

APOLLOFY

**PRODUCT REQUIREMENTS
DOCUMENT**

ÍNDICE

[Referencia](#)

[Planteamiento](#)

[Alcance del proyecto](#)

[Objetivos](#)

[Gestión de Proyecto](#)

[¿Qué es Scrum?](#)

[¿Qué es un Sprint?](#)

[¿Cómo se definen las tareas en Scrum?](#)

[¿Existen distintos roles en SCRUM?](#)

[¿Qué meetings se realizarán para gestionar el proyecto?](#)

[¿Cómo se distribuirán los sprints para el proyecto Apollofy?](#)

[Tareas a realizar durante el Project Planning](#)

[Tareas a realizar durante los Sprints](#)

[Tareas a realizar durante la presentación](#)

[Descripción del Producto](#)

[State of the Art](#)

[Objetivo del Producto](#)

[¿Qué es Apollofy?](#)

[¿Por qué el nombre Apollofy?](#)

[¿Cuál es el futuro de Apollofy?](#)

[¿Qué funcionalidades incluirá Apollofy?](#)

[Epics e Historias de Usuario MUST HAVE](#)

[Epics](#)

[Historias de Usuario](#)

[Mockups-Stories Orientation](#)

[Pantalla Home principal](#)

[Gestión de la cuenta](#)

[Publicación, visualización y gestión de canciones](#)

[Reproductor de Canciones](#)

[Búsqueda de canciones, usuarios, playlists y álbumes](#)

[Publicación, visualización y gestión de playlists](#)

[Visualización y gestión de usuarios/artistas](#)

[Visualización de estadísticas](#)

[Dataflow Diagrams](#)

[Registro usuario](#)

[Login Usuario](#)

[Subida de canción](#)

[Apéndice I. Arquitectura de Software](#)

[Apéndice II. Arquitectura de Base de Datos](#)

[Apéndice III. Arquitectura de Base de Datos no Relacional](#)

[Apéndice IV. Arquitectura de la API](#)

[Registro de usuario](#)

[Inicio de sesión](#)

[Creación de canciones](#)

[Creación de playlists](#)

[Apéndice V. Backend - Frontend development independency](#)

[Apéndice VI. Epics y User Stories opcionales](#)

[Historias de Usuario opcionales](#)

[Mockups-Stories Orientation](#)

[Registro y Autenticación](#)

[Pantalla Home principal](#)

[Publicación, visualización y gestión de canciones](#)

[Reproductor de Canciones](#)

[Búsqueda de canciones, usuarios, playlists y álbumes](#)

[Publicación, visualización y gestión de playlists](#)

[Visualización y gestión de usuarios/artistas](#)

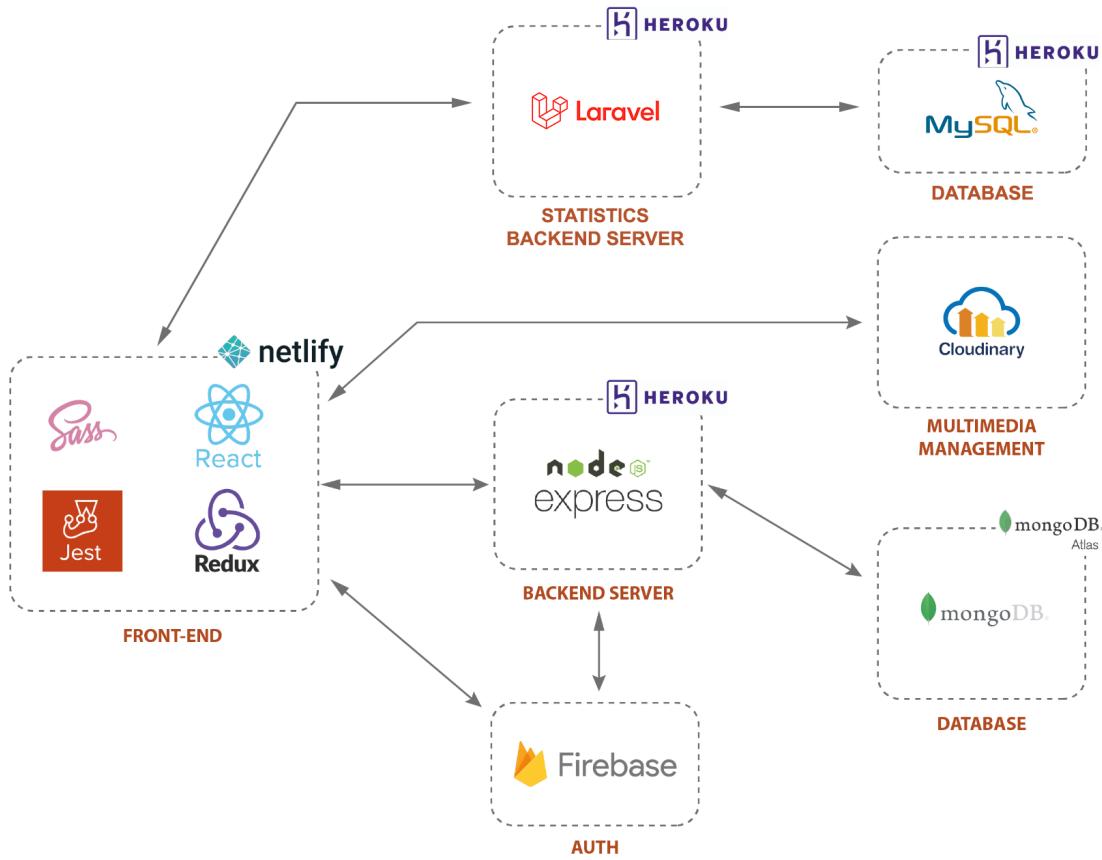
Referencia

Target Release	1.0
Repository	<p><u>Repository semilla</u>, infraestructura para empezar a desarrollar el proyecto con:</p> <ul style="list-style-type: none"> • Funcionalidades básicas de <i>Registro</i> y Autenticación con Firebase (<i>Login, Reset Password</i>) • <u>Template de Github Project</u> con todas las User Stories con Must Have y los <u>4 Sprints</u> + <u>User Stories</u> asignadas a cada uno de los 4 Sprints. • <u>4 User Stories implementadas de ejemplo</u> en las Feature Branches <u>AWS API Apollofy</u> <p>Cada <i>equipo</i> generará su propio repositorio como público en GitHub. El equipo de alumnos proporcionará acceso al repositorio de GitHub del proyecto al equipo académico, para poder realizar el feedback:</p> <ul style="list-style-type: none"> • <u>https://github.com/natanverdes</u> • <u>https://github.com/Pau111111</u> • <u>https://github.com/manuelsanchosanchez</u> • <u>https://github.com/JCarri14</u> • <u>https://github.com/cristiam86</u>
Document status	<i>Released</i>

Planteamiento

Este proyecto propone la ideación y creación de una plataforma web de música basada en Spotify/SoundCloud. Los alumnos se encargarán de desarrollar un sistema cliente-servidor, empleando **NodeJS**, **Express**, **Laravel**, **MySQL**, **MongoDB**, **Firebase** y **Cloudinary** para la implementación del back-end, y **ReactJS + Redux** para el front-end.

De esta manera, el despliegue del sistema es el siguiente:



Alcance del proyecto

Este documento de requerimientos representa la especificación de un producto realista desde el punto de vista empresarial. El conjunto global de requisitos se incluye en una planificación extensa que abarca más allá del tiempo estimado en el contexto del máster. La visión del equipo académico es transmitir a los alumnos un reto que les prepare de verdad para su nueva etapa profesional una vez concluido el máster.

Considerando que es un proyecto sofisticado y complejo, hemos separado claramente las funcionalidades **Must Have** de las funcionalidades que son opcionales: **Should Have** y **Could Have**. ¿Cuál es el objetivo de esta diferenciación? El conjunto de tareas Must Have se han planificado, de tal manera, que se puedan completar en el marco temporal de los 4 sprints de una semana estimados para el proyecto.

Desde el equipo académico, estamos muy ilusionados en que la vida de vuestro proyecto se desarrolle mucho más allá del máster. Vuestra plataforma de música se implementará siguiendo una licencia open source, que posibilitará que sigáis colaborando en equipo para mejorar y evolucionar el proyecto, disfrutando de una relación enriquecedora con vuestros compañeros del máster. Las [funcionalidades opcionales](#) ofrecen un Roadmap para lograr un resultado espectacular que se difunda en vuestro LinkedIn y otras redes sociales.

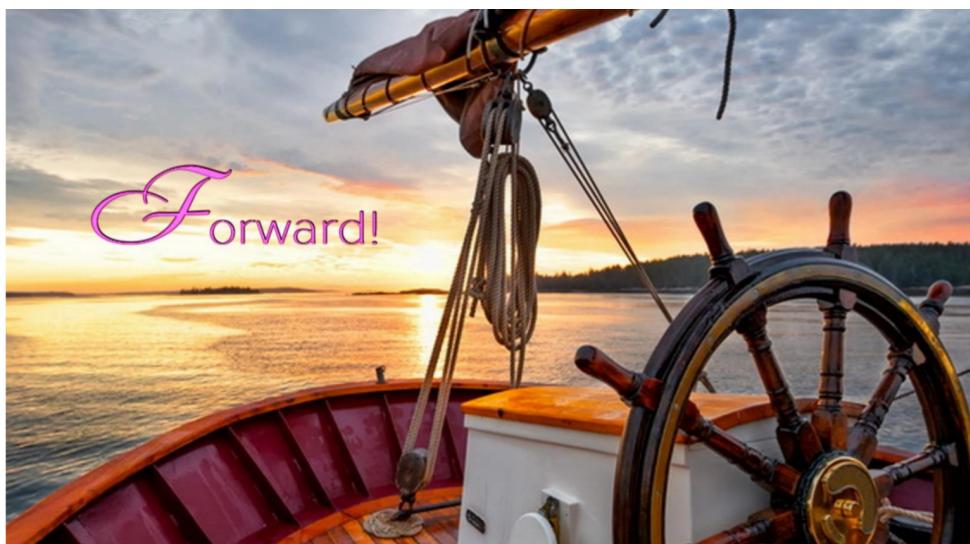
Objetivos

- Aprender a desarrollar un proyecto profesional en equipo, aplicando [SCRUM](#) como una de las metodologías **Agile más usadas** en el mercado.
- Profundizar y fortalecer los conocimientos en desarrollo back-end con tecnologías como **Node, Express, MongoDB** (stack [MERN](#)) y de PHP con [Laravel](#) y para la parte de base de datos con **MySQL**.
- Profundizar y fortalecer los conocimientos en desarrollo front-end con **React** y **Redux**.
- Enriquecer el Portafolio personal con un proyecto **enterprise-grade open source** que refleje la calidad del aprendizaje a nivel tecnológico y de **soft skills**.

Gestión de Proyecto

El éxito de un proyecto de software no solamente depende de una buena calidad del código. La alineación del equipo y la distribución correcta de tareas y fechas de entrega es indispensable para un completo éxito. Por ello, es importante que el proyecto esté correctamente separado en pequeñas tareas que puedan irse distribuyendo entre los distintos integrantes del equipo, y que estas se vayan completando paulatinamente a lo largo de las semanas del proyecto.

Existen distintas metodologías para gestionar proyectos. Para este proyecto en concreto aplicaremos la metodología [Scrum](#) gracias a la guía de un experto en [Agile](#).



¿Qué es Scrum?

[Scrum](#) es un framework de trabajo que permite mejorar el trabajo colaborativo en equipo y a su vez una metodología dentro de la mentalidad Agile. En Scrum, lo importante es trabajar de una manera que permite iterar las entregas del desarrollo. En cada una de estas entregas, se revisa la situación, se aclaran y corrigen desviaciones, y se prepara un nuevo conjunto de tareas para la siguiente iteración, con la decisión propia de cada componente del equipo de responsabilizarse en cada una de las tareas para la siguiente entrega.

¿Qué es un Sprint?

Pero, ¿qué queremos decir con una iteración? En Scrum no se realiza un flujo de trabajo constante (como si fuésemos cogiendo a diario tareas de un gran bote de tareas hasta finalizarlas), sino que este se realiza mediante iteraciones, llamadas [Sprints](#). Los Sprints son períodos breves de tiempo fijo en los que un equipo trabaja para completar unas tareas previamente escogidas. En cada Sprint el equipo irá trabajando, entonces, sobre las tareas previamente decididas. Por lo general, estos Sprints suelen ser de dos semanas (standard) o de una semana (con mayor control).

Una vez finalizado el Sprint, el equipo realizará una serie de reuniones en las que revisará y hará una demostración interna de lo que se ha desarrollado, confirmará la finalización de todas las tareas marcadas como completadas, y se volverá a distribuir un nuevo conjunto de tareas para la siguiente iteración.



¿Cómo se definen las tareas en Scrum?

En Scrum, un proyecto acaba disecándose en un conjunto de tareas, llamadas [Historias de Usuario](#) (User Stories). Se llaman historias de usuario porque siempre van enfocadas a una funcionalidad particular que el usuario podrá disfrutar. Estas [Historias de Usuario](#) se agruparán en [Epics](#), maneras de agrupar las tareas en un gran bloque que pretende solventar una funcionalidad principal. Un ejemplo de una historia de usuario sería “El usuario podrá registrarse en el sistema suministrando los datos: Nombre, Apellido, E-mail, Username y Password”, dentro del Epic “Registro y Autenticación”. Para el correcto desarrollo de estas Historias de Usuario, se deberán desarrollar ciertas subtareas técnicas, como por ejemplo, la de desarrollar una API de registro o la de desarrollar una vista de registro.

Las tareas en Scrum, en vez de ser estimadas por horas (horas de trabajo), se estimarán mediante un número de puntos de historia o *story points*. Este número será una manera de añadir peso a cada una de las tareas, entendiendo que las de mayor peso requieren más esfuerzo, y las de menor peso requieren menos esfuerzo. En SCRUM, este número de story points suele definirse a partir de la secuencia de Fibonacci. Además, suele existir un límite máximo de story points, para no aceptar la definición de tareas que pudieran ser demasiado extensas o demasiado complejas, obligando a dividir estas tareas en varias de menor complejidad. La secuencia a emplear sería:

1, 2, 3, 5, 8, 13, 21

Vamos a ver algunos ejemplos de tareas definidas mediante SCRUM:

User Story title	User Story description	Story Points	Priority
Autentificación de Usuario	El usuario podrá iniciar sesión introduciendo sus credenciales de acceso al sistema: Username y password	5	MUST HAVE
Dar “Me gusta” a una canción	El usuario podrá dar “Me gusta” a las canciones, guardándolas automáticamente en el apartado “Mis canciones”	13	MUST HAVE

¿Existen distintos roles en SCRUM?

Sí. Existen ciertos roles en la metodología SCRUM que deberemos conocer antes de iniciar el proyecto. Vamos a ver cuáles son:

- **Equipo de desarrollo:** este está formado por todo el equipo de desarrolladores de software, diseñadores, etc. En este proyecto, los **5 integrantes del equipo** de software serán el equipo de desarrollo.
- **Scrum Master:** El Scrum Master es el encargado de liderar el uso de la metodología Scrum dentro del equipo, y de solventar cualquier duda o bloqueo durante el transcurso del desarrollo de las tareas, para que el equipo consiga llegar correctamente a cada una de las finalizaciones de los sprints. Cualidades como el liderazgo, las habilidades comunicativas, la empatía y la resolución de problemas serán muy valiosas para este tipo de rol. Deberá estar al corriente de las tareas del equipo en el día a día, y ayudar en cualquier momento que un integrante del equipo dependa de otro, por ejemplo. La figura de Scrum Master recaerá sobre el equipo académico.

- **Propietario de producto:** se trata del encargado de definir las funcionalidades del producto tecnológico que se va a desarrollar. Este debe tener una gran capacidad de visión para poder visualizar el producto dentro del mercado, y para poder organizar y diseccionar detalladamente todas las funcionalidades y requerimientos que hará cada una de las vistas, etc. También es el encargado de priorizar el desarrollo de estas funcionalidades dentro de las fases del proyecto.
 - *En este proyecto:* El equipo académico ha definido y priorizado previamente la mayoría de las funcionalidades del proyecto Apollofy generando una tabla de historias de usuario en este documento. En este listado se pueden encontrar funcionalidades obligatorias, necesarias y opcionales. De todas formas, el equipo