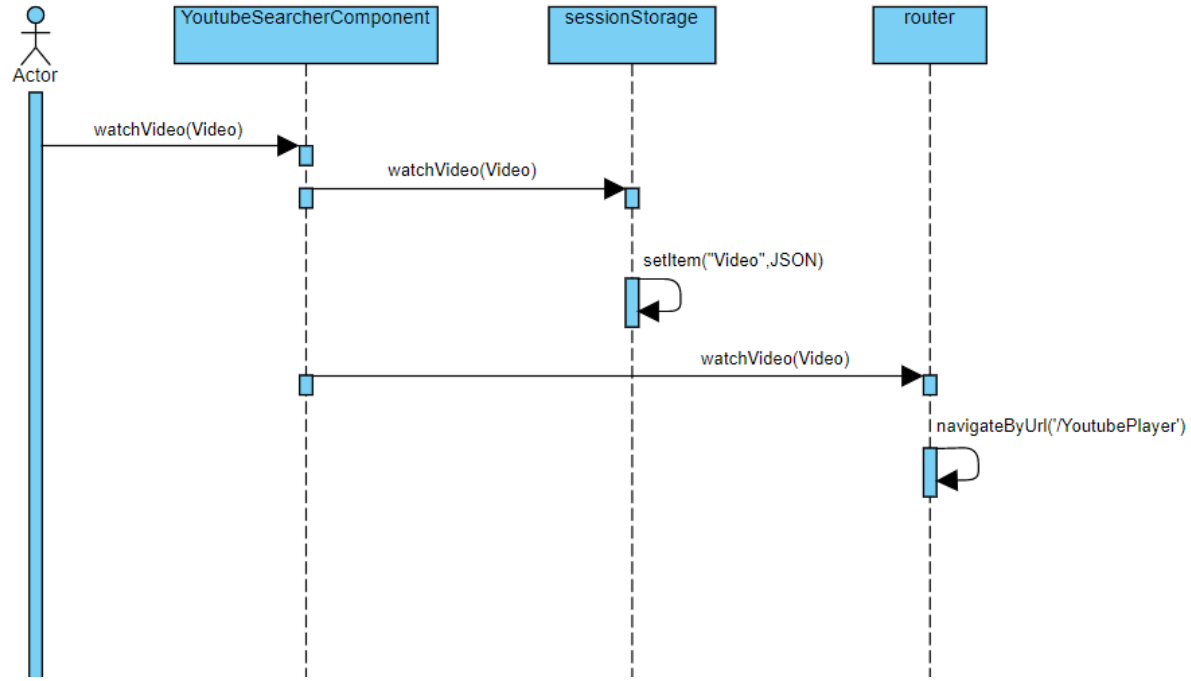
6.2 Registro de usuario



6.3 Búsqueda de videos Youtube

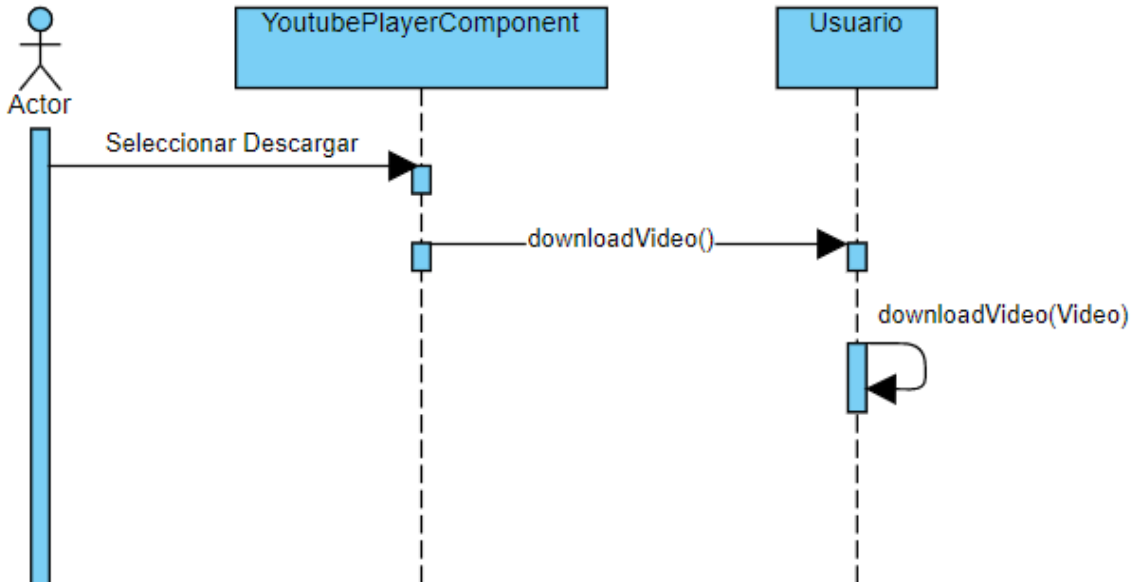


6.4 Reproducción de video Youtube

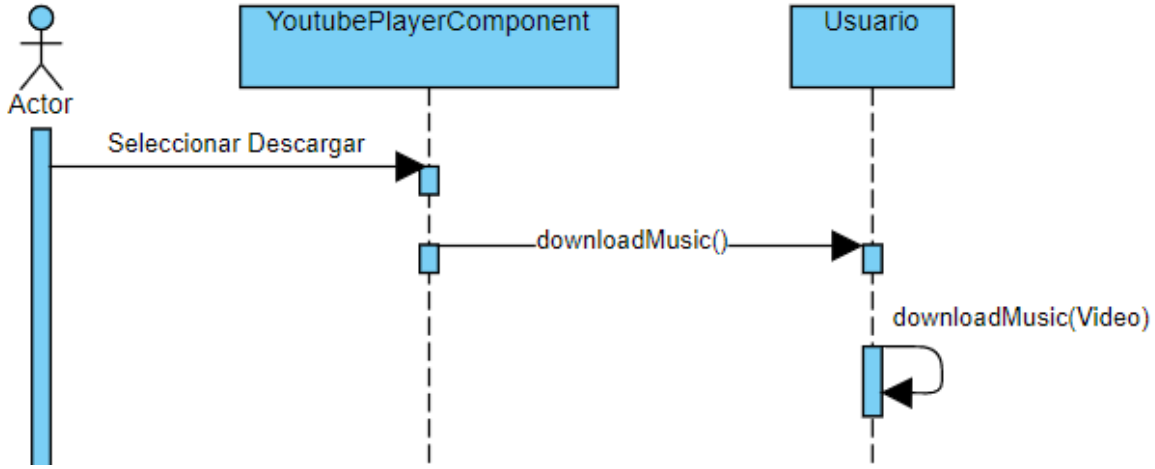


6.5 Descarga de archivo

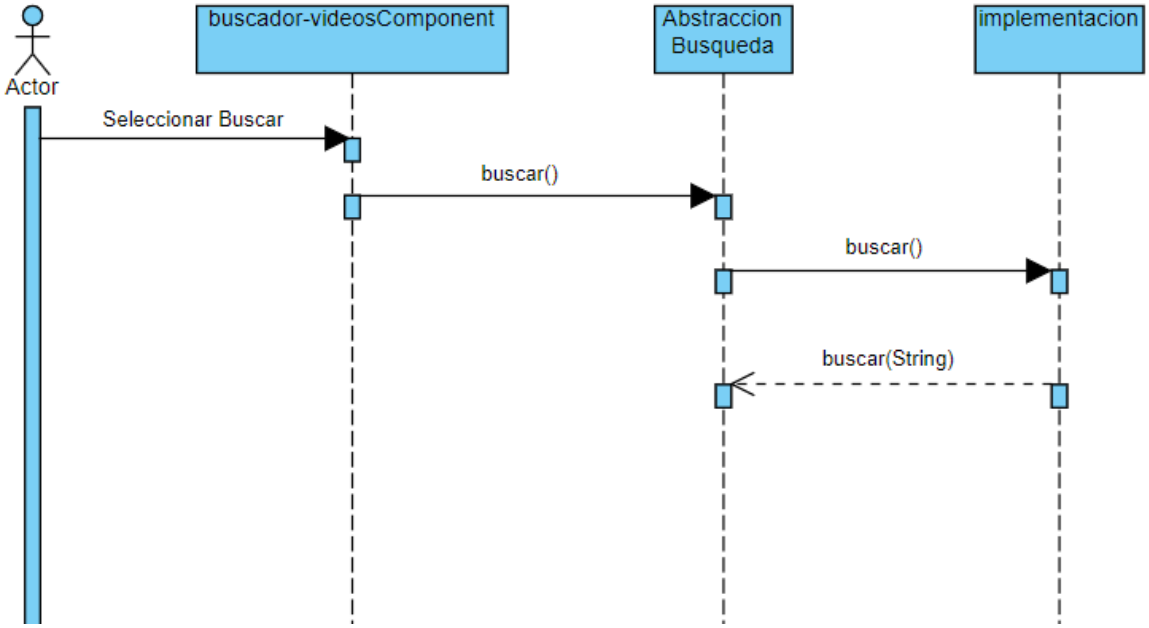
Descarga de Video



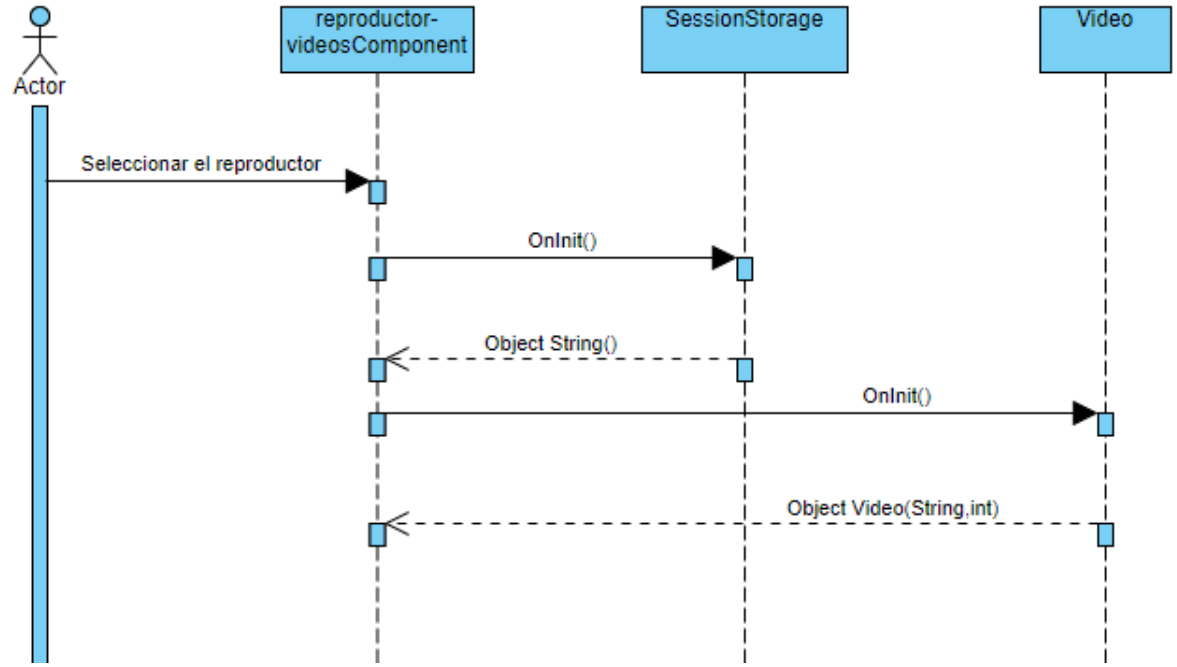
Descarga de Música



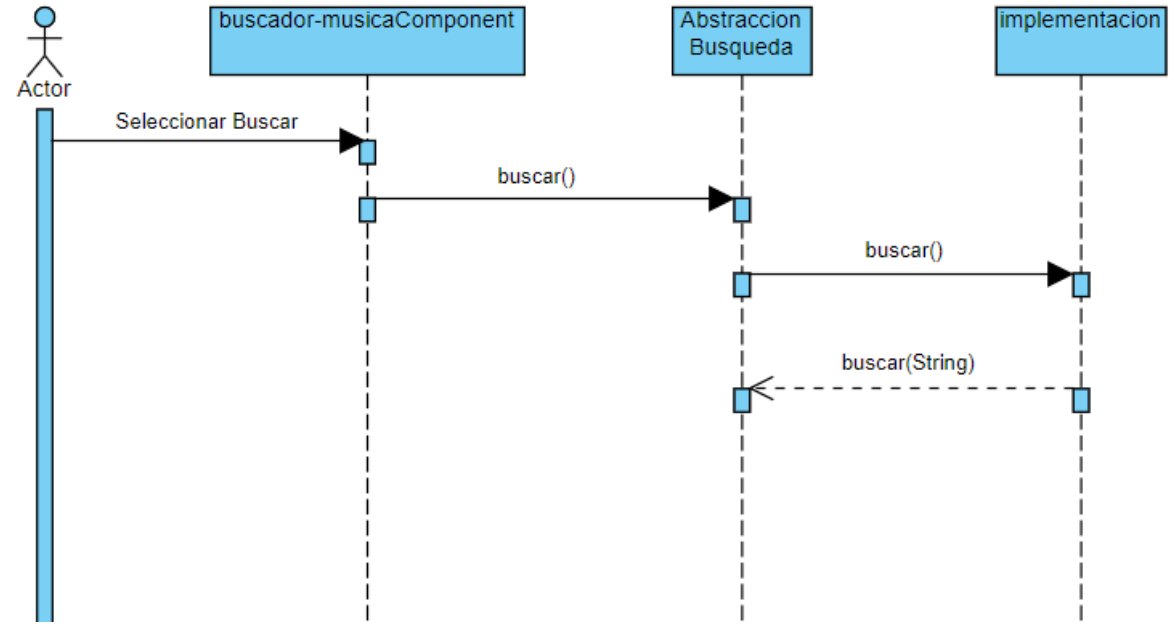
6.6 Búsqueda de video offline



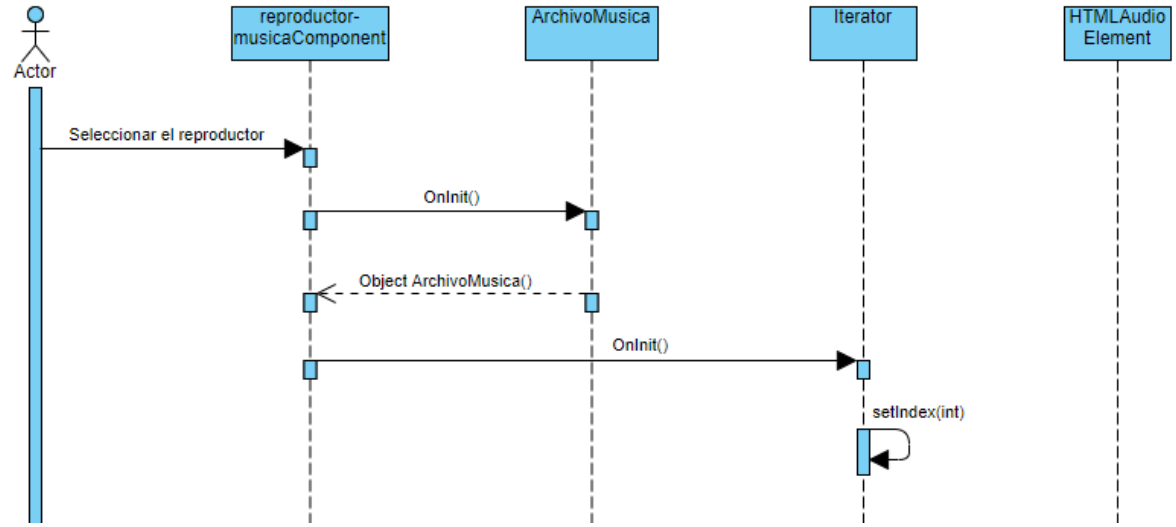
6.7 Reproducción de video offline



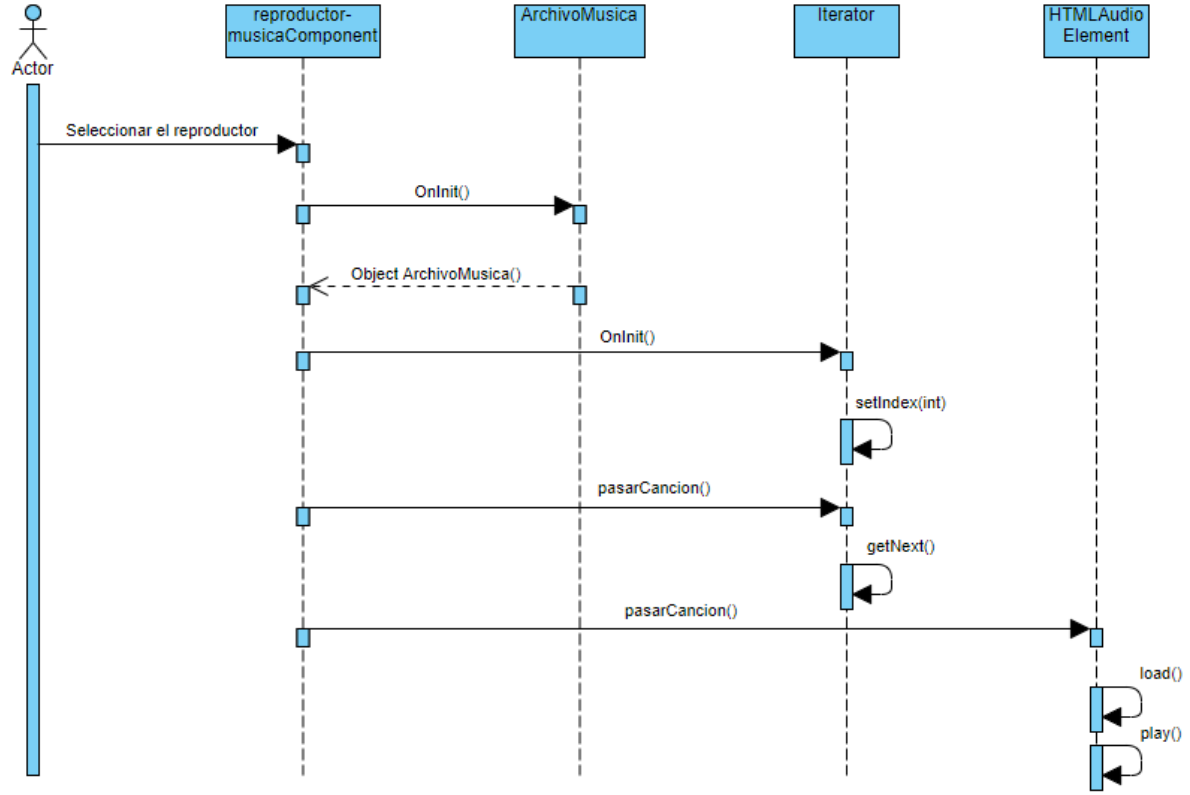
6.8 Búsqueda de música offline



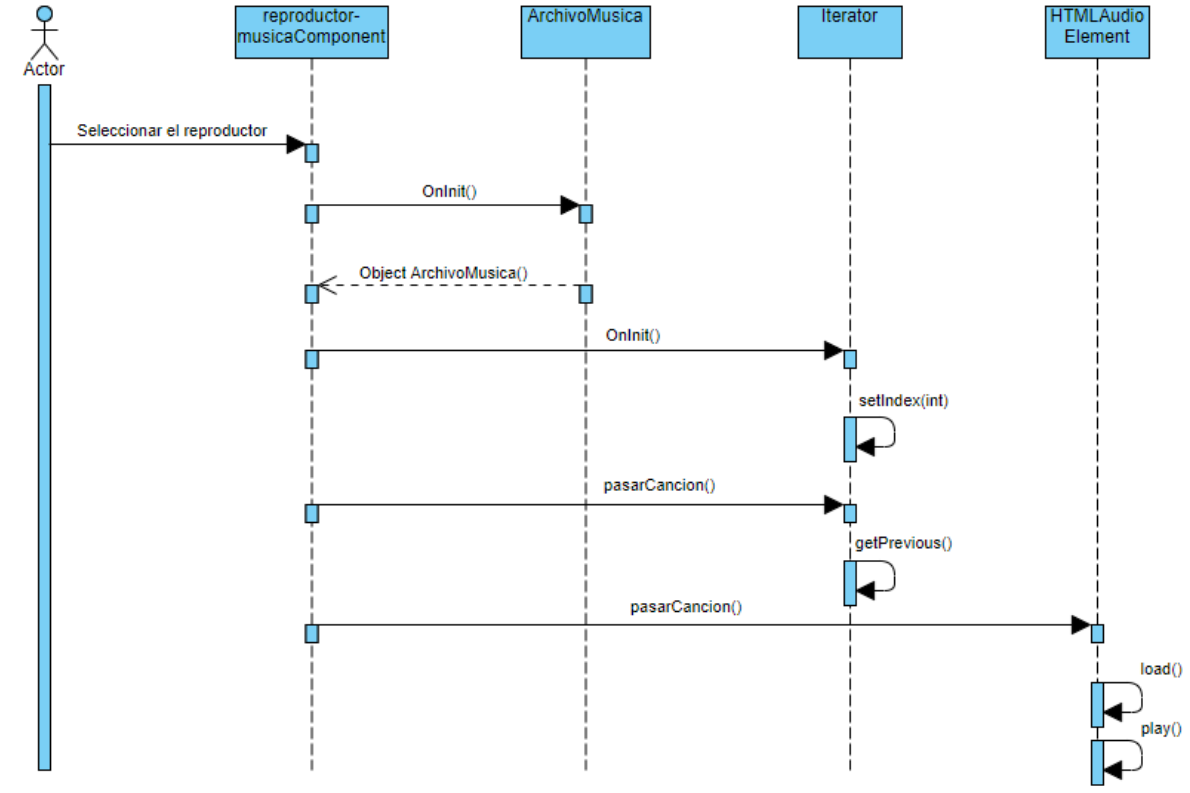
6.9 Reproducción de música offline



Siguiente Canción



Canción Anterior



7.MANUAL DE USUARIO

BASE DE DATOS

Esta aplicación Web utiliza una base de datos MySql, para almacenar los dos tipos de usuario del sistema, administradores e invitados.
El servidor utilizado es el servidor predeterminado por MySql, con el usuario root sin contraseña.
Para la creación de la base de datos se siguen los siguientes pasos:

- mysql -h localhost -u root -p

Una vez iniciado sesión, se utiliza el script .sql ofrecido, “database.sql”, que contiene los siguientes comandos:

- CREATE DATABASE App_Hubb;
- USE App_Hubb;
- CREATE TABLE user (

  email NVARCHAR(50) NOT NULL PRIMARY KEY,

  password NVARCHAR(20)

  );

- ALTER TABLE user

  ADD CONSTRAINT my_constraint CHECK(user.email LIKE '%_@gmail.com');

- CREATE TABLE guest (

  username NVARCHAR(25) NOT NULL PRIMARY KEY,

  password NVARCHAR(20)

  );

API REST

Adicionalmente, se ha creado una API REST que contemple varias funcionalidades:

1. Conexión directa con la base de datos, para realizar aquellas operaciones que la involucren
2. Download de archivos, funcionalidad por la cual se descargan archivos de tipo .mp4 de youtube y se guardan los archivos en el sistema (convirtiéndolos en archivos .mp3 en caso de ser archivos de sonido)

    - Utiliza el módulo youtube-dl para la descarga de los videos de youtube
    - Utiliza el módulo fs para guardar los archivos en el sistema

Esta aplicación se ha desarrollado en NODE.js, el cual se puede obtener de manera gratuita desde https://nodejs.org/es/
Una vez instalado correctamente con sus variables de entorno, es necesario instalar por terminal las librerías necesarias para el correcto funcionamiento, se realizará con los siguientes comandos:

- npm i mysql
- npm i express morgan promise-mysql cors
- npm install -g typescript
- npm i nodemon -D
- npm i @types/express -D
- npm i promise-mysql@3.3.1
- npm install youtube-dl

Una vez instaladas se inicia el servidor con el comando:

- npm run dev

El cual iniciará un proceso que estará escuchando las peticiones en el puerto 3000.
Este servidor funciona con peticiones https, devolviendo respuestas en formato json. Sus rutas de comunicación posible son:

- localhost/api/users
    - GET '/' : devuelve una lista de todos los usuarios en la base de datos
    - GET '/email' : devuelve las características del usuario con el email especificado
    - GET '/exists' : comprueba si existe el usuario con el email especificado
    - GET '/login' : comprueba si el usuario y contraseña son correctos y devuelve el token de autenticación
    - POST '/' : se inserta el usuario
    - PUT '/email' : se actualiza el usuario con el email especificado
    - DELETE '/email' : se borra el usuario con el email especificado
- localhost/api/guest
    - GET '/login' : devuelve una lista de todos los usuarios en la base de datos
    - GET ''/username' : devuelve las características del usuario con el username especificado
    - POST '/' : se inserta el usuario
    - PUT '/username', : se actualiza el usuario con el username especificado
    - DELETE '/username' : se borra el usuario con el username especificado
- localhost/download
    - GET /video: descarga el archivo .mp4, lo guarda en el sistema y devuelve el estado final de la descarga.
    - GET /music: descarga el archivo .mp4, lo convierte a .mp3 y lo guarda en el sistema y devuelve el estado final de la descarga.

      Debido a las políticas de youtube, hay videos que no ofrecen enlaces de descarga, por lo que hay videos que no se pueden descargar (está contemplado dentro del funcionamiento, devolviendo el estado final correspondiente).

Aplicación WEB

La aplicación WEB desarrollada se ha implementado con el framework de Angular, para esto, se instala con el siguiente comando:

- npm install @angular/cli -g

Una vez instalado, se instalan las librerías correspondientes al correcto funcionamiento del proyecto:

- npm install @angular/cli -g
- npm install boostrap
- npm install bootstrap jquery @popperjs/core
- npm install @angular/youtube-player
- npm i ng-connection-service
- npm i --save-dev @types/lodash
- ng add @ng-bootstrap/ng-bootstrap

Tras la correcta instalación de todas las librerías anteriores, se ejecuta la aplicación web con el siguiente comando:

- ng serve --live-reload false

Una vez iniciada, se ejecutará en localhost:4200, válido en cualquier navegador disponible.
Una vez especificada la ruta, al no esta logueados nos llevará al Login de la aplicación (en caso de estarlo, nos llevará por defecto al módulo de Youtube), donde se pueden hacer dos cosas:

- Registro de usuario
- Login

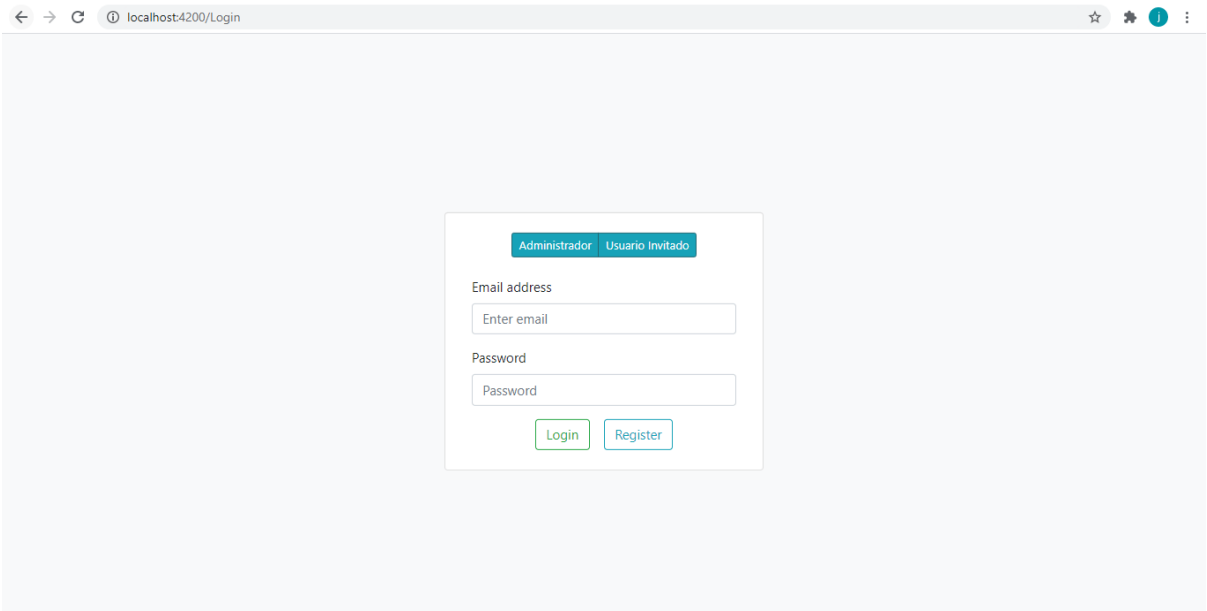
Al no estar logueados, no nos permitirá visitar cualquier funcionalidad adicional.

Login

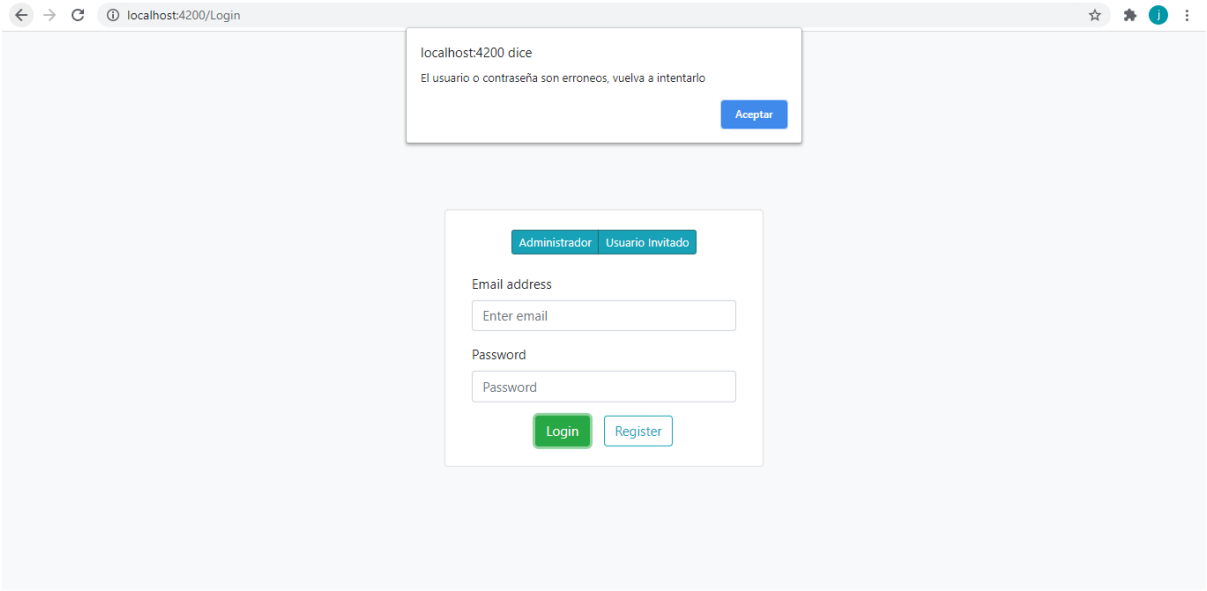
En este apartado hay dos opciones posible:

- Login como administrador
- Login como usuario invitado

Esto se realiza seleccionando cualquiera de los dos botones superiores, pudiendo elegir de forma rápida cualquier forma de login.
Una vez introducidos los datos, se pulsará el botón de login, que nos permitirá el acceso en caso correcto.



En caso de haber introducido algún campo erróneo, se incluirá el siguiente mensaje de error

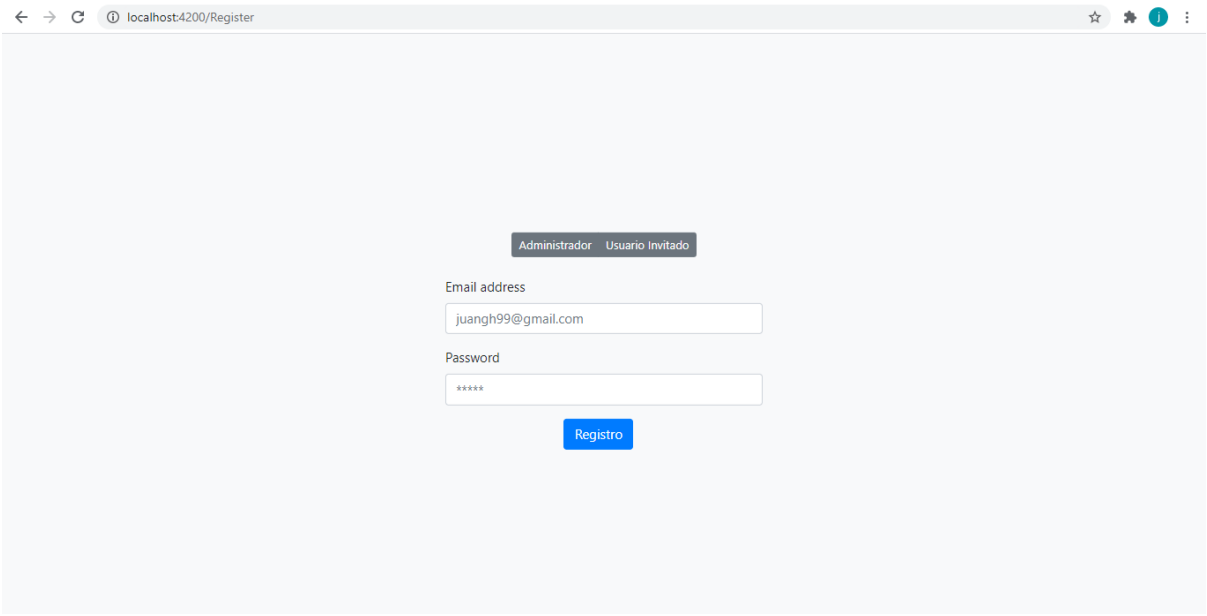


Adicionalmente, en caso de no tener un usuario registrado, se puede pulsar el botón registro, que nos llevará al formulario de registro.

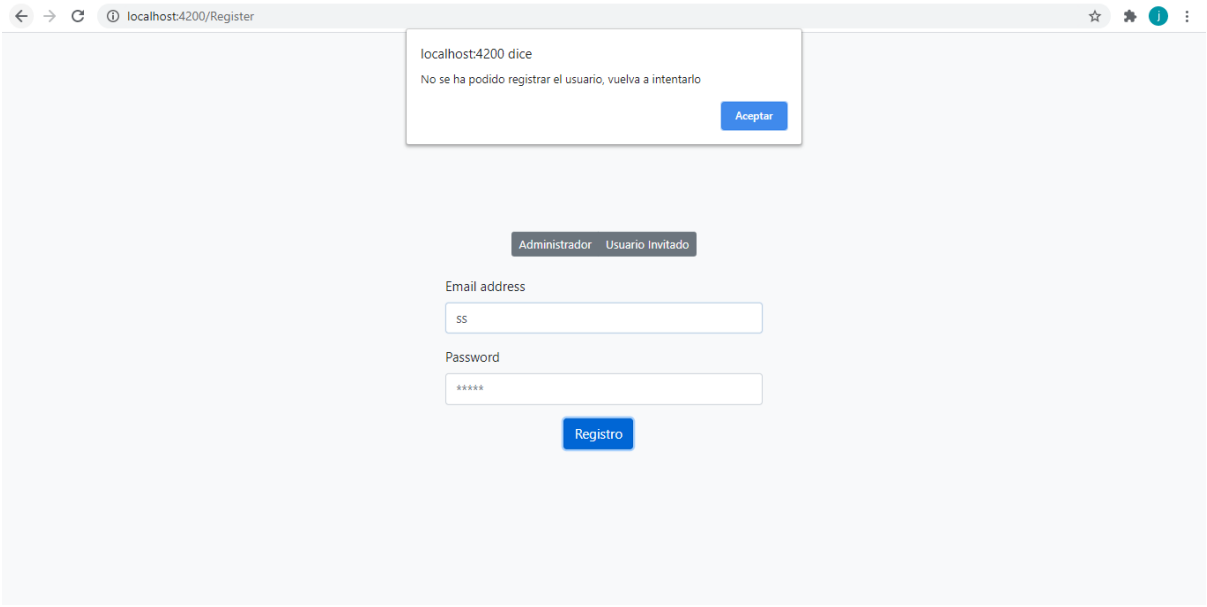
Registro de Usuario

En este apartado, al igual que en el login, se podrá registrar un usuario de ambos tipos, administrador e invitado.
Cada lección constará de campos de ejemplos a seguir, para poder dar de alta al usuario.
Una vez se introducen los campos se pulsa el botón de registro, el cual nos llevará de nuevo a la pantalla login para poder iniciar la aplicación.
Cabe destacar que

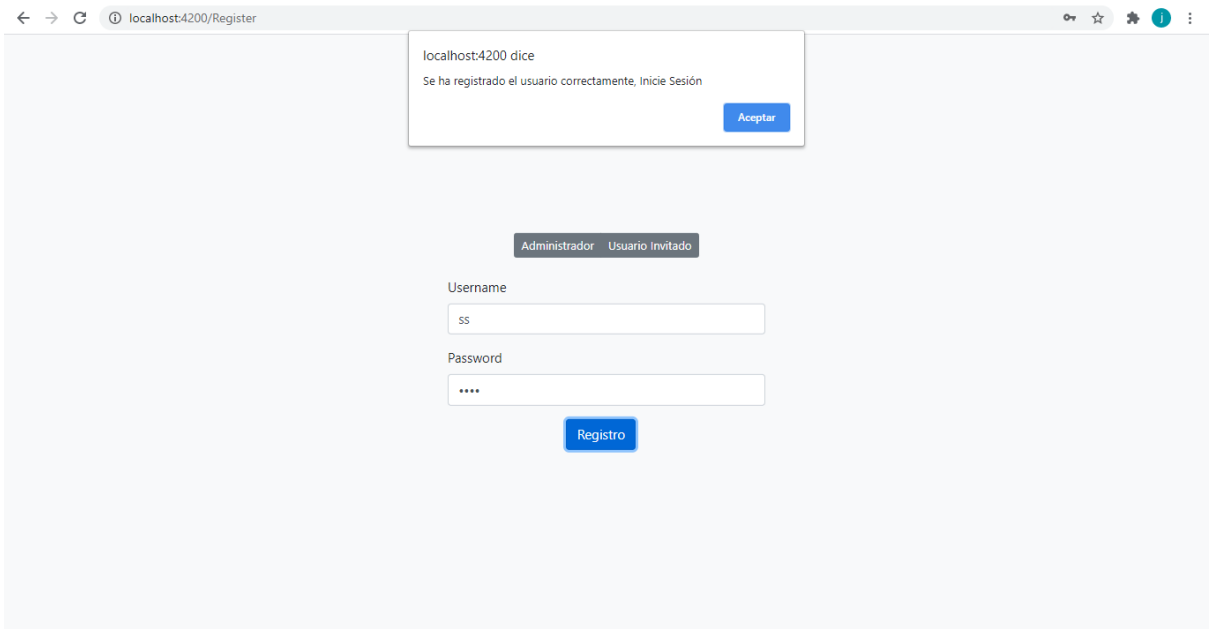
- Al elegir un usuario administrador, el email tendrá que ser de la forma “_@gmail.com”, en caso contrario dará error.
- Al elegir un usuario invitado, un nombre común de usuario será válido.



En caso de haber introducido datos erróneos se mostrará el mensaje de error

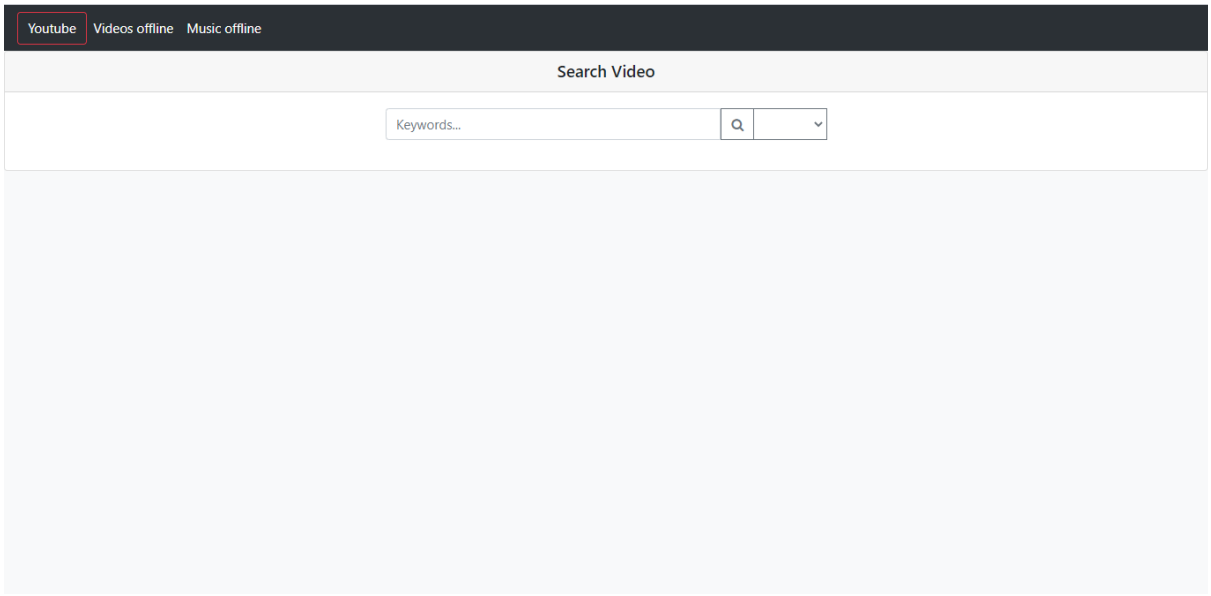


En caso de corrección

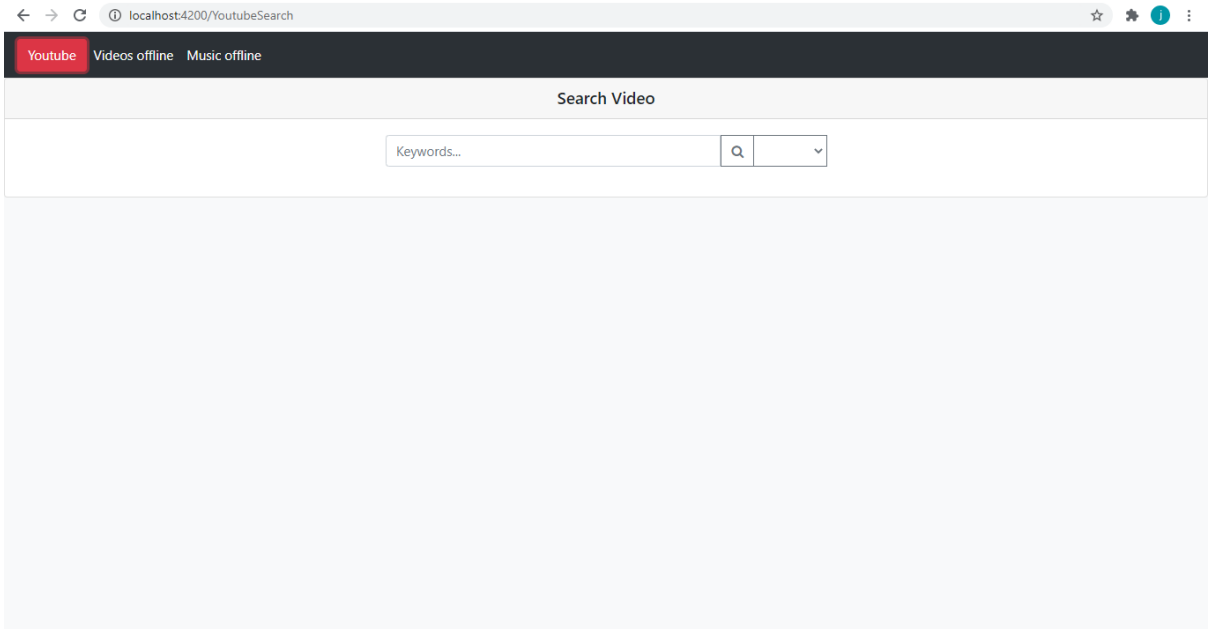


Búsqueda de Videos en Youtube

Una vez realizado un login correcto, se lleva automáticamente a la aplicación de Youtube, de forma que se presenta el buscador.

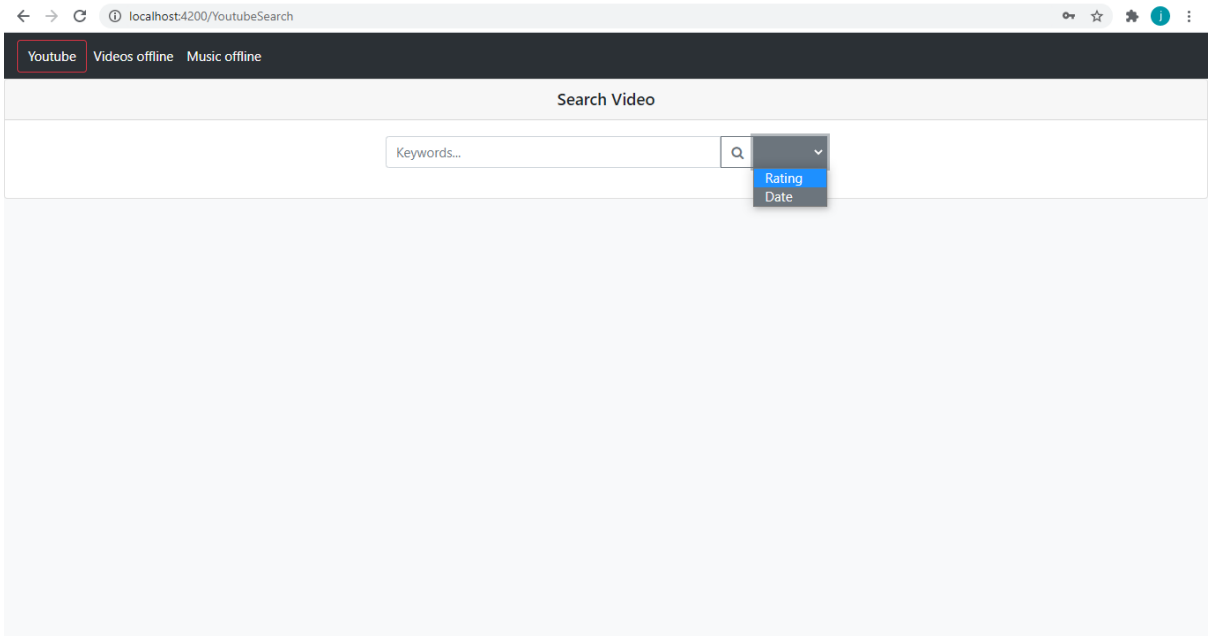


En caso de querer navegar desde otra aplicación del Hub, se utilizará el panel de navegación superior, pulsando el botón de la aplicación a la que queramos ir.

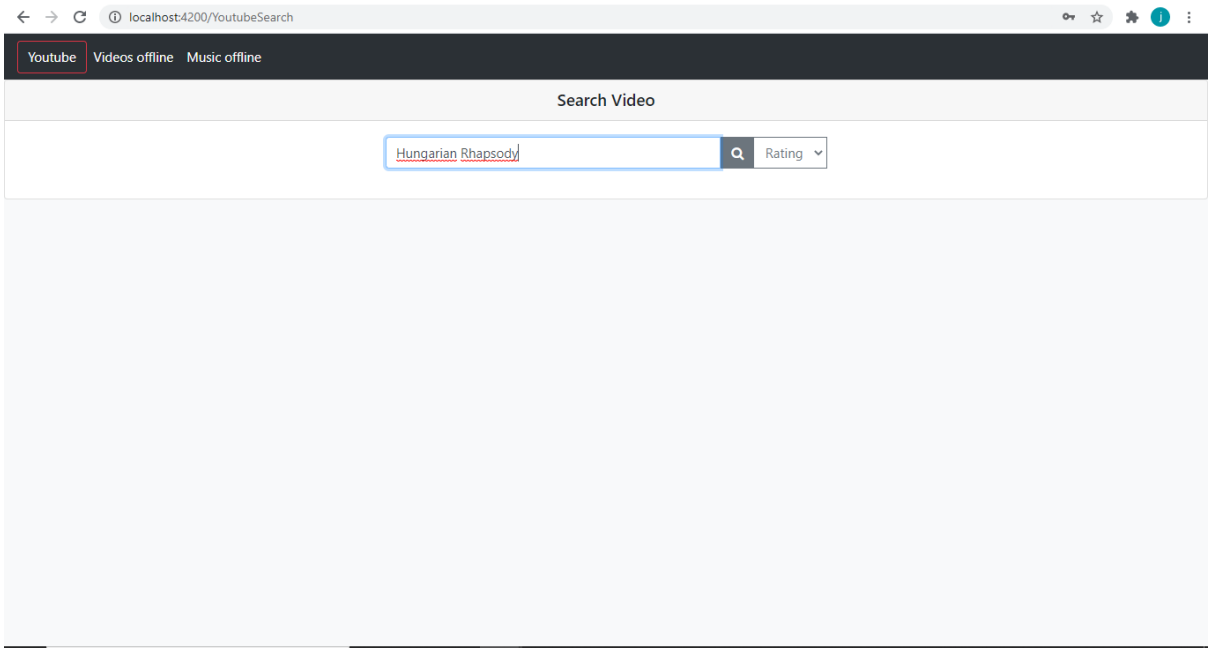


Hay dos opciones de búsqueda en este caso:

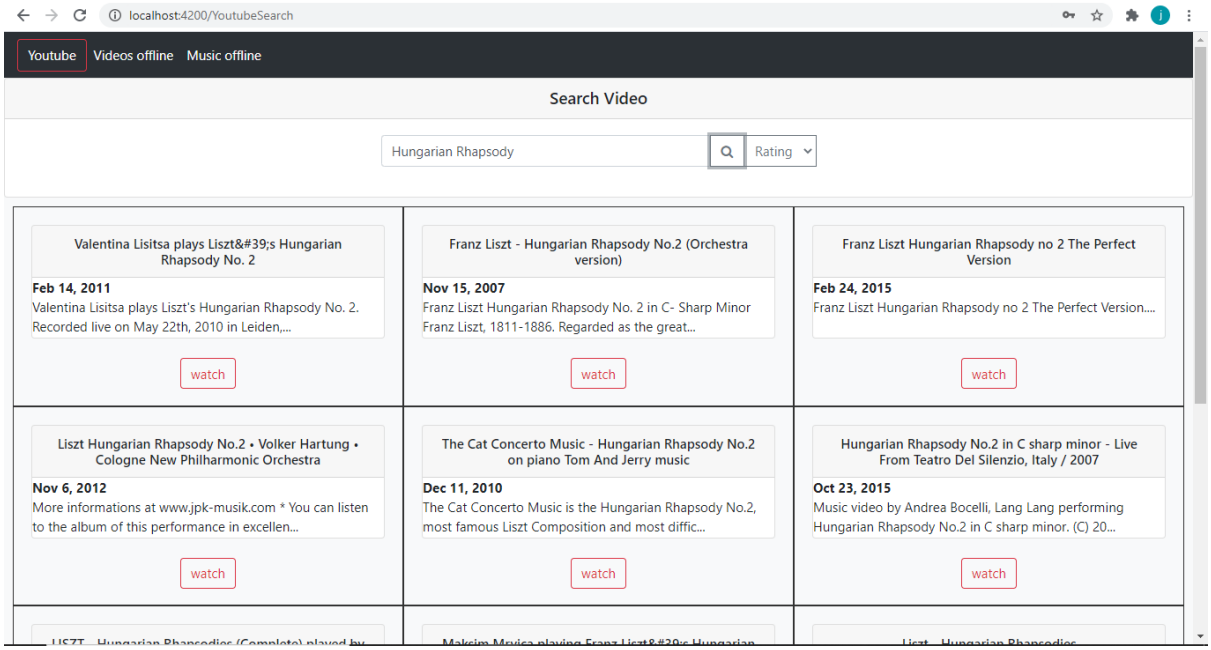
- Ordenación por Date
- Ordenación por Rate



Una vez seleccionada una opción, se introduce la keyword o keywords que se desean buscar y se pulsa el botón de búsqueda.

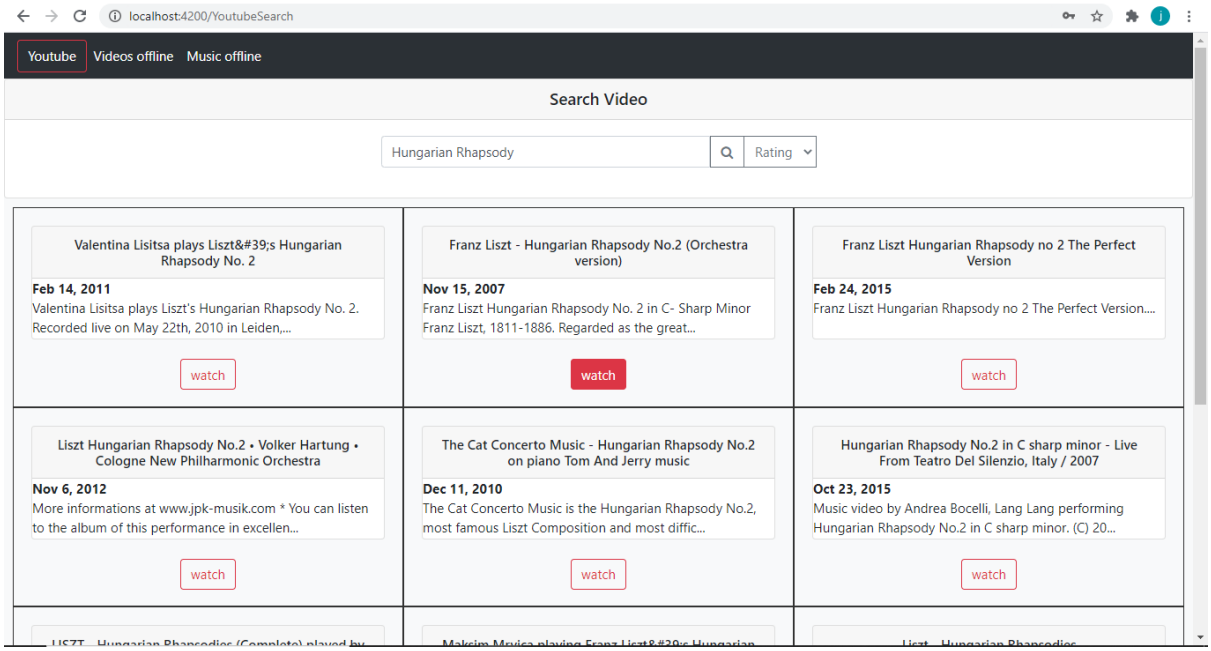


Una vez se realiza la búsqueda, aparecerán los vídeos relacionados con las palabras introducidas.

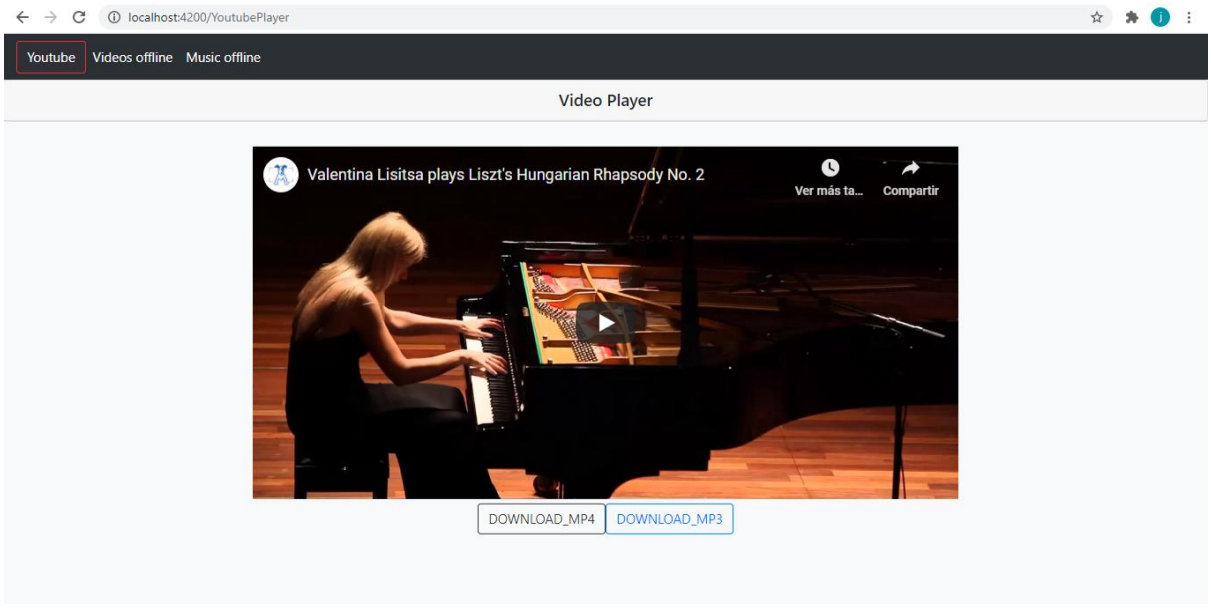


Reproducción de Videos en Youtube

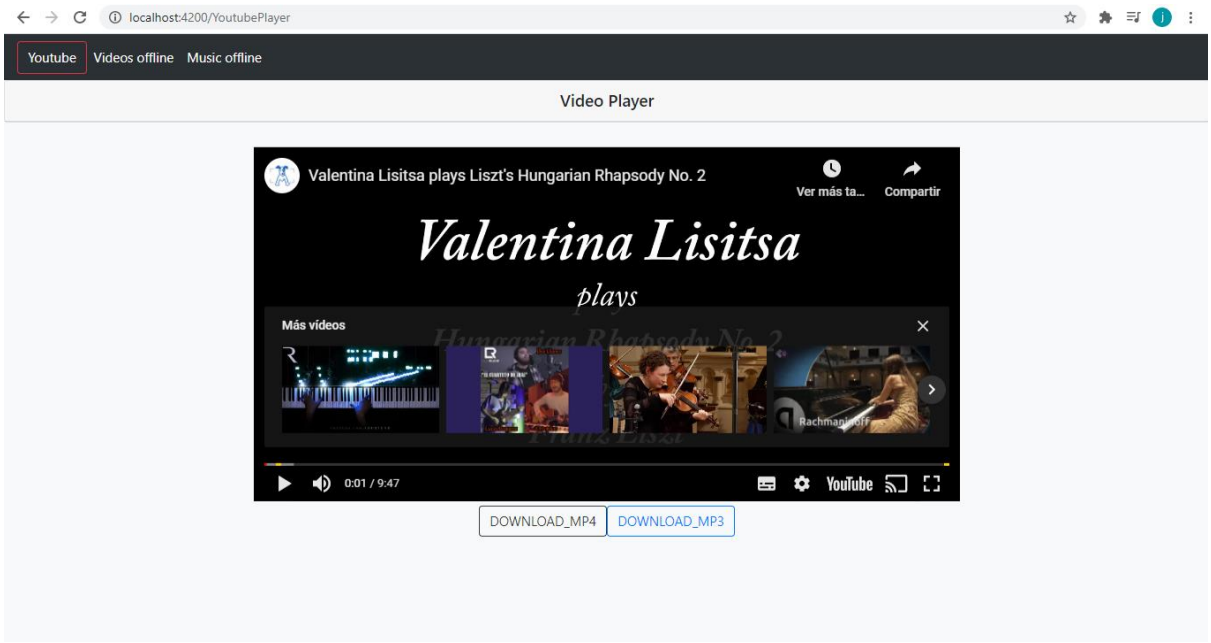
De esta lista de videos se podrá escoger uno para su visualización, pulsando el botón de watch.



Esto nos llevará a un reproductor de video de youtube, donde se podrá visualizar el video seleccionado.



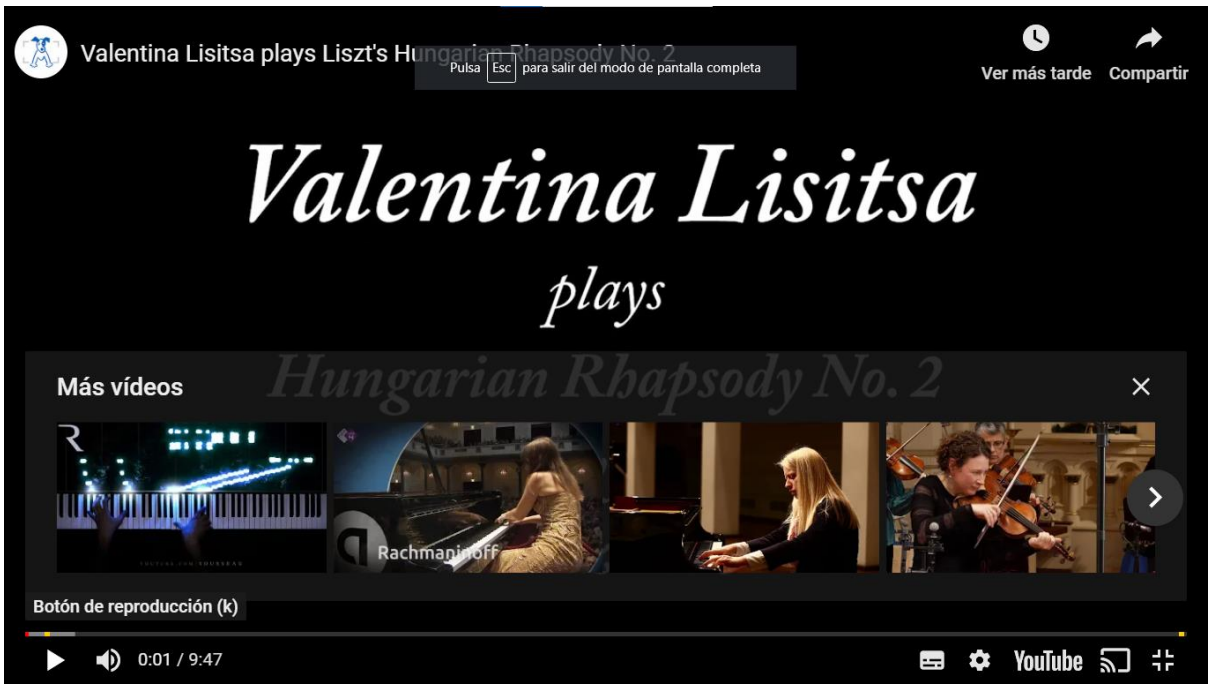
En este apartado, contaremos con todas las funcionalidades posibles que nos ofrece un reproductor de youtube.



Como se puede ver, se cuentan con los controles propios:

- Play/Pause
- Añadir a ver más tarde
- Bajar/Subir volumen
- Cambiar Configuración de reproducción
- Pantalla Completa
- Avanzar en el tiempo
- …

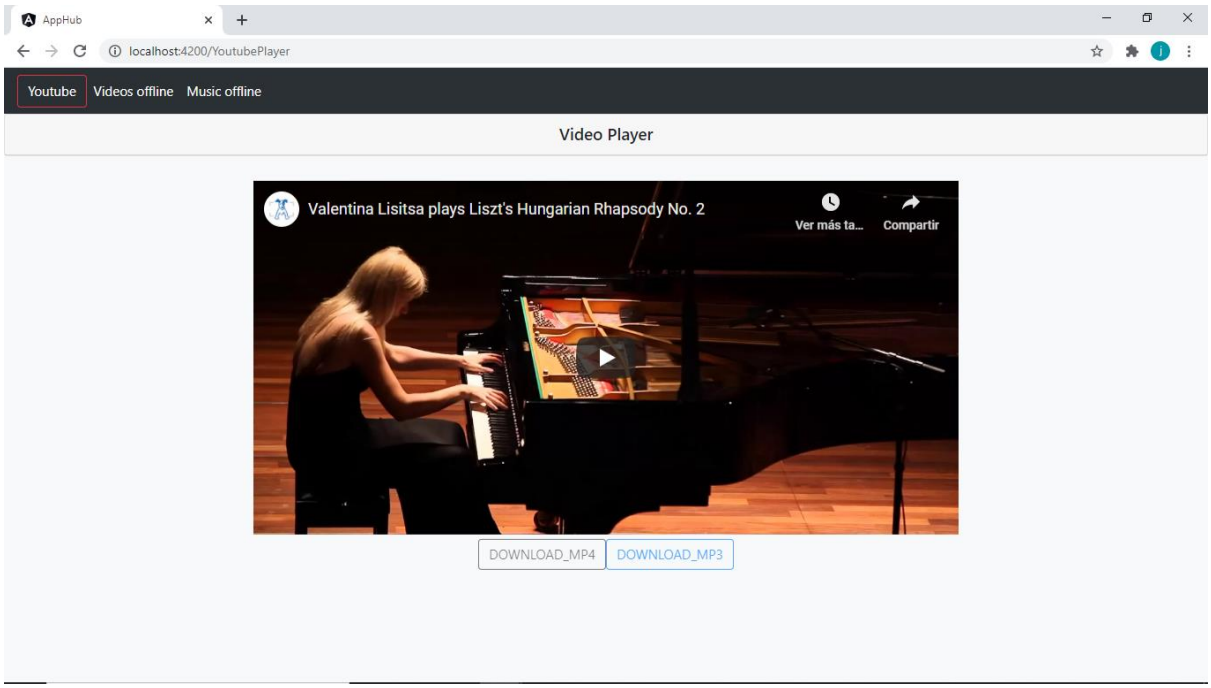
Visualización en pantalla completa



Descargar Archivo

Adicionalmente, en la misma pantalla de reproducción se contará con dos funcionalidades extra en caso de ser un usuario administrador.

Si no se es administrador, los dos botones no serán funcionales.

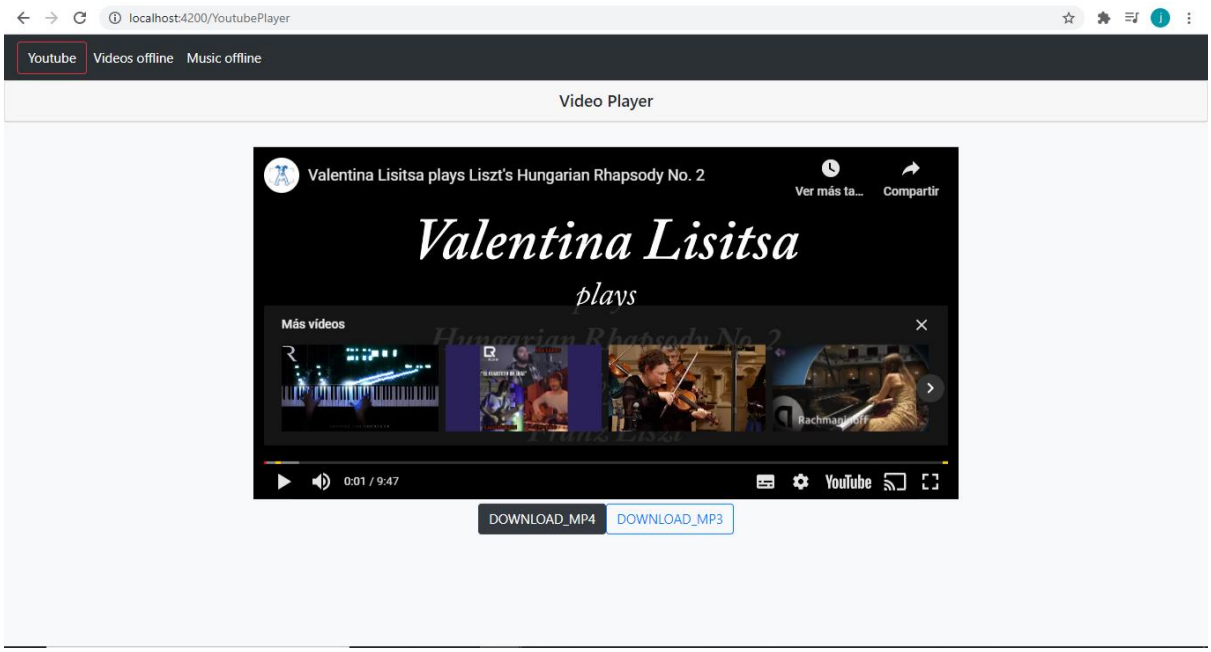


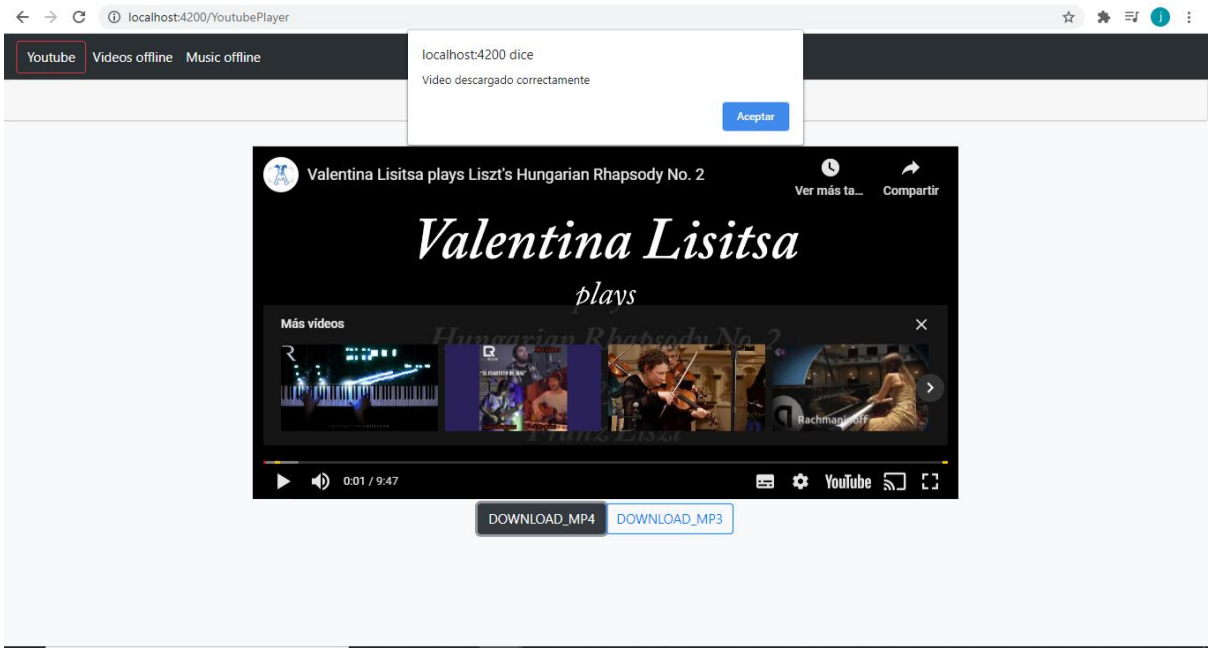
En caso de serlo, los dos botones aparecerán funcionales, por lo que se puede escoger el deseado:

- Descarga de video
- Descarga de audio

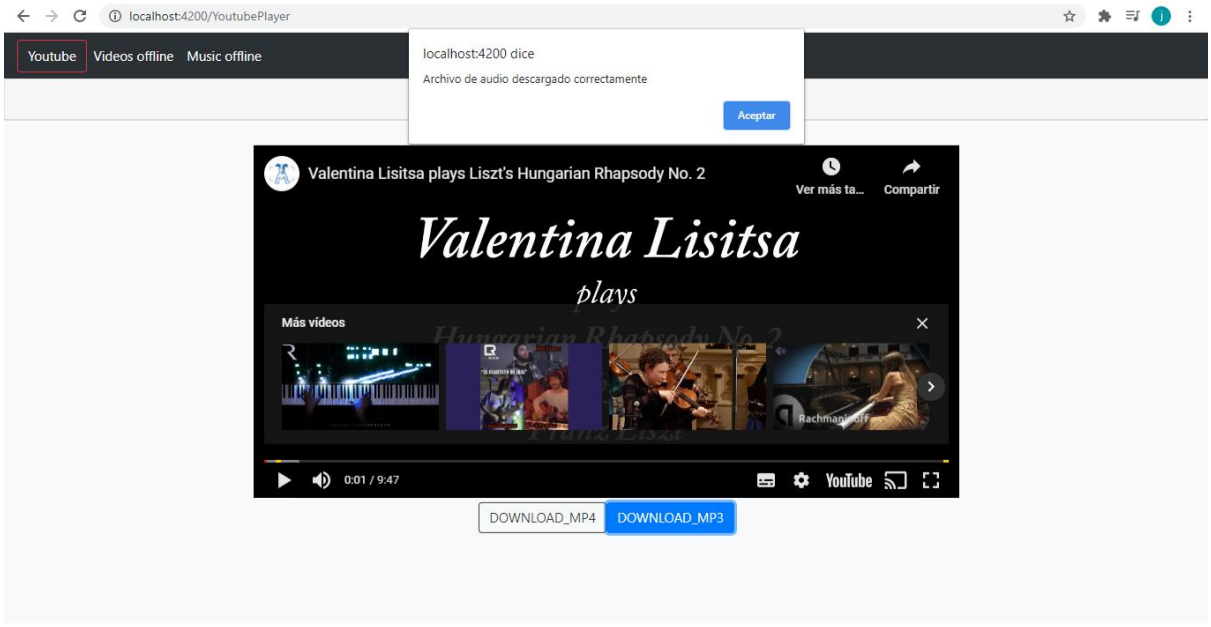
Dependiendo del botón escogido, se descargará el archivo .mp4 o el .mp3.

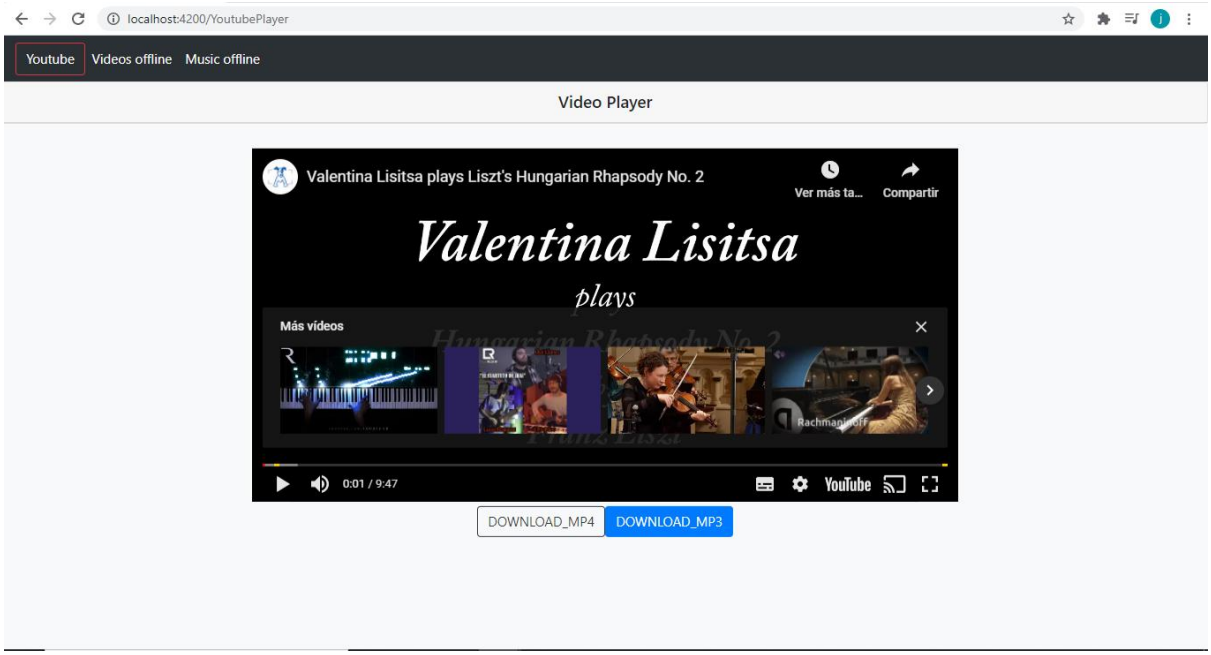
Descarga de Video correcta


localhost:4200/YoutubePlayer
Youtube
Videos offline
Music offline
Video Player
Valentina Lisitsa plays Liszt's Hungarian Rhapsody No. 2
Ver más ta...
Compartir
Valentina Lisitsa
plays
Más vídeos
0:01 / 9:47
YouTube
DOWNLOAD_MP4
DOWNLOAD_MP3

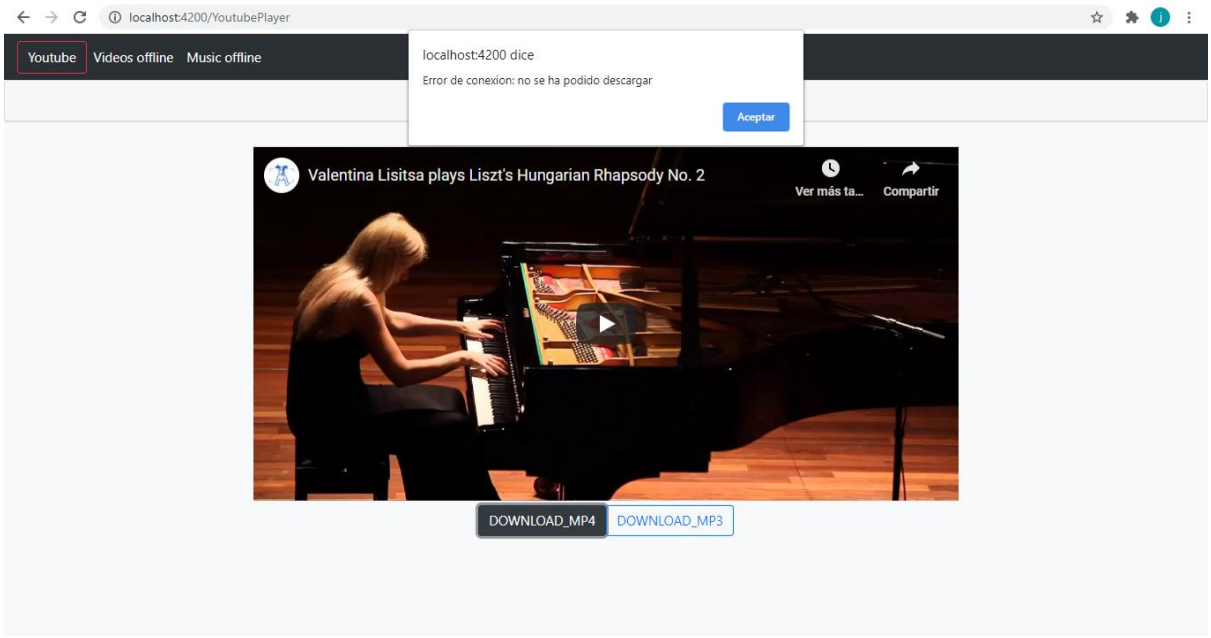

localhost:4200/YoutubePlayer
Youtube
Videos offline
Music offline
localhost:4200 dice
Vídeo descargado correctamente
Aceptar
Valentina Lisitsa plays Liszt's Hungarian Rhapsody No. 2
Ver más ta...
Compartir
Valentina Lisitsa
plays
Más vídeos
0:01 / 9:47
YouTube
DOWNLOAD_MP4
DOWNLOAD_MP3

Descarga de Música correcta

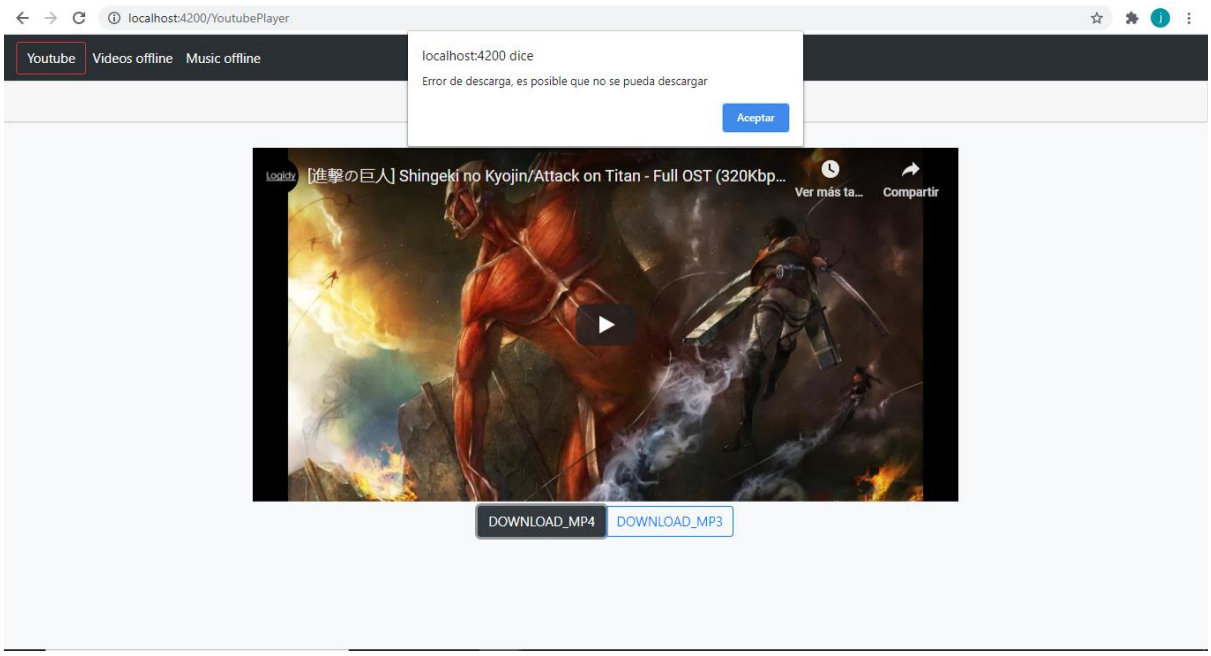




En caso de haber un error de conexión se mostrará el siguiente mensaje



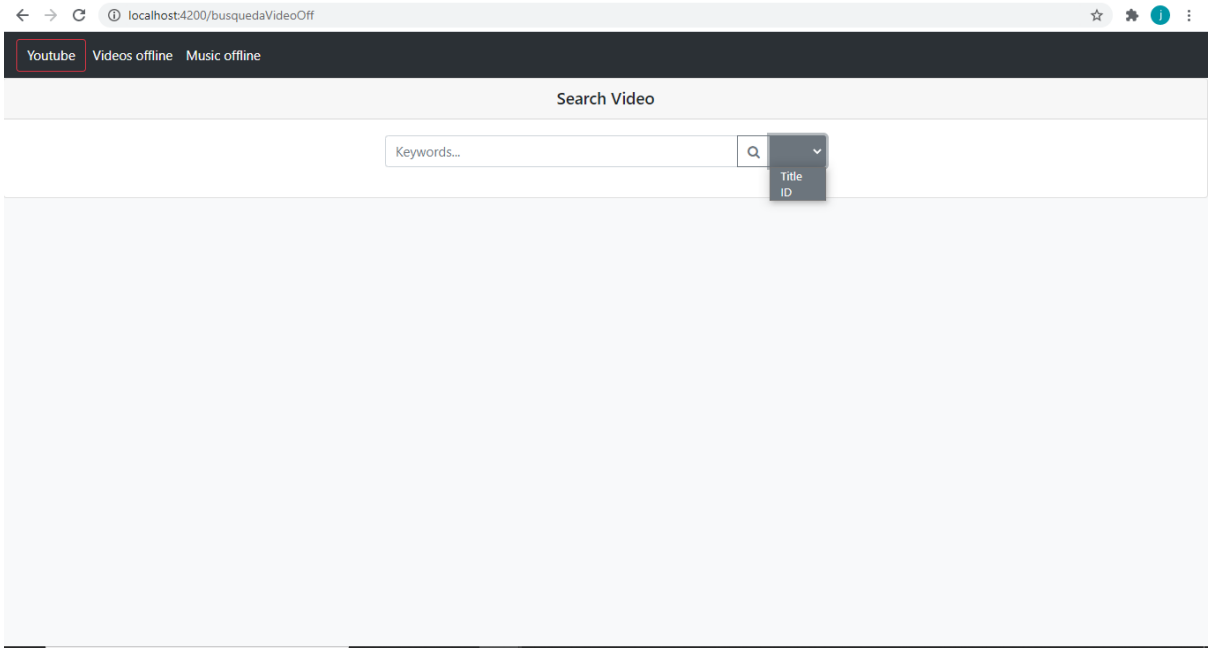
Debido a las políticas de Youtube antes mencionadas, hay urls que no son descargables, por lo que se mostrará el siguiente mensaje en caso de darse la situación.



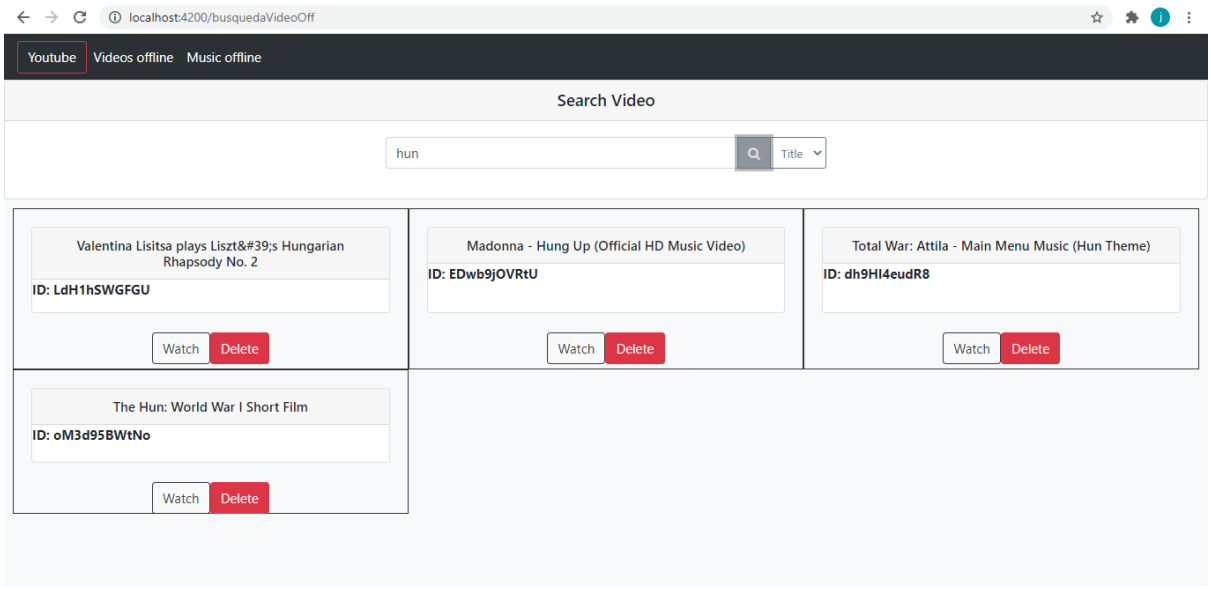
Búsqueda de Videos offline

Una vez que se quieran visualizar los vídeos ya descargados, y no necesitar la conexión a internet, se podrá utilizar la barra de navegación para llegar hasta aquí, pulsando “Vídeos offline”.

Como se puede ver, es un buscador similar al buscador de youtube, con la diferencia de que, en este caso, la elección no es de ordenación, si no de búsqueda, pudiendo buscar vídeos por su id o por su título.



Aparecerán aquellos cuyas características tengan un grado de similitud con lo introducido

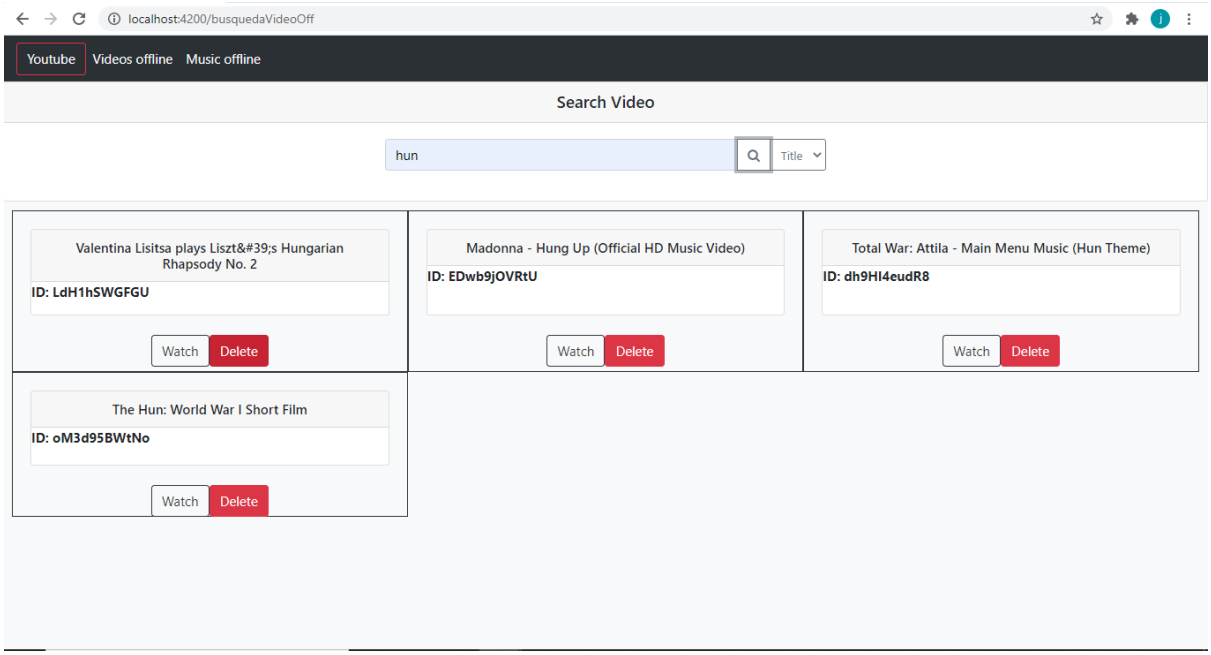


En este punto, se pueden diferenciar dos opciones:

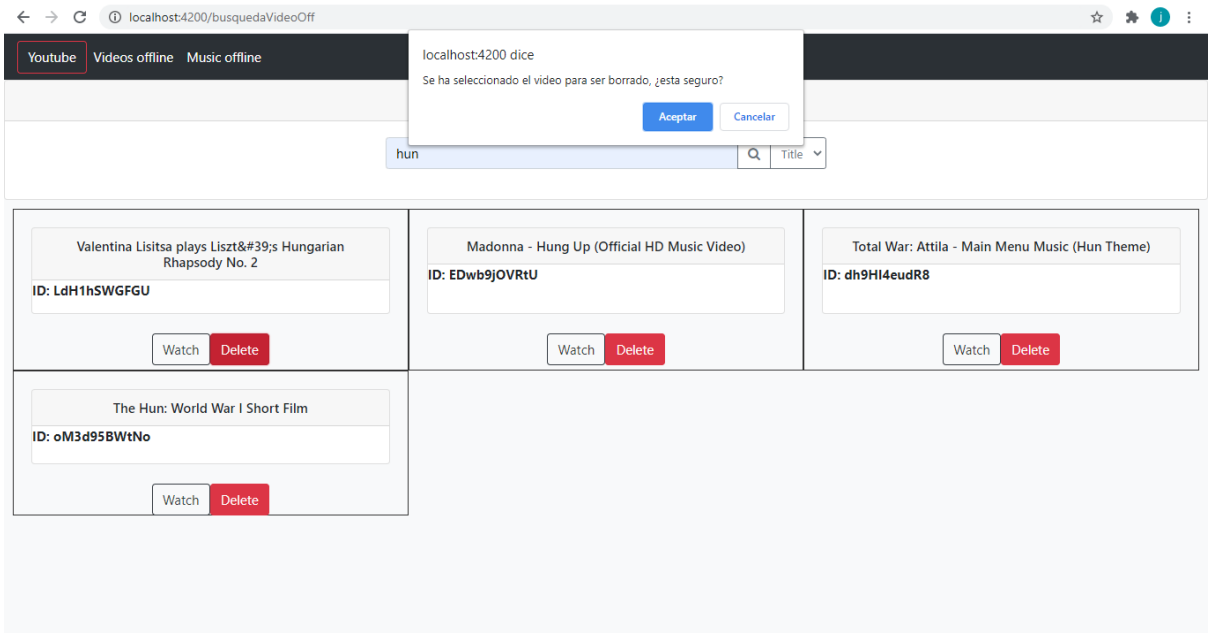
- Borrado de video
- Reproducción de video

Borrado de Videos

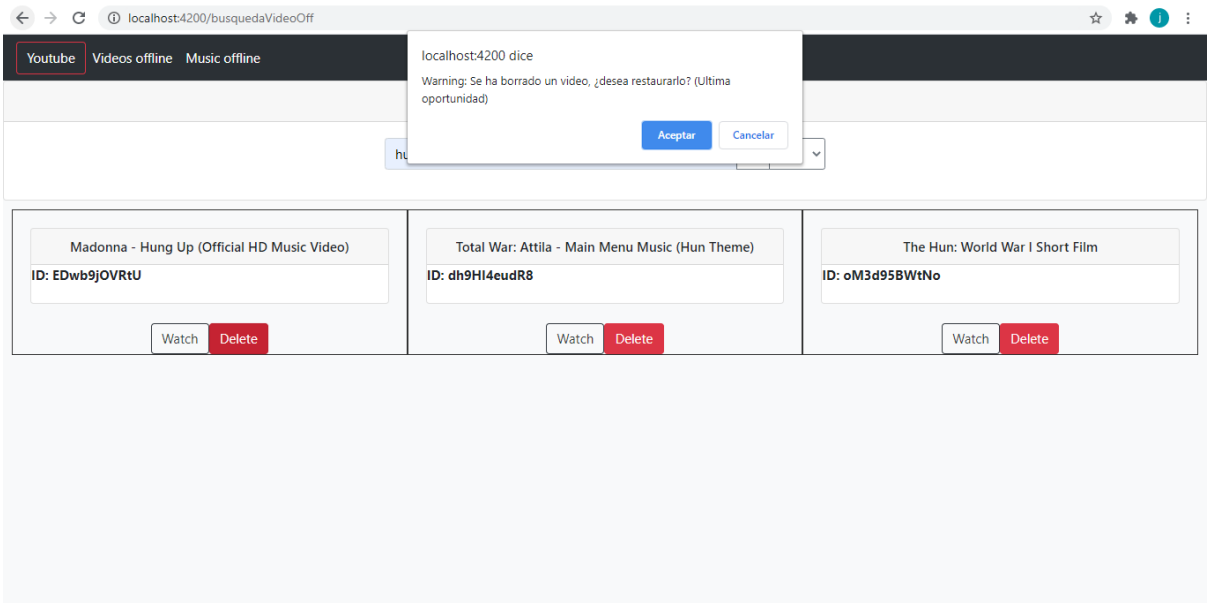
En caso de querer borrar un video de la lista, se podrá seleccionar el botón delete del video que queramos.



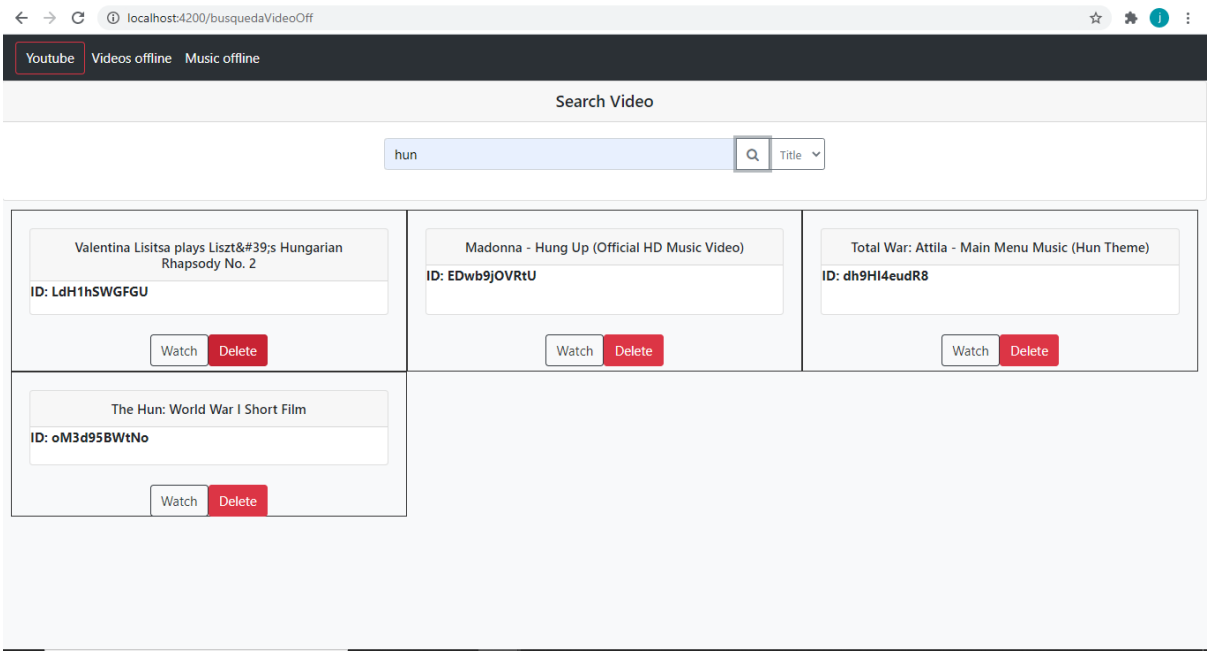
Esto nos mostrará un mensaje con un aviso, por el cual se confirme si el borrado del video es o no intencionado.



En caso de pulsar Aceptar, se mostrará un último mensaje de comprobación, para que, en caso de haberse borrado un video equivocado, este se pueda restaurar.



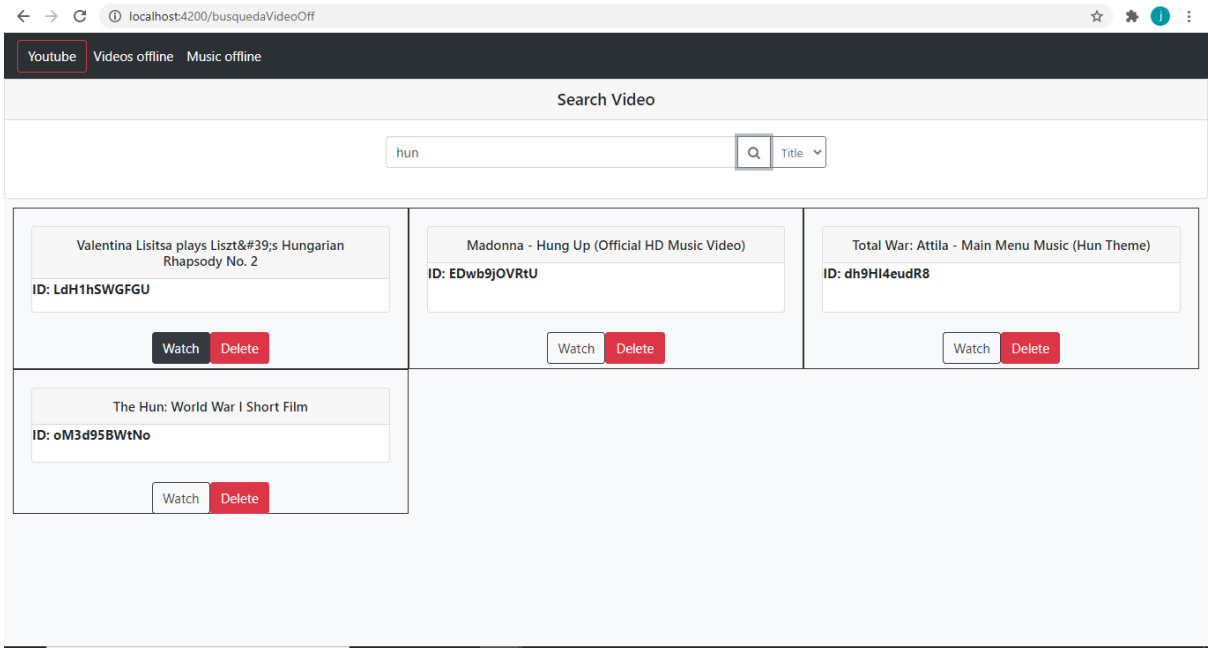
En caso de restaurar el video, este reaparecerá dentro de la lista de videos visibles, siendo por lo tanto, restaurado el archivo y enlace para su visualización



Finalmente, en caso de que se quiera eliminar el video de forma definitiva, una vez pulsado el botón cancelar, se borrará el video.

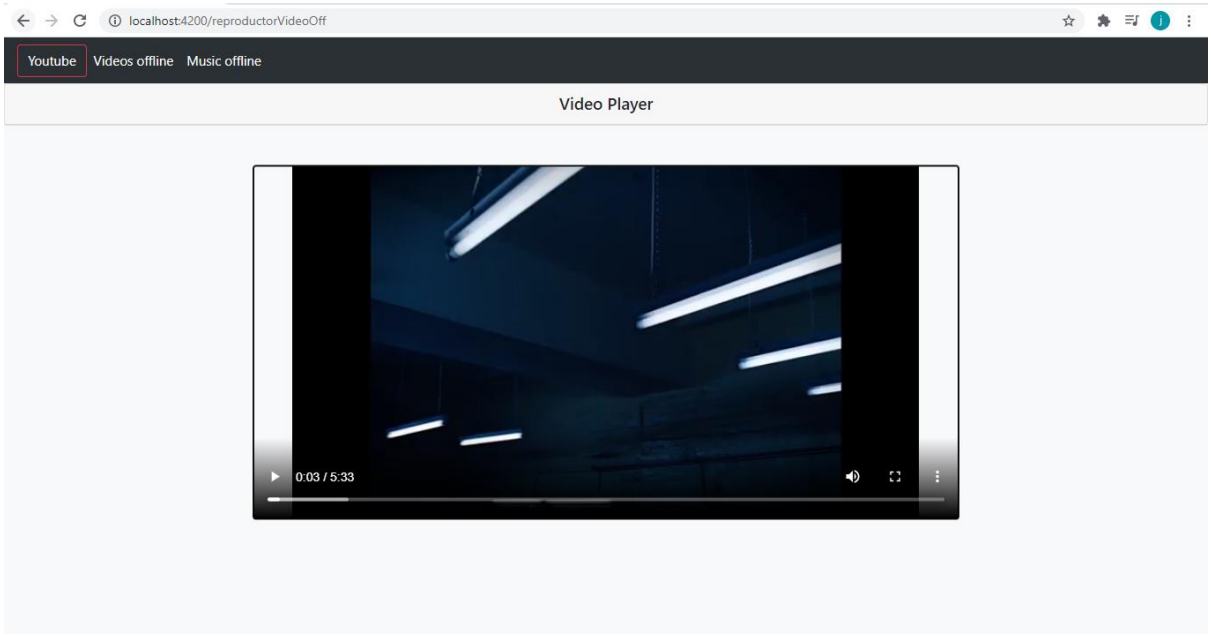
Reproducción de Vídeos offline

En caso de querer reproducir un video, se pulsará el botón de Watch el video que queramos visualizar.

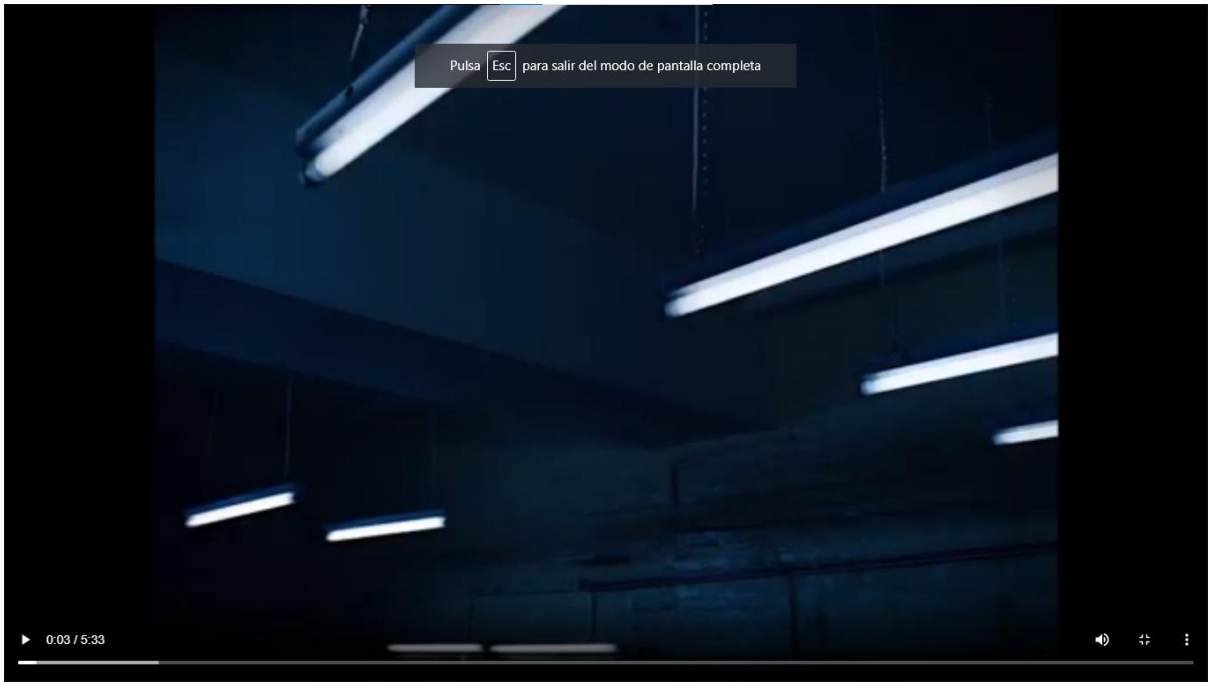


Esto nos llevará al reproductor de videos offline, que tendrá, similar al reproductor de youtube, las siguientes funciones de control:

- Play/Pause
- Bajar/Subir volumen
- Cambiar Configuración de reproducción
- Pantalla Completa
- Avanzar en el tiempo
- ...



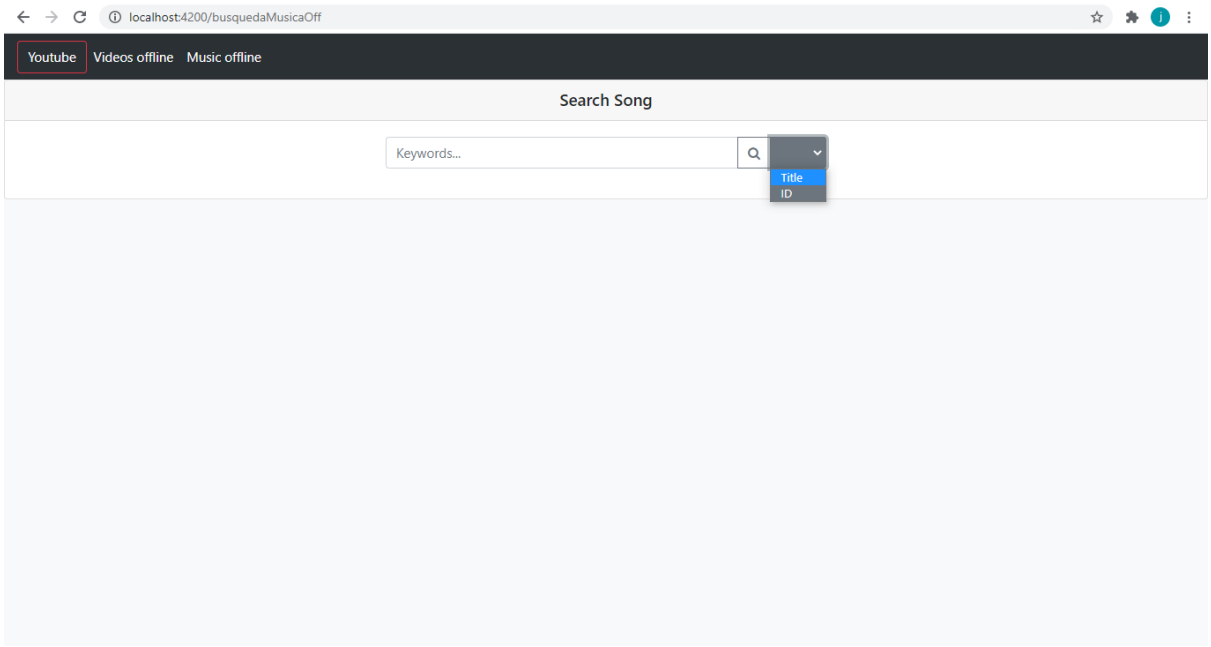
Visualización en pantalla completa (en caso de disminuir, pulsar el botón ESC)



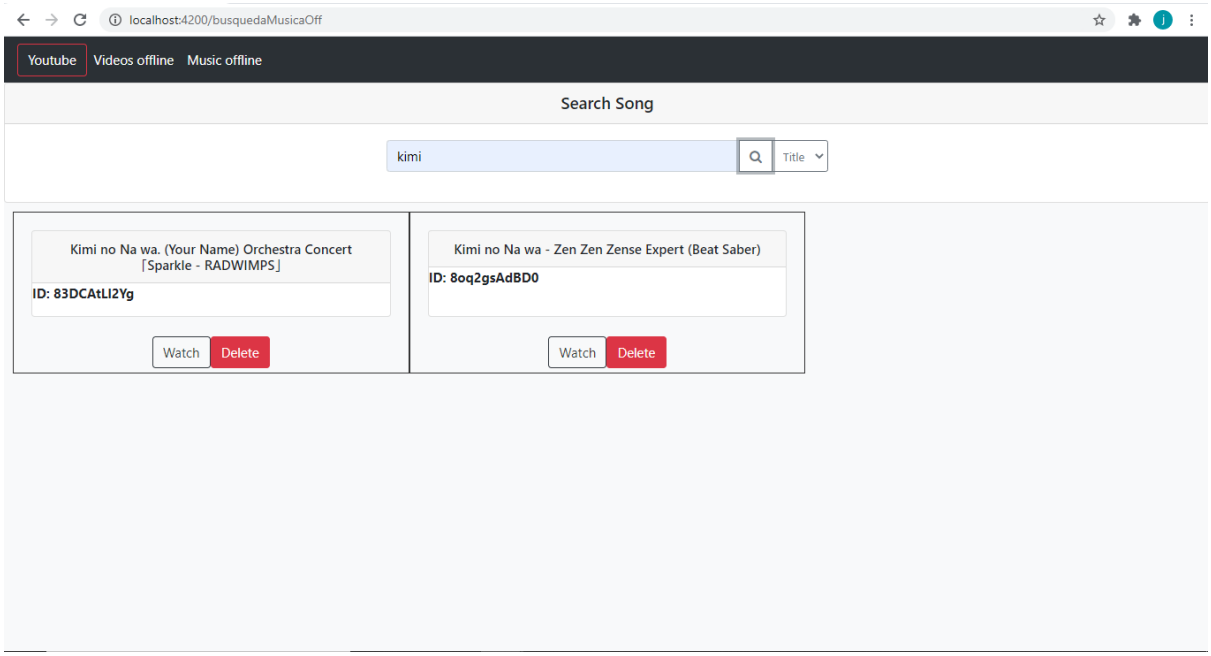
Búsqueda de Música offline

Una vez que se quieran visualizar los vídeos ya descargados, y no necesitar la conexión a internet, se podrá utilizar la barra de navegación para llegar hasta aquí, pulsando “Música offline”.

Como se puede ver, es un buscador similar al buscador de videos offline, contando con las mismas características de búsqueda (por id o por título)



Aparecerán aquellas canciones/archivos de audio cuyas características tengan un grado de similitud con lo introducido

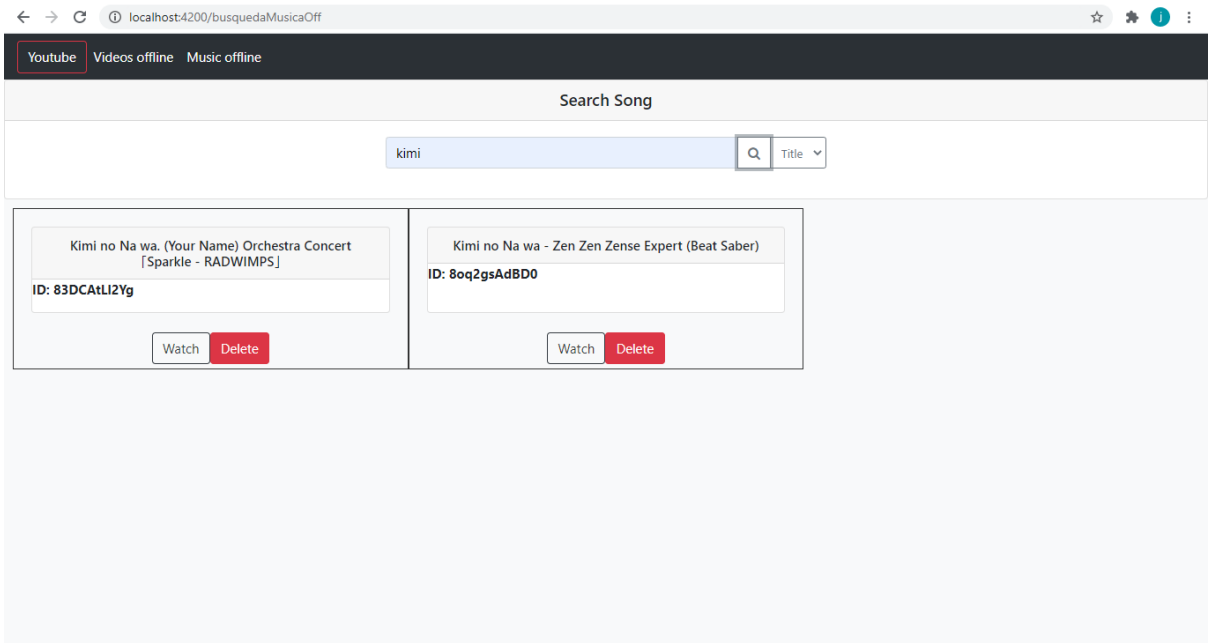


En este punto, se pueden diferenciar dos opciones:

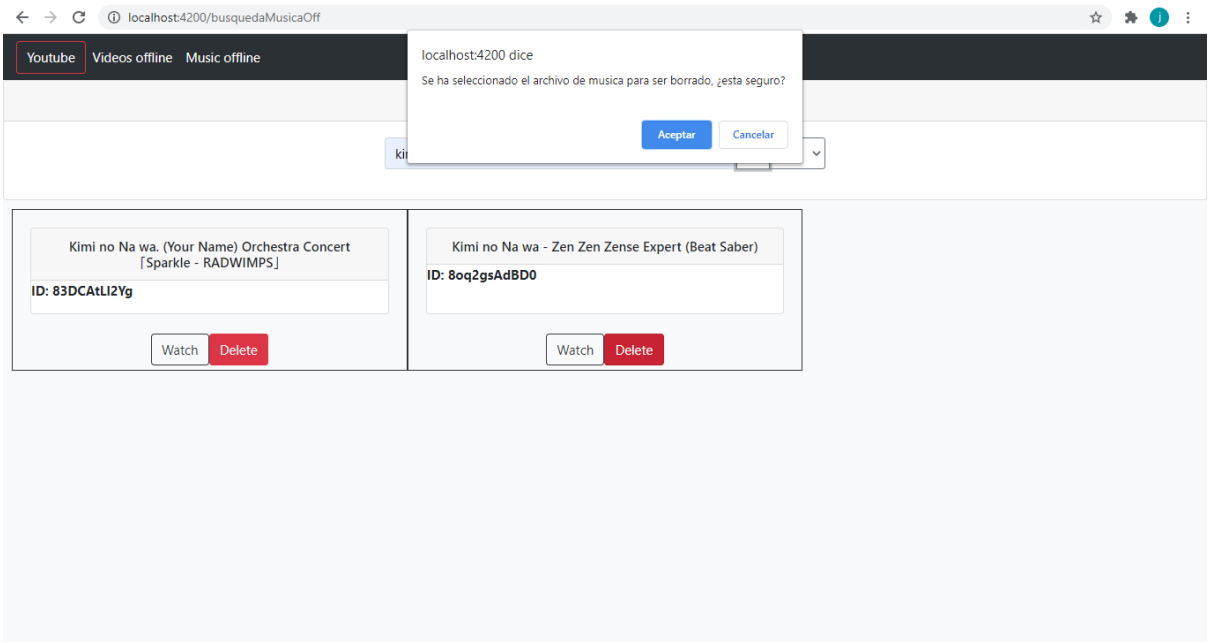
- Borrado de video
- Reproducción de video

Borrado de Música

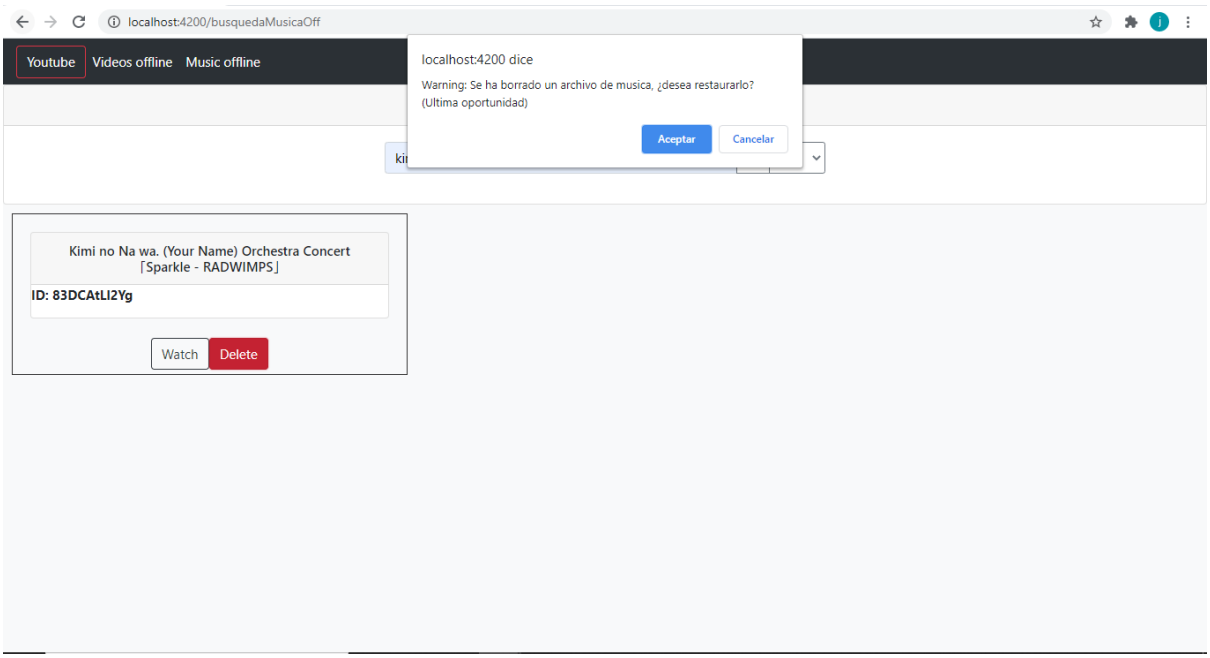
En caso de querer borrar un archivo de audio de la lista, se podrá seleccionar el botón delete del video que queramos.



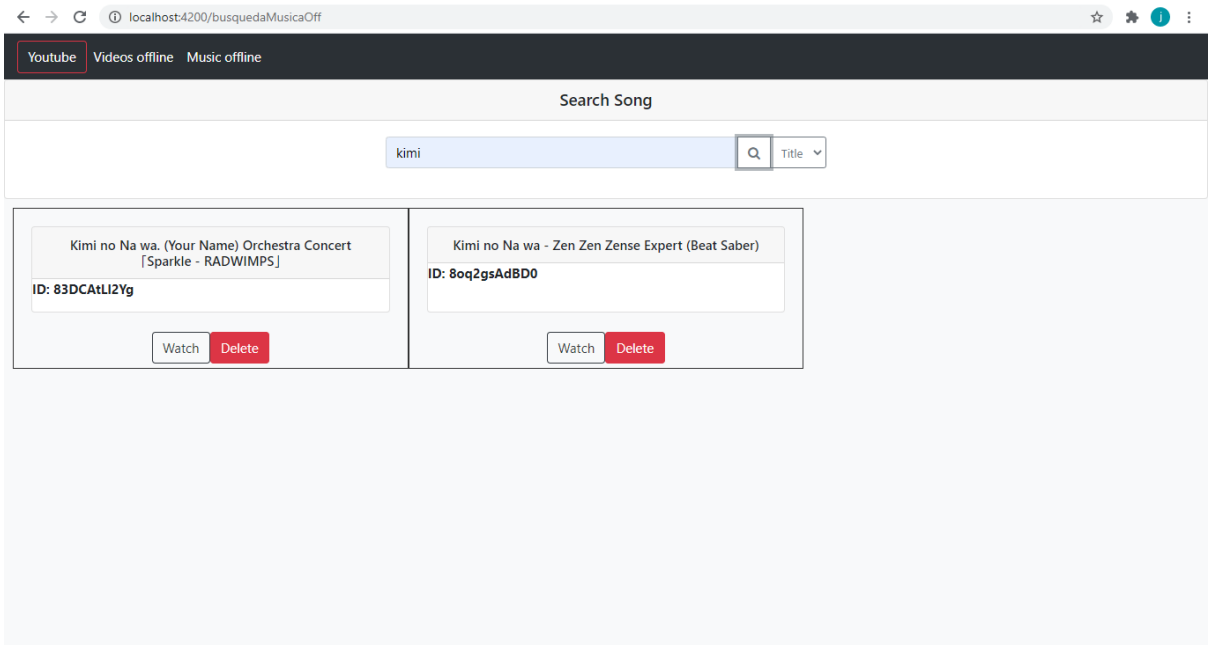
Esto nos mostrará un mensaje con un aviso, por el cual se confirme si el borrado del archivo de audio es o no intencionado.



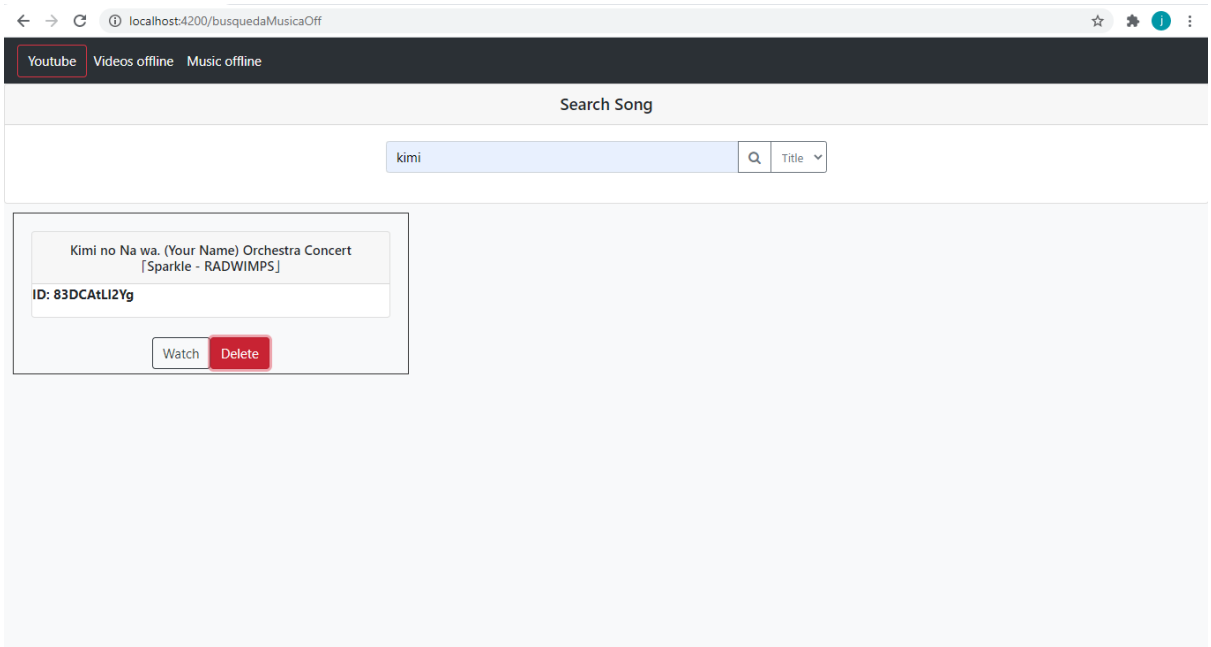
En caso de pulsar Aceptar, se mostrará un último mensaje de comprobación, para que, en caso de haberse borrado un archivo de audio equivocado, este se pueda restaurar.



En caso de restaurar el archivo, este reaparecerá dentro de la lista de archivos de audio visibles, siendo, por lo tanto, restaurado el archivo y enlace para su reproducción

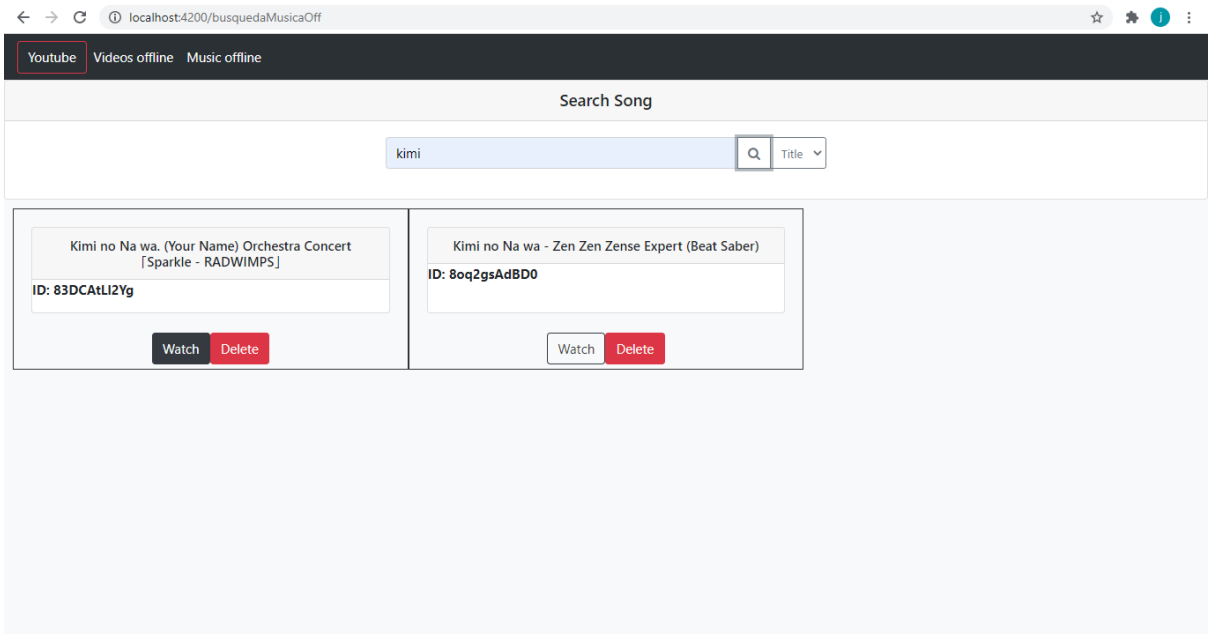


Finalmente, en caso de que se quiera eliminar el archivo de forma definitiva, una vez pulsado el botón cancelar, se borrará el archivo de audio.

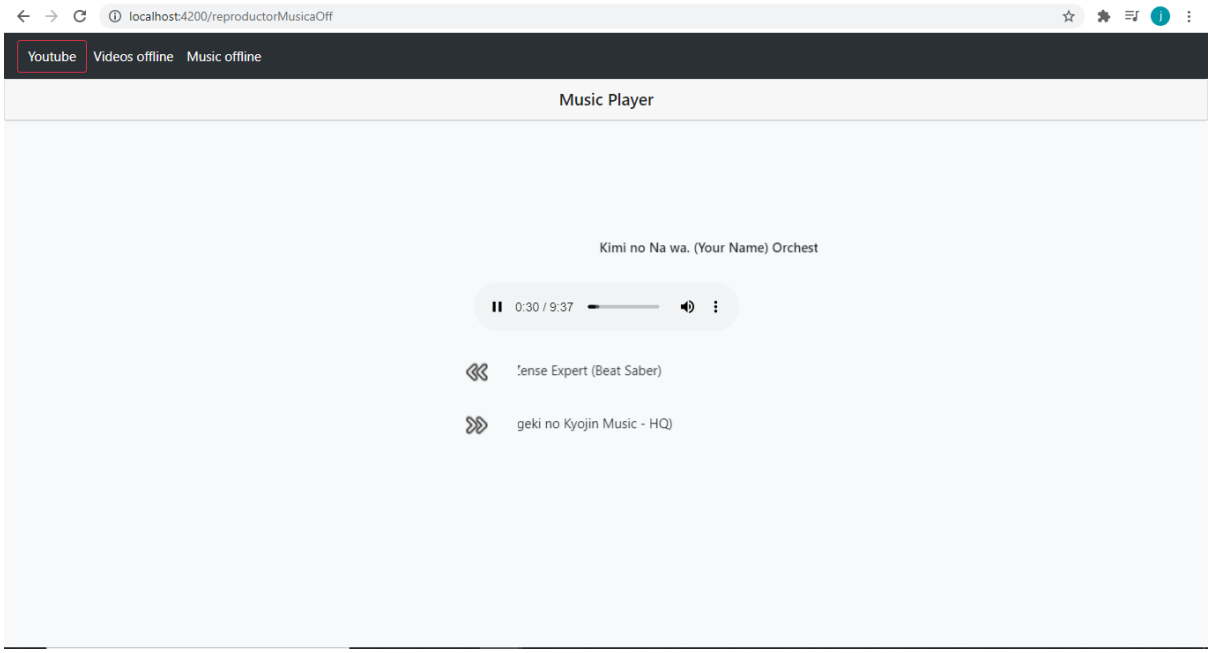


Reproducción de Música offline

En caso de querer reproducir un video, se pulsará el botón de Watch el video que queramos visualizar.



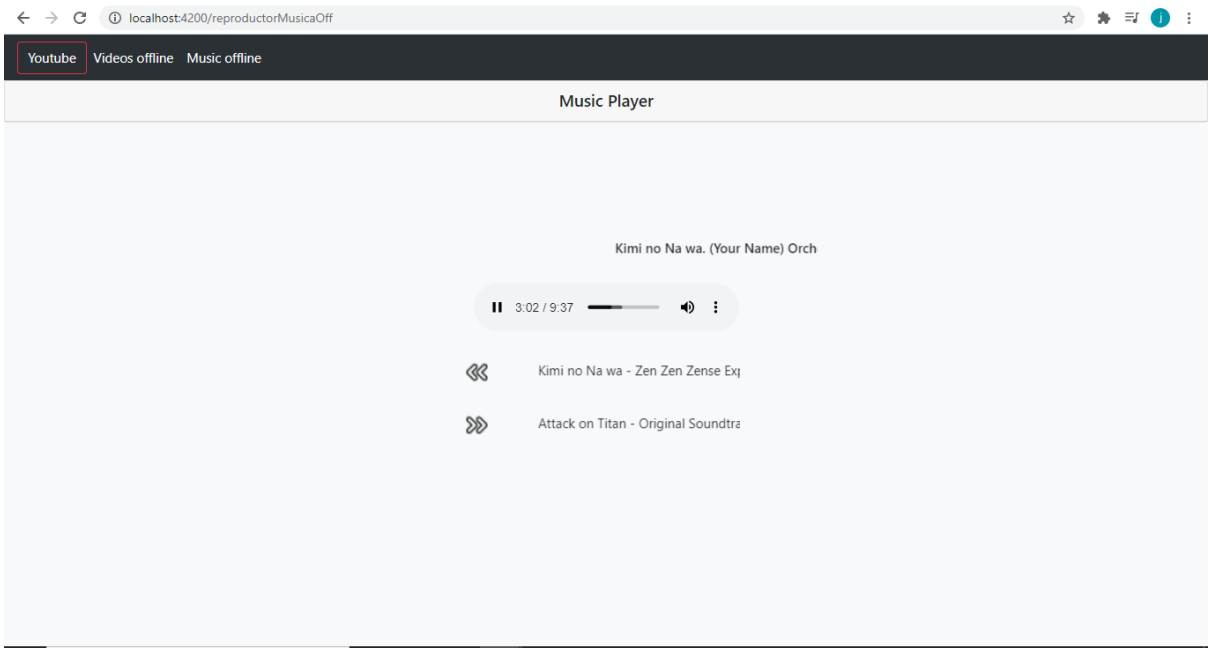
Esto nos llevará al reproductor de archivos de audio, donde podremos escuchar el audio seleccionado.



El audio se reproducirá de manera automática, y se contarán con los controles de:

- play/pause
- Avanzar en el tiempo
- Disminuir/Aumentar volumen

Adicionalmente, se podrá visualizar el título de la canción anterior y posterior a la canción que estamos escuchando, de forma que, se pueda cambiar a la canción posterior o anterior.



Al seleccionar cualquiera de las dos opciones, la canción se reproducirá automáticamente.