

PROYECTO FINAL DESARROLLO DE APPS



Universidad Veracruzana

Experiencia educativa: Desarrollo de aplicaciones

Docente: María Angélica Torres Huesca

Alumno: Aarón Hernández Lara

Introducción	3
Instrucciones de uso	3
Inicio de servidor	3
Cliente	4
Requerimientos	4
Diagrama de Contexto	4
Clases de usuario	5
Pototipos	6
Requisitos funcionales	8
Diseño	11
Diseño Arquitectónico	11
Vista de casos de uso	11
Vista lógica(Diagrama de clases)	12
Vista de despliegue	13
Modelo de datos	14
Modelo de datos BdUsuarios	15
Modelo de datos BdMusica y BDMusicaPrivada	16
Modelo de datos BdMultimedia	17
Construcción	25
Selección justificada de la tecnología	25
Aplicación de cliente rico	25
Servidor	25
Prácticas de construcción realizadas	26
Estrategia de despliegue	26
Conclusiones.	27

Introducción

En este documento se da una breve explicación sobre la funcionalidad del proyecto de LiberMusicMobile y un resumen general de las subsiguientes secciones del documento.

LiberMusicMobile es una aplicación móvil de reproducción de música en streaming basado en el estándar DASH, sin anuncios ni restricciones de saltos de canciones, y totalmente gratuita. Esta aplicación también permite bibliotecas de contenido privado para quienes solamente deseen almacenar su contenido pero no hacerlo público. Con esta aplicación puedes crear listas de reproducción con la música que mas te guste para encontrarla disfrutarla desde un solo sitio

Antes de explicar las secciones subsiguientes de este documento es necesario aclarar que proyecto se basa en la idea del servicio de streaming conocido como Spotify, sin embargo el equipo de desarrollo se deslinda de cualquier tipo de plagio y /o copia mal intencionada, ya que solo tomamos la inspiración para utilizar esto en el ámbito escolar, para esto utilizamos microservicios y un api, la cual traduce las llamas del cliente y las contesta utilizando los micro servicio de los que tiene conocimiento.

En las siguientes secciones se brindan las instrucciones de uso del sistema, para el cliente y para los microservicios. Posteriormente se presentan los requisitos y diagramas relacionados a estos. Después se muestra la sección de diseño con sus correspondientes diagramas. Después en la sección construcción se explican las razones de la selección de la tecnología utilizada, así como las practicas utilizadas. Subsecuentemente la sección estrategia de despliegue explica la forma y herramientas utilizadas para desplegar la aplicación

Instrucciones de uso

Inicio de servidor

1. clonar o copiar el contenido de la rama main del proyecto de github
2. comentar los bloques de los servicios que no se quieran ejecutar y descomentar aquello que se desean ejecutar, cada servicio se compone de un contenedor de BD y un contenedor de aplicación, los nombres de los contenedores son intuitivos, los que empiezan con la palabra "mysql" son de Base de datos, y los que empiezan con "ms" son los contenedores de las aplicaciones de servicio, deberas descomentar el correspondiente contenedor de BD y el de su aplicación relacionada, por ejemplo "mysql_musica" corresponde al servicio "ms_musica".
3. Ejecutar el archivo docker-compose.yml con docker compose.
4. Esperar a que se construyan los contenedores de los servicios.

5. Repetir los pasos anteriores para cada uno de los servicios que se deseen ejecutar, si es que estos se van a ejecutar en una computadora diferente

Cliente

El cliente solo funcionara si se tienen corriendo el servidor con todos sus servicios correspondientes, si no se corrieran todos, algunas funciones darán resultados parciales o no funcionarán en absoluto

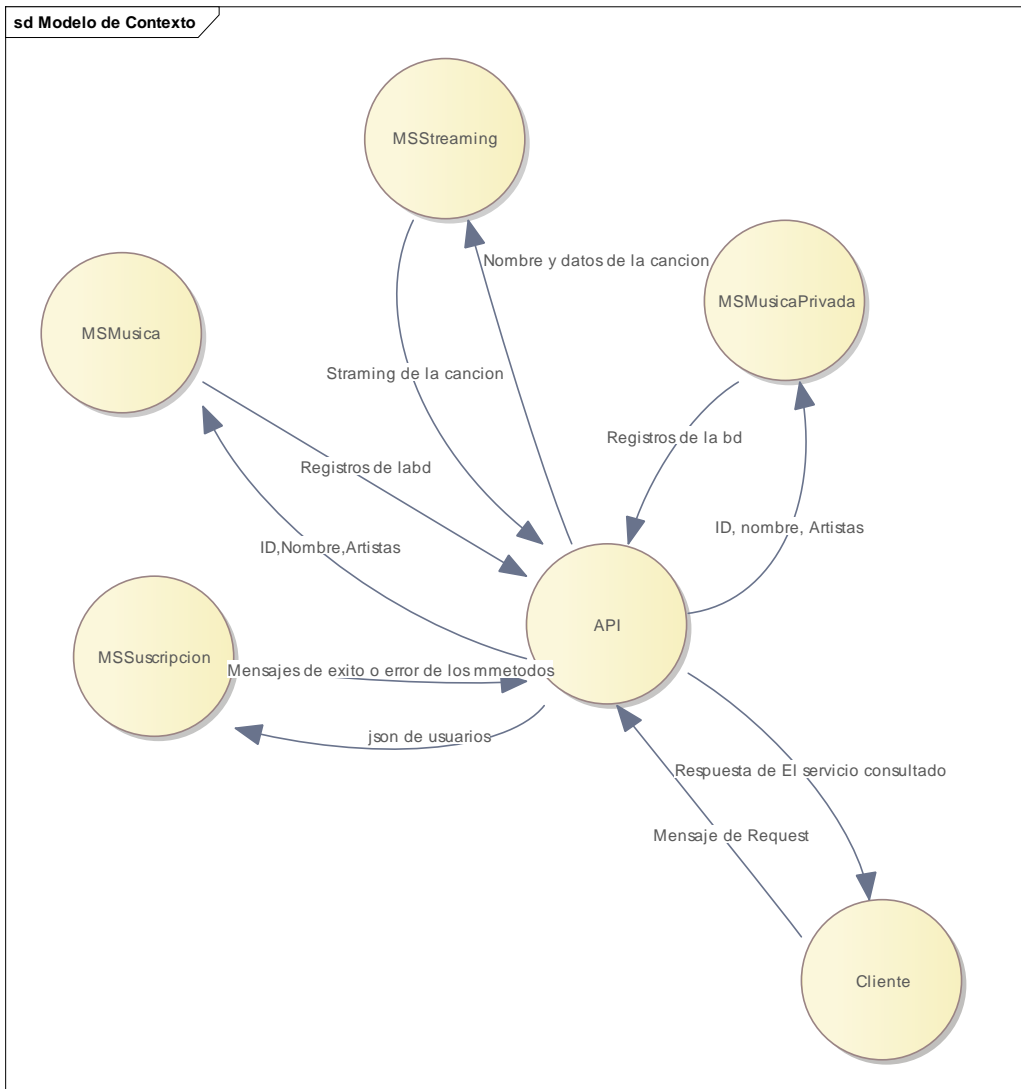
1. Copiar o clonar el contenido del repositorio <https://github.com/aaron-77/LiberMusicMobile> de la rama main.
2. Abrir el proyecto con android estudio o cualquier IDE para desarrollo en android y ejecutar la aplicación.
3. En pantalla de login seleccionar la opción registrar usuario.
4. Llenar los datos del formulario y registrar el usuario, apuntar los datos registrados para poder iniciar sesión.
5. Con los datos registrados anteriormente iniciar sesión en la pantalla de login.
6. Una vez iniciada sesión dirigirse a las funciones de interés ubicadas en la barra de menú inferior. Seleccione la opción home para ver todas las canciones disponibles. Seleccione la opción reproductor para ir al reproductor de música. Seleccione la opción listas de reproducción para ver sus listas de reproducción registradas. Seleccione la opción cuenta para cerrar sesión. Dentro de esta, seleccione la opción administrar contenido (en caso de estar registrado como artista) para registrar álbumes y subir canciones.
7. Al terminar puede cerrar su sesión y cerrar la aplicación.

5.- si se desean probar los servicios por medio de un cliente distinto como postman, hacer las peticiones al apigateway en una dirección y puerto local, a excepción de los servicios que requieran subir archivos, estos deben realizarse directamente a esos servicios.

Requerimientos

Diagrama de Contexto

El siguiente diagrama muestra el flujo de comunicación de Libermusic.



Como se puede observar en la imagen de arriba la aplicación de libermusicMobile(cliente)se comunica con el Api del backend de la aplicación, esta se encargara de comunicarse con los servicios necesarios brindar los datos requeridos en la aplicación cliente .

Clases de usuario

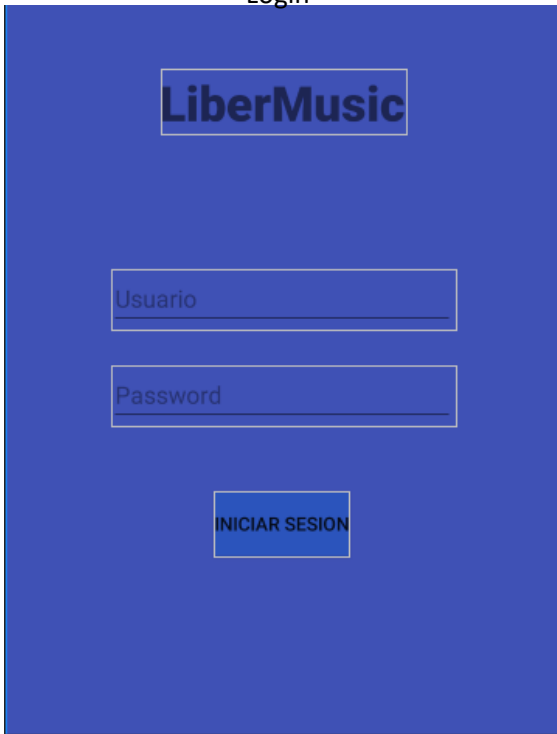
Las clases de usuario en la aplicación son 2:

Usuario común: Usuario registrado en la aplicación LiberMusicMobile que puede reproducir canciones, crear listas de reproducción y realizar búsquedas de canciones.

Artista: usuario de LiberMusic que tiene la capacidad de subir canciones a esta aplicación y administrar las mismas, así como también puede realizar las funciones del **usuario común**

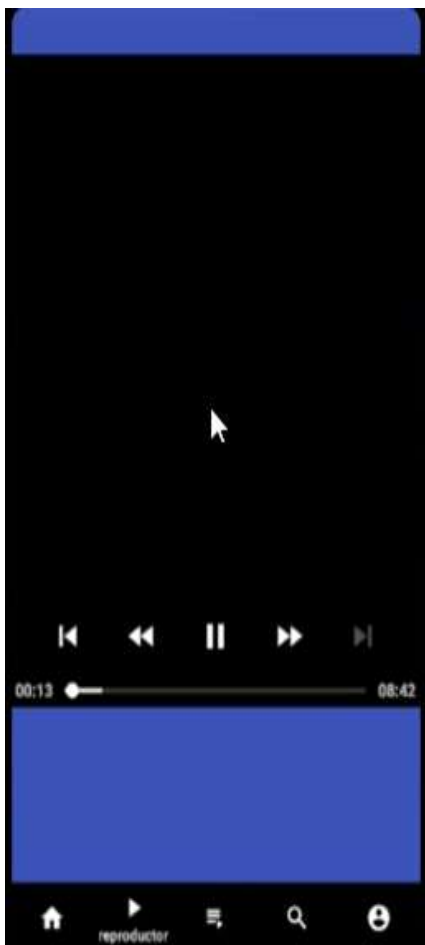
Pototipos

Login



The login screen has a solid blue background. At the top center is the 'LiberMusic' logo in white. Below it are two white rectangular input fields, the first labeled 'Usuario' and the second 'Password'. At the bottom center is a white rectangular button with the text 'INICIAR SESION' in blue.

Reproductor



Listados de



Registro de artista



The artist registration form has a light gray background. It starts with a 'REGRESAR' button at the top left. Below it is a red square placeholder for a profile picture. Then is a button labeled 'ELEGIR DEL DISPOSITIVO'. This is followed by five text input fields: 'tu nombre', 'nombre artistico', 'año de nacimiento', 'pagina web', and 'nacionalidad'. At the bottom right is a button labeled 'REGISTRAR'.

Registro de usuario



REGRESAR

eliminar album

ELEGIR IMAGEN DEL DISPOSITIVO

nombre album

fecha de lanzamiento

duracion

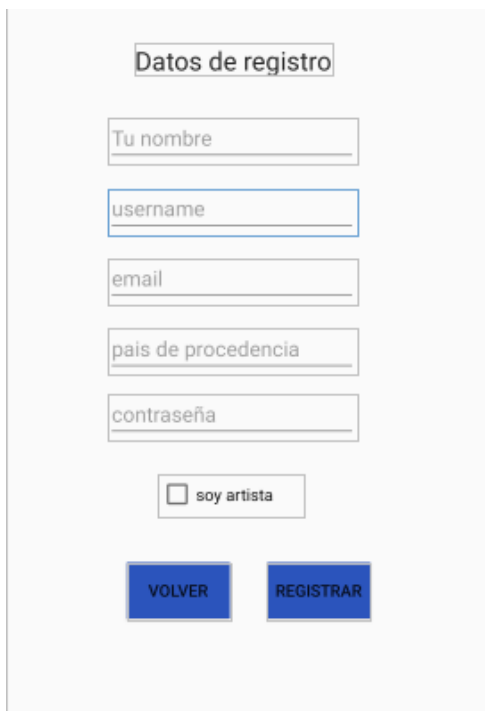
numero de tracks

disquera

tipo album

canciones

Registro de usuario



Datos de registro

Tu nombre

username

email

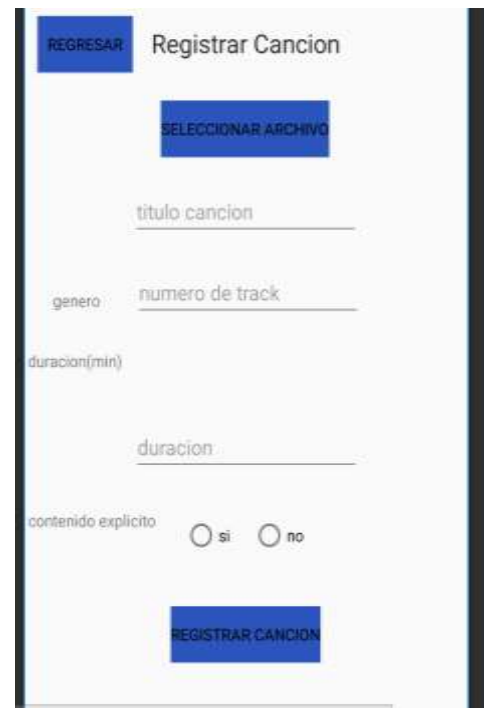
pais de procedencia

contraseña

☐ soy artista

VOLVER REGISTRAR

Registro de cancion



REGRESAR Registrar Cancion

SELECCIONAR ARCHIVO

titulo cancion

genero numero de track

duracion(min)

duracion

contenido explicito ☐ si ☐ no

REGISTRAR CANCION

Requisitos funcionales

RF-01	<p>La interfaz gráfica de la ventana de detalle de un álbum debe contener un elemento gráfico que al darle click abra un formulario para registrar una canción con los siguientes datos y elementos gráficos:</p> <p>datos:</p> <ul style="list-style-type: none"> album relacionado titulo de la canción numero de track género código Isrc duración contenido explicito archivo de la cancion en mp3 <p>elementos gráficos:</p> <ul style="list-style-type: none"> elemento para registrar la canción <p>cuando el artista de click en el elemento para registrar la canción tiene que iniciarse el proceso de registro de la canción.</p>
RF-02	<p>El reproductor de música debe tener las siguientes funciones</p> <ul style="list-style-type: none"> Iniciar/pausar canción Adelantar canción Retroceder canción
RF-03	La interfaz de usuario de LiberMusic debe tener una función que permita eliminar las listas reproducción de un usuario de la base de datos de liberMusic
RF-04	La interfaz de usuario de LiberMusic debe tener una función que permita cambiar el nombre de una lista de reproducción.
RF-05	La interfaz gráfica de LiberMusic debe tener la función que permita agregar una canción a una lista de reproducción.
RF-06	La interfaz gráfica de LiberMusic debe tener la función que permita eliminar una canción de una lista de reproducción.
RF-07	<p>La interfaz de usuario debe contar con la función de registrar usuarios en la aplicación con el rol de usuario común y artista. Para ello el formulario debe contener los siguientes datos:</p> <ul style="list-style-type: none"> elemento gráfico para indicar si el usuario es artista nombre completo correo electrónico nombre de usuario contraseña
RF-08	LiberMusic debe tener en su interfaz grafica un elemento grafico que al dar click sobre él muestre una barra de búsqueda y permita reproducirlas al seleccionarl as o bien agregarlas a una lista de reproducción.
RF-09	La interfaz gráfica del reproductor debe realizar RF-02 de acuerdo con el funcionamiento típico de los reproductores de música, teniendo un icono representativo de cada función definida en RF-02 y que al dar click en el se realice su correspondiente función.
RF-	La interfaz gráfica de LiberMusic debe tener un icono o elemento grafico

10	<p>representativo que al dar click sobre él muestre un formulario para crear una lista de reproducción con los siguientes datos y elementos:</p> <p>nombre de la lista de reproducción elemento gráfico para confirmar la creación de la lista de reproducción. Cuando el usuario de click en el elemento gráfico para confirmar la creación de la lista de reproducción debe iniciarse el proceso de registro de la lista de reproducción.</p>
RF-11	<p>Cuando el mecanismo establecido el RNF-03 detecte que la entrada del usuario no es valida, LiberMusic debe indicar al usuario el error cometido sin brindar información que pueda servir a un atacante para inferir el funcionamiento del sistema.</p>
RF-12	<p>La interfaz gráfica en la ventana de listas de reproducción de LiberMusic debe tener un icono o elemento gráfico representativo que al dar click sobre él muestre el buscador de RF-10 y el usuario al seleccionar una canción mostrada por el buscador esta se agregue a una lista de reproducción.</p>
RF-13	<p>La interfaz gráfica en la ventana de una lista de reproducción específica de un usuario en LiberMusic debe tener un icono o elemento gráfico representativo junto a cada canción en el que al dar click se elimine la canción correspondiente de la lista de reproducción del usuario correspondiente</p>
RF-14	<p>La interfaz gráfica de liberMusic debe tener un elemento gráfico que permita al usuario artista registrar un álbum de música con los siguientes datos :</p> <p>artista propietario título del album; duración número de tracks(canciones) del álbum compañía productora tipo de álbum (sencillo,edicion especial) fecha de lanzamiento url de portada.</p> <p>cuando el usuario de click sobre el elemento grafico correspondiente liberMusic debe iniciar el proceso de registro de album del artista</p>
RF-15	<p>Liber music debe tener una función que permite al supervisor de contenido aprobar o rechazar la publicación de una canción en la plataforma según las pautas de LiberMusic sobre los derechos de autor, la cual muestra un listado de las canciones que los artistas han registrado y no han sido aprobadas y junto a cada una de ellas una elemento gráfico para aprobar la canción y otro para rechazarla, y que al dar click en alguno de ellos se realice su correspondiente función</p>
RF-16	<p>LiberMusic debe realizar la reproducción a partir del consumo del API del servidor</p>

Requisitos no funcionales

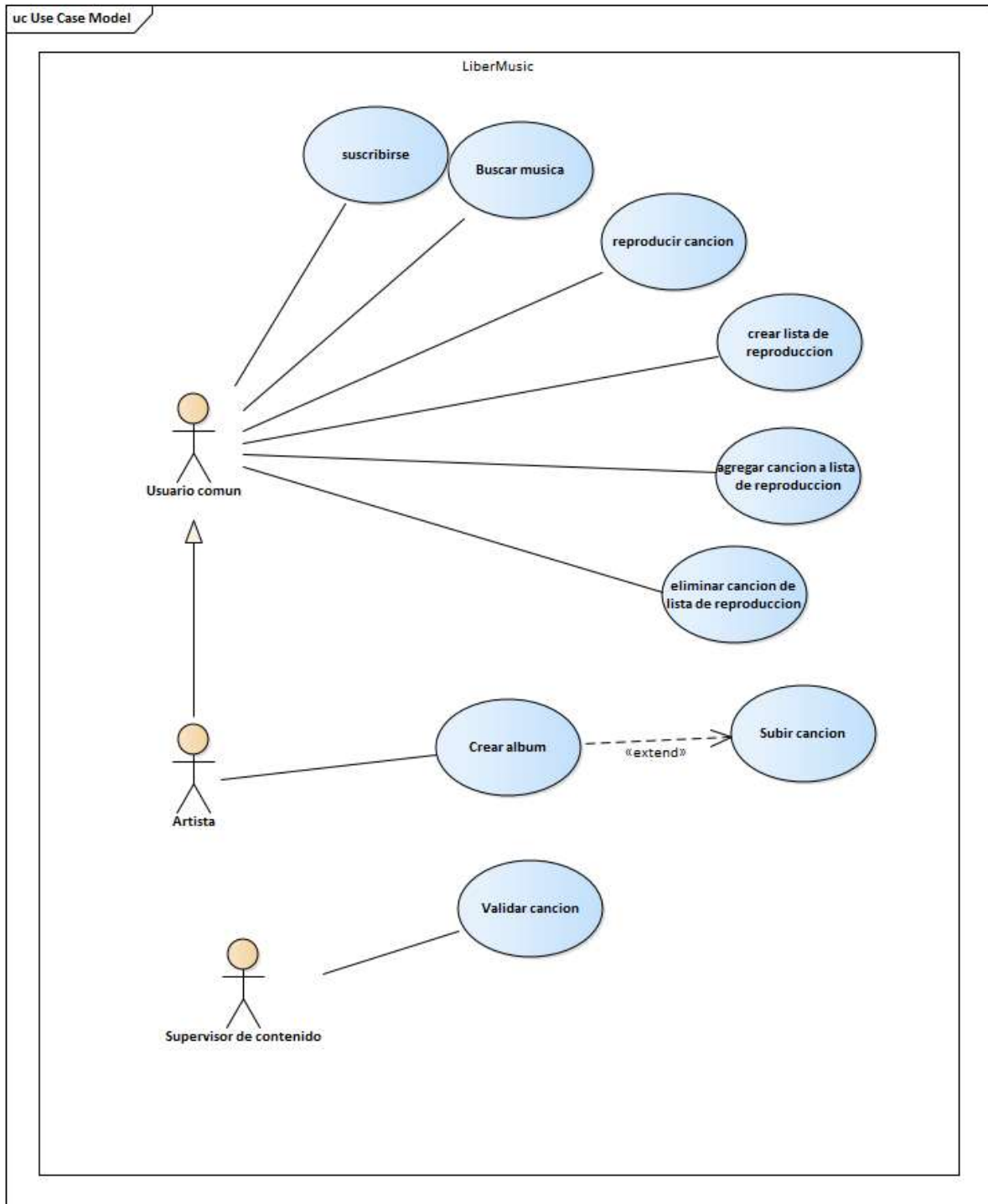
RNF-	LiberMusic debe apegarse a las pautas mas modernas de usabilidad para
-------------	--

01	aplicaciones móviles
RNF-02	LiberMusic debe tener en su interfaz gráfica iconos que representen las funciones ofrecidas por la aplicación, y al dar click sobre estos iconos LiberMusic debe realizar las acciones correspondientes al icono oprimido
RNF-03	Liber music debe establecer un mecanismo de validación de datos que contenga criterios claros que eviten que el usuario ingrese entradas malintencionadas o con caracteres no válidos de acuerdo al información que desea registrar y este se active cada que el usuario registre o edite información registrada en la base de datos
RNF-04	Libermusic debe implementar comunicaciones seguras como medida de seguridad para la comunicación en todas la peticiones de todos los roles de usuario y en todas las reproducciones de canciones
RNF-05	LiberMusic debe implementar un mecanismo de autenticación de usuarios para poder ingresar al sistema
RNF-06	Liber music debe implementar un mecanismo que asegure que se respeten los derechos de autor en la musica que se publica en la plataforma.

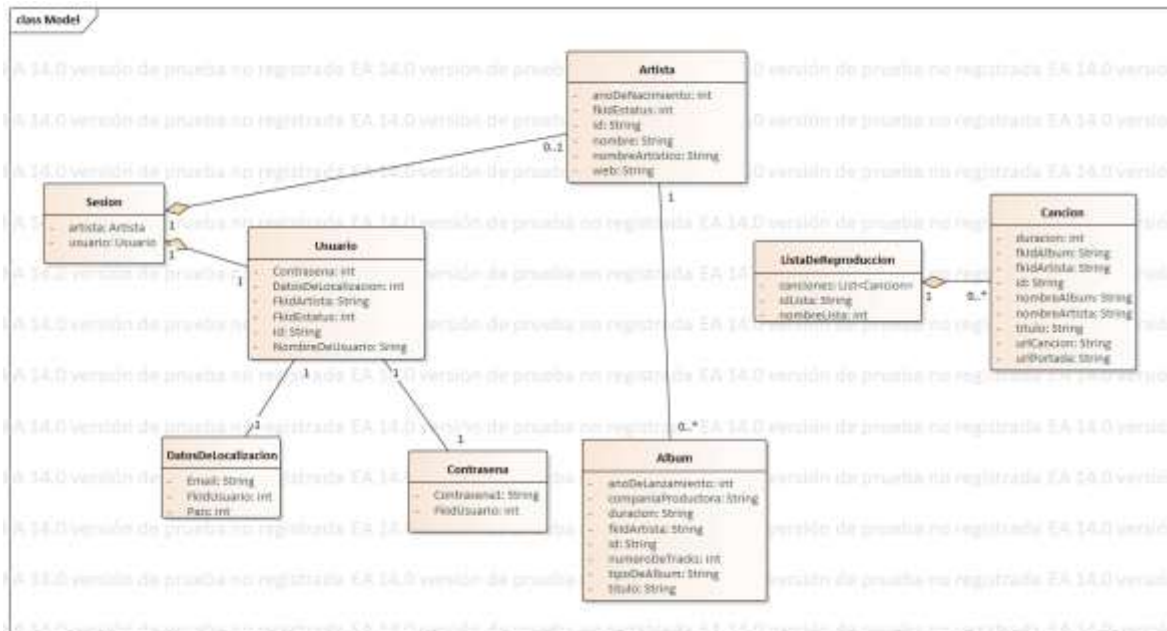
Diseño

Diseño Arquitectónico

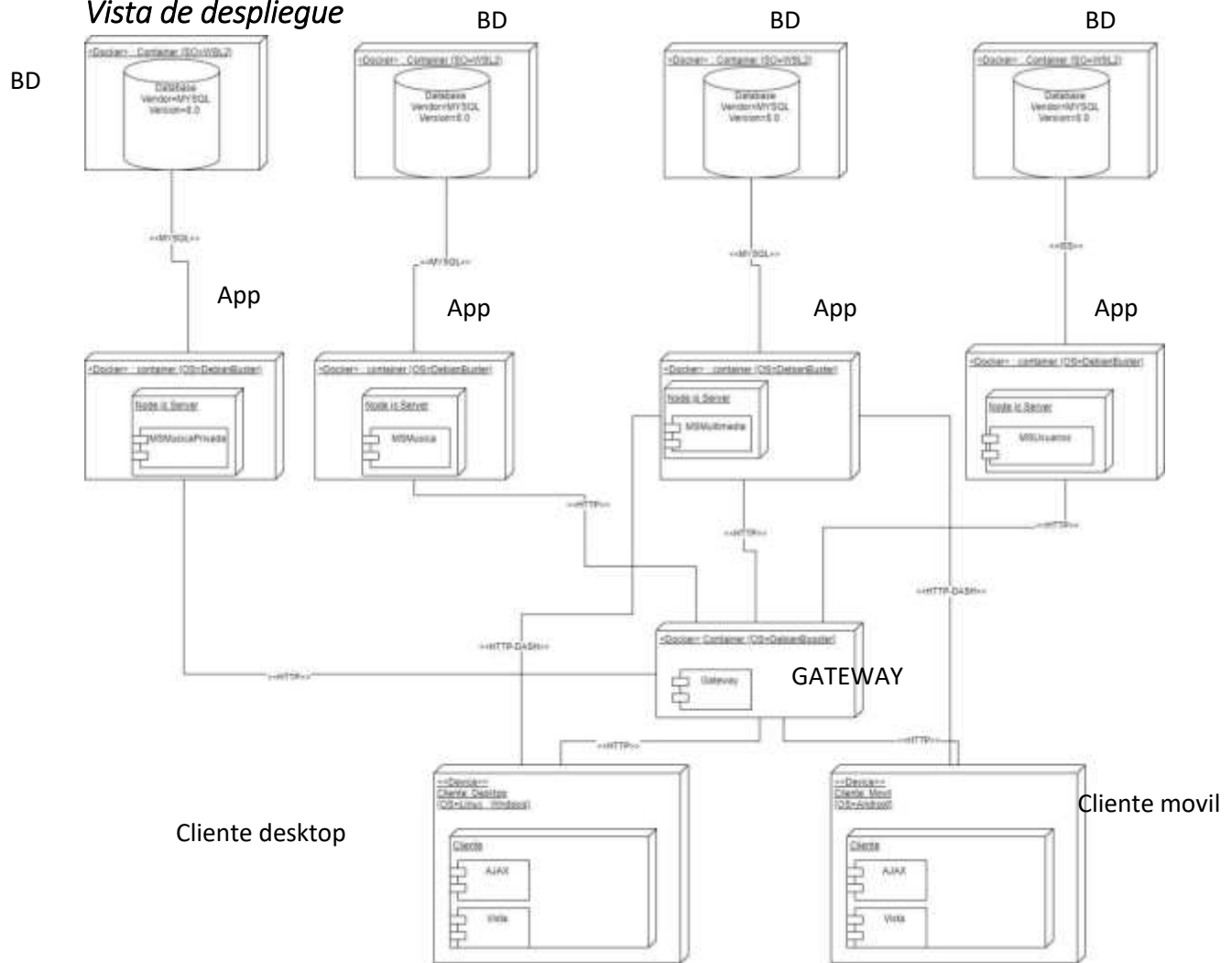
Vista de casos de uso



Vista lógica(Diagrama de clases)



Vista de despliegue

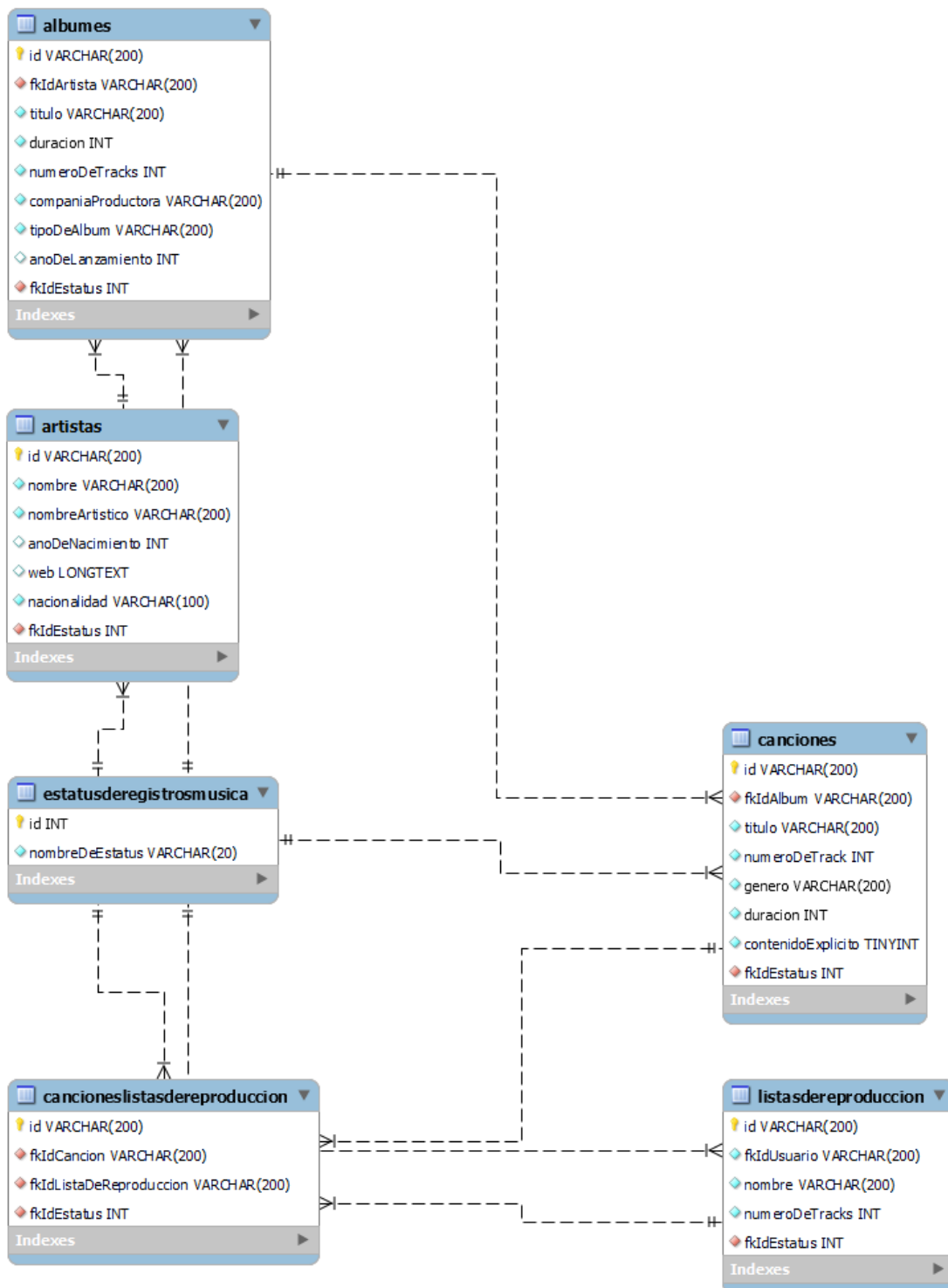


El diagrama anterior se muestran los componentes del sistema desplegados, los componentes para los servicios de musica,multimedia,musica privada y usuarios,los recuadros marcados con la etiqueta BD son contenedores con bases de datos Mysql, cada uno de estos contenedores esta ligado a un contenedor marcado con la etiqueta APP, por lo que cada servicio completo se compone de un contenedor BD junto con un contenedor APP, y sus correspondientes conexiones con el componente GATEWAY.Las imagenes docker seleccionadas para crear los contenedores se basaron en las imágenes oficiales para cada caso, en el caso de los contenedores BD estos utilizan la imagen oficial de Mysql, la cual opera con wsl2,en el caso de los contenedores APP de MSMusica,MSMultimedia y MSMusicaPrivada,se escogio la imagen oficial de nodejs, ya que estos servicios fueron implementados con nodejs,sin embargo para el servicio de MSMultimedia fue necesario instalar aplicaciones adicionales que son necesarias para el funcionamiento del servicio, en este caso ffmpeg y Gpac;ffmpeg se utiliza para la conversion de archivos de mp3 a m4a , necesaria para la segmentacion de archivos de musica en estandar DASH,y Gpac(una suite con

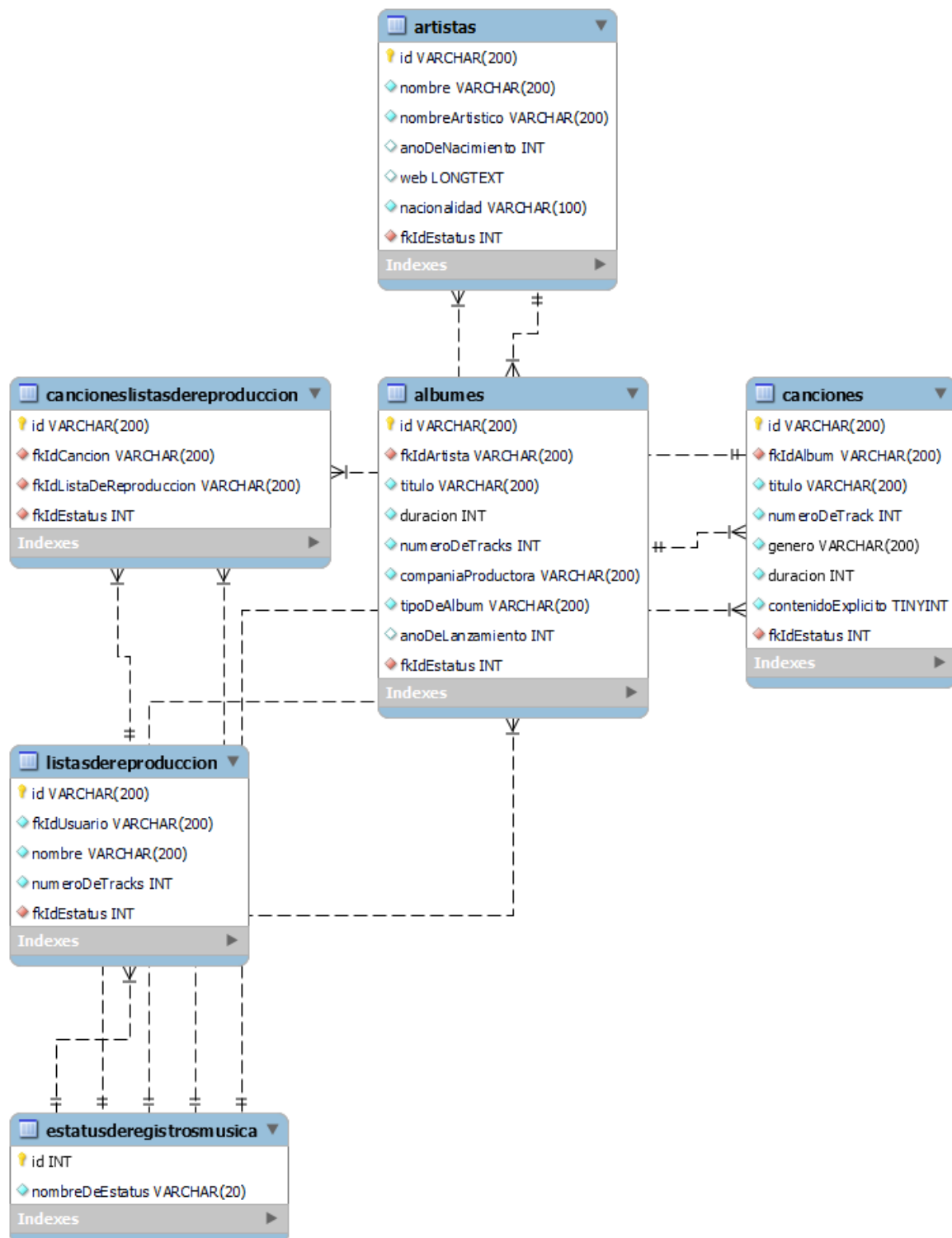
varias aplicaciones para edicion de audio y video) especificamente su herramienta MP4Box con la cual se hace la creacion de los segmentos de streaming requeridos por el estandar DASH junto con la creacion de su archivo de inidice para ligar todos los segmentos y poderlos obtener por http.

Modelo de datos

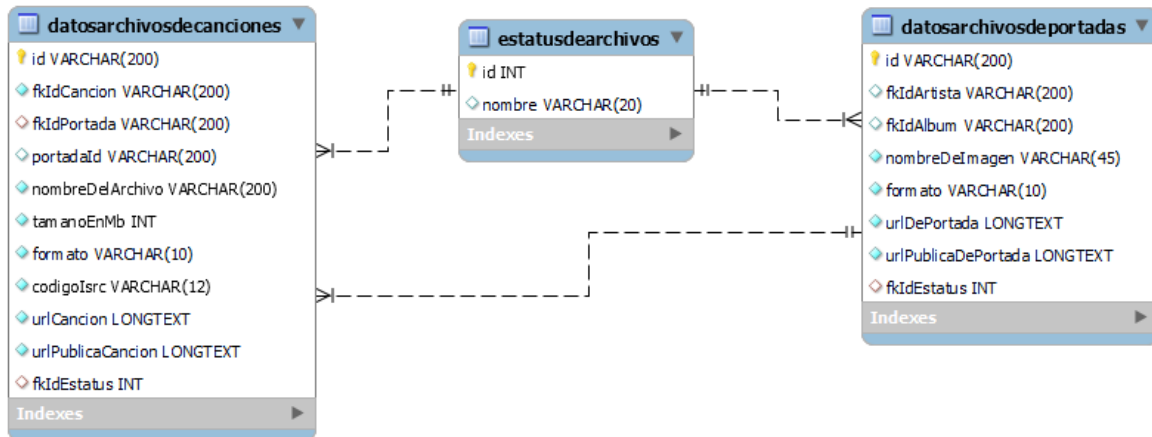
Modelo de datos BdUsuarios



Modelo de datos BdMusica y BDMusicaPrivada



Modelo de datos BdMultimedia



Descripciones de casos de uso

ID:	CU01
Nombre:	Suscribir
Autor(es):	Aarón Hernández Lara
Actor(es):	Usuario
Precondiciones:	Ninguna
Flujo normal:	1. – El Usuario da click en el botón para suscribirse en el menú inicial de LiberMusic . 2.- LiberMusic muestra un formulario para registrar al cliente. 3.- El Usuario rellena el formulario y da click en un botón para enviar. 4.- LiberMusic valida los datos ingresados 4.1 si los datos pasan la validación LiberMusic registra al cliente(VER EX01) 4.2 si los datos no pasan la validación LiberMusic notifica los errores al cliente y no registra al cliente. 5.- Fin del caso de uso.
Excepciones:	EX01 Error en la transacción de base de datos <ol style="list-style-type: none"> 1. LiberMusic deshace la transacción y muestra al Usuario el mensaje “algo salió mal inténtalo de nuevo”. 2. El usuario da click en el botón aceptar del mensaje mostrado en el paso anterior. 3. LiberMusic regresa al paso 3 del flujo normal.
Postcondiciones:	POST-1: el cliente es registrado en LiberMusic .

ID:	CU-02
-----	-------

Nombre:	Buscar música
Autor(es):	Aarón Hernández Lara
Actor(es):	Usuario
Precondiciones:	Ninguna
Flujo normal:	<ol style="list-style-type: none"> 1. El clie escribe una palabra en el buscador y da click en la opción para buscar. 2.- LiberMusic consulta las canciones,álbumes y artistas que coincidan total o parcialmente con la palabra ingresada y los muestra los resultados 3.- Fin del caso de uso
Postcondiciones:	POST-1: LiberMusic muestra los resultados de la búsqueda realizada.

ID:	CU03
Nombre:	Reproducir canción
Autor(es):	Aarón Hernández Lara
Actor(es):	Usuario común
Precondiciones:	Ninguna
Flujo normal:	<ol style="list-style-type: none"> 1. El Usuario común realiza CU-02(buscar musica). 2.El usuario común selecciona la canción que desea reproducir. 2. LiberMusic establece una conexión con el servidor, obtiene la canción seleccionada, y empieza la reproducción. 3. Termina el caso de uso
Excepciones:	<p>EX01 Se perdió la conexión con el servidor de streaming liberMusic y no logra reestablecerse la conexión</p> <ol style="list-style-type: none"> 1. LiberMusic intenta reconectar con el servidor 2 veces y no se logra la reconexión y LiberMusic muestra el mensaje “ Se ha perdido la conexión”. 2. El usuario común da click en aceptar al mensaje del paso anterior y espera a que se reestablezca la conexión. 3. Termina EX02 <p>EX02 Se perdió la conexión con el servidor de streaming de LiberMusic y LiberMusic logra reestablecer la conexión</p> <ol style="list-style-type: none"> 1. LiberMusic intenta reconectar con el servidor 2 veces y se reconecta

	<p>exitosamente</p> <p>2. El usuario común regresa a paso 3 del flujo normal</p>
Postcondiciones:	POST-1: el usuario común establece una conexión con el servidor y reproduce una canción.

ID:	CU04
Nombre:	Administrar listas de reproducción
Autor(es):	Aarón Hernández Lara
Actor(es):	Usuario comun
Precondiciones:	Ninguna
Flujo normal:	<p>1. – El Usuario común da click en el icono crear lista de reproducción</p> <p>2.- Usuario común muestra un formulario para poner el nombre de la lista de reproducción personal.</p> <p>3.- el Usuario común ingresa el nombre de la lista de reproducción personal y da click en un botón para aceptar.</p> <p>4.- LiberMusic valida el nombre de la lista de reproducción</p> <p>4.1 si el nombre es valido registra la lista de reproducción personal</p> <p>4.2 si el nombre no es valido notifica el error al cliente.</p> <p>5 Termina el caso de uso.</p>
Flujos alternos	<p>FA-01 El Usuario comun agrega una canción a una lista de reproducción.</p> <ol style="list-style-type: none"> 1. El usuario común entra a la lista de reproducción vacia. 2. LiberMusic muestra la lista de reproducción vacia con el botón agregar Canciones. 3. El usuario común dará clic en el botón “Agregar canciones” 4. LiberMusic mostrará un buscador de canciones. 5. El usuario común buscará la canción en el cuadro de búsqueda.

	<ol style="list-style-type: none"> 6. Liber music mostrara los resultados. 7. El usuario dará clic en las opciones de la canción. 8. Liber music desplegará las opciones de la canción seleccionada. 9. El usuario dará clic en agregar a la lista. 10. Liber music agregará la canción a la lista de reproducción. <p>FA-02 El usuario comun elimina una canción de la lista de reproducción.</p> <ol style="list-style-type: none"> 1. El usuario dará clic a la lista de reproducción 2. LiberMusic Mostrará Las canciones contenidas en la lista de reproducción. 3. El usuario dará clic en las opciones de la canción. 4. Liber music desplegará las opciones de la canción seleccionada. 5. El usuario dará clic en el botón eliminar. 6. Liber music eliminará la canción seleccionada. <p>FA-03 El usuario común elimina una lista de reproducción</p> <ol style="list-style-type: none"> 1. El usuario dará clic a la lista de reproducción 2. LiberMusic Mostrará la lista de reproducción seleccionada. 3. El usuario dará clic al botón gestionar. 4. LiberMusic Mostrará El botón "Eliminar lista de reproduccion" 5. El usuario dará clic a el botón "Eliminar lista de reproduccion" 6. LiberMusiis Mostrará una alrta con el mensaje "¿Esta seguro de hacer esto?". 7. El usuario dará clic en el botón aceptar 8. LiberMusic eliminará la lista de Reproducción seleccionada.
Excepciones	<p>EX1: No hay respuesta por parte del servidor.</p> <ol style="list-style-type: none"> 1. El cliente muestra que no fue posible conectar al servidor para crear la lista de reproducción. 2. El usuario común dará clic al botón de aceptar en la alerta 3. LiberMusic Regresa al paso numero 2 del flujo normal.
Postcondiciones:	<p>POST-1: LiberMusic registra una lista de reproducción personal.</p> <p>POST.2 LiberMusic agrega una canción a una lista de reproducción</p> <p>POST-3 LiberMusic Elimina una canción de una lista de reproducción</p> <p>POST-4 LiberMusic Elimina una lista de reproducción</p>

ID:	CU-05
Nombre:	Crear album
Autor(es):	Aarón Hernández Lara
Actor(es):	Artista
Precondiciones:	Ninguna
Flujo normal:	<p>1. – El artista da click sobre la opción administrar contenido.</p> <p>2.- LiberMusic muestra los álbumes que artista ha registrado previamente (se muestra vacío si aun no hay álbumes), las canciones que contiene, una opción para agregar canciones y una opción crear álbum.</p> <p>3.-El artista da click sobre la opción crear álbum.</p> <p>4.- LiberMusic muestra un formulario con los datos para registrar un album y un botón seleccionar portada y un botón subir canción.</p> <p>5 El artista llena los datos del formulario y da click en seleccionar portada</p> <p>6.- LiberMusic muestra un selector de archivos</p> <p>7 El cliente selecciona la imagen de portada que desea da click en subir</p> <p>8 LiberMusic valida el archivo</p> <p>8.1 si el archivo pasa la validación lo precarga para el envío</p> <p>8.2 si el archivo no pasa la validación notifica el error al cliente y rechaza el archivo</p> <p>9 si lo desea el artista lleva a cabo CU-06 (subir cancion).</p> <p>10.LiberMusic valida los datos del formulario</p> <p>10.1 Si se aprueba la validación LiberMusic registra el álbum en la base de datos y sube el archivo de la portada al servidor de streaming de liberMusic y muestra el mensaje “ y también el archivo de la canción en caso de haberse subido. Tu canción se registró correctamente” .</p> <p>10.2 Si no se aprueba la validación LiberMusic muestra al artista el mensaje de error “Hay errores en tus datos” y le muestra cuales son los campos con errores.</p> <p>11 Termina el caso de uso</p>
Excepciones	EX01 Error al registrar la cancion

	<ol style="list-style-type: none"> 1. LiberMusic deshace la transacción y evita que se suba el archivo relacionado y muestra el mensaje “ no se ha podido subir la canción”. 2. El artista da click en aceptar en el mensaje del paso anterior. LiberMusic regresa al paso 9 del flujo normal
Postcondiciones:	POST-1: ArsMusic registra una canción y sube su archivo asociado al servidor de streaming de LiberMusic

ID:	CU-06
Nombre:	Subir canción
Autor(es):	Aarón Hernández Lara
Actor(es):	Artista
Precondiciones:	Ninguna
Flujo normal:	<ol style="list-style-type: none"> 1. – El artista da click sobre la opción administrar contenido. 2.- LiberMusic muestra los álbumes que artista ha registrado previamente, las canciones que contiene y una opción para agregar canciones y junto a cada canción una opción para eliminarlas. 3.-El artista da click sobre la opción agregar canción. 4.- LiberMusic muestra un formulario con los datos para registrar una canción y un botón seleccionar archivo y un botón subir canción. 5 El artista llena los datos del formulario y da click en seleccionar archivo 6.- LiberMusic muestra un selector de archivos 7 El cliente selecciona el archivo da click en subir 8 LiberMusic valida el archivo 8.1 si el archivo pasa la validación lo precarga para el envío 8.2 si el archivo no pasa la validación notifica el error al cliente y rechaza el archivo 9 El artista da click en la opción subir canción. 10.LiberMusic valida los datos del formulario 10.1 Si se aprueba la validación LiberMusic registra la canción en la base de datos y sube el archivo de la canción al servidor de streaming de liberMusic y muestra el mensaje “Tu canción se registro correctamente”. 10.2 Si no se aprueba la validación LiberMusic muestra al artista el mensaje de error “Hay errores en tus datos” y le muestra cuales son los campos con errores. 11 Termina el caso de uso

Excepciones	<p>EX01 Error al registrar la canción</p> <ol style="list-style-type: none"> 3. LiberMusic deshace la transacción y evita que se suba el archivo relacionado y muestra el mensaje “ no se ha podido subir la canción”. 4. El artista da click en aceptar en el mensaje del paso anterior. <p>LiberMusic regresa al paso 9 del flujo normal</p>
Postcondiciones:	POST-1: ArsMusic registra una canción y sube su archivo asociado al servidor de streaming de LiberMusic

ID:	CU-07
Nombre:	Eliminar canción
Autor(es):	Aarón Hernández Lara
Actor(es):	Artista
Precondiciones:	Ninguna
Flujo normal:	<ol style="list-style-type: none"> 1. – El artista da click sobre la opción administrar contenido. 2.- LiberMusic muestra los álbumes que artista ha registrado previamente, las canciones que contiene y una opción para agregar canciones y junto a cada canción una opción para eliminarlas. 3.-El artista da click sobre la opción eliminar canción 4.- LiberMusic muestra el mensaje “Estas seguro que deseas eliminar esta canción” . 5 El artista da click en aceptar en el mensaje del paso anterior. 6.- LiberMusic music elimina el registro de la base de datos de la canción y el archivo relacionado muestra el mensaje “ la canción ha sido eliminada” . 7 .-El artista selecciona aceptar en el mensaje del paso anterior. 8.- Termina el caso de uso.
Flujos alternos:	<p>EX01 Error al borrar la canción</p> <ol style="list-style-type: none"> 5. LiberMusic deshace la transacción y evita el borrado del archivo relacionado y muestra el mensaje “ no se ha podido borrar la canción”. 6. El artista da click en aceptar en el mensaje del paso anterior. 7. LiberMusic regresa al paso 3 del flujo normal

Excepciones	<p>EX1: No es posible Conectarse al servidor.</p> <ol style="list-style-type: none"> 1. El cliente muestra que no fue posible conectar al servidor. 2. El usuario común dará clic al botón de aceptar en la alerta 3. LiberMusic Regresa al paso numero 2 del flujo normal.
Postcondiciones:	POST-1: LiberMusic elimina una canción y borra su archivo asociado en servidor de streaming de liberMusic

ID:	CU-08
Nombre:	Validar canción
Autor(es):	Aarón Hernández Lara
Actor(es):	Supervisor de contenido
Precondiciones:	Ninguna
Flujo normal:	<p>1.- LiberMusic muestra las canciones que los artistas han registrado previamente y que tienen el estado “no validado”.</p> <p>3.-El Supervisor de contenido da click sobre la canción que desea validar</p> <p>4.- LiberMusic muestra el mensaje “Estas seguro que desea validar la canción”</p> <p>5 El artista da click en aceptar en el mensaje del paso anterior.</p> <p>6.- LiberMusic cambia el estado de la canción a “validado” y la elimina de los resultados”.</p> <p>7.- Termina el caso de uso.</p>
Flujos alternos:	<p>EX01 Error al validar la canción</p> <ol style="list-style-type: none"> 1. LiberMusic deshace la transacción, mantiene la canción en los resultados y muestra el mensaje “ no se ha podido validar la canción”.

	2. El artista da click en aceptar en el mensaje del paso anterior. 3. LiberMusic regresa al paso 2 del flujo normal
Excepciones	EX1: No es posible Conectarse al servidor. 4. El cliente muestra que no fue posible conectar al servidor. 5. El usuario común dará clic al botón de aceptar en la alerta 6. LiberMusic Regresa al paso numero 2 del flujo normal.
Postcondiciones:	POST-1: LiberMusic cambia el estado de una canción a “validado”

Construcción

Selección justificada de la tecnología

Aplicación de cliente rico

El cliente fue desarrollado en java para android, principalmente debido a que es la tecnología base que se imparte en el curso, pero por otro lado el desarrollo en java permite hacer aplicaciones nativas de android, que como punto de partida para aprender desarrollo móvil me pareció mas adecuado, ya que permite conocer una plataforma especifica de desarrollo mas a profundidad, lo cual para términos de aprendizaje para esta ocasión me pareció mas importante. Por otro lado las librerías necesarias para la reproducción de música están disponibles en android, y tienen una documentación relativamente buena, lo cual no estaría muy seguro en otras plataformas, y tampoco con pwa, sin mencionar por supuesto que IOS esta fuera del alcance por el momento debido a que es muy costoso desarrollar para esta plataforma debido que requiere equipos de la marca Apple y licencias para ello. Sin embargo si hubiera la posibilidad de desarrollar para IOS también me habría inclinado por android debido a la gran cantidad de dispositivos que existen con este sistema operativo, muy superiores a la cantidad de usuarios IOS.

Servidor

Las tecnologías utilizadas en este proyecto para el backend fueron .Net Core y nodejs en arquitectura microservicios(requisito del curso) con typeScript. La selección de .NET Core fue debido en mayor medida a que es la tecnología con la que se llevo a cabo el curso, es mas bien el punto de partida del proyecto, por otro lado la experiencia del equipo de desarrollo del proyecto es mayor en esta tecnología por lo que permitiría un desarrollo mas ágil y a su vez es una tecnología con una buena cantidad de documentación lo cual es de ayuda a la hora de resolver problemas durante el desarrollo. Por otro lado nodejs se selecciono por varios motivos, el primero de ellos es que la funcionalidad central del proyecto, es decir la reproducción de musica, es mas afin para desarrollarse en node js, si se compara una implementación del servidor de streaming en .net, esta resulta mucho mas compleja que una en node js, se requiere una cantidad de código considerablemente mayor para implementar el servidor de streaming en .net core y por otro lado

la documentación disponible para hacer servicio de streaming con estándar dash en .net core es básicamente inexistente, si bien tampoco abunda en nodejs si se logra encontrar mas información. Otra razón por la que se escogió nodejs es la facilidad que tiene para crear API REST, es mucho mas intuitivo y legible leer y crear un API REST en nodejs y aun mas usando TypeScript ;mientras en .net core el url de api, se tiene que deducir del nombre del controlador, sin el postfijo controller mas el nombre de la ruta en el decorador HTTP de cada método, en nodejs se exporta un modulo que explícitamente contiene las urls y sus manejadores, de la siguiente forma: `this.router.post('/crear',express.json(),this.registrarArtista)`, el primer parámetro es la ruta relativa explicita del servicio en cuestión en relación a una url general definida en otro archivo, por ejemplo la siguiente `this.express.use('/artistas',artistasApi)`, asi la url completa seria <http://ip/artistas/crear, lo> cual resulta mas explicito y legible pues pueden verse claramente los nombres de las rutas, el método HTTP utilizado y la función que se encarga de manejar la petición. Por otro lado al ser nodejs también permite una forma simple de manejar la asincronia con `async` y `await` apartir de ECMAScript 2017, con la ventaja adicional(según sea el caso)de la versatilidad de javascript con respecto al tipado de datos, lo cual puede observarse en la facilidad para crear estructuras de datos json sin hacer modelos estáticos especiales para estas,lo cual brinda una mayor facilidad para el desarrollo y en caso de requerirse un nivel mas estricto de tipado, también puede conseguirse por medio de typescript. Por último pero no menos importante, otra razón es la importancia que nodejs tiene actualmente en el mundo del desarrollo, teniendo una aceptación cada vez mayor lo que la convierte en una tecnología de uso muy extendido, por lo cual aprenderla y manejarla es de bastante utilidad.

Herramientas de despliegue y control de versiones

Para control de versiones se utilizó github, debido a que es un servicio de uso generalizado y con mucha documentacion, y por supuesto cuenta con todo lo necesario para el desarrollo del proyecto y la materia en general. En cuando a la herramienta de despliegue utilizada, es decir, docker , en especifico docker-compose,también se escogio por su gran documentacion y relativa facilidad de uso, sobre todo para la necesidades del proyecto, ya que permite orquestar contenedores con relativa facilidad en casos donde no se requieren orquestaciones muy complejas o personalizadas lo cual es justo lo necesario para este proyecto, ya que herramientas mas completas como kubernetes no serían realmente explotadas en su totalidad para el caso de este proyecto.

Prácticas de construcción realizadas

Nombres de clases,métodos y variables descriptivos

Separación de responsabilidades

Estrategia de despliegue

Para el despliegue de la aplicación se utilizo docker, por medio de la herramienta docker-compose, para orquestar todos los contenedores ,los servicios de música, multimedia y música privada están desplegados en la nube en distintos proveedores. De manera local corren los servicios de suscripción y el api gateway.

Conclusiones.

Este proyecto fue una buena oportunidad para acercarme al desarrollo móvil y ver las sustanciales diferencias con respecto al desarrollo de apps para pc. Otro punto muy importante, el mas importante al menos para mi, es el conocimiento y acercamiento sobre devops , la integración continua y el despliegue continuo y el control de versiones son aspectos sumamente importantes hoy en día y tener conocimiento sobre ello, conocer herramientas para esto es sumamente valioso, pues se convierte en una ventaja competitiva como estudiante en la mayoría de los casos. El conocimiento sobre estas practicas y las herramientas para llevarlas a acabo son algo sumamente valioso.

Este proyecto también me permitió conocer mas a fondo el funcionamiento de los sistemas distribuidos, sobre todo por el hecho de que varios servicios fueron desplegados en la nube, lo cual es un entorno mas realista respecto a sistemas distribuidos. Hay cosas que mejorar en el proyecto, como por ejemplo refactorización, se espera que esto sea corregido en tiempo futuro. Por otro lado también fue posible darme cuenta de la enorme cantidad de recursos de que se requieren para desarrollar microservicios y la importancia que esto tiene en un entorno de producción real y para el desarrollo, ya que es una consideración que no se puede pasar por alto.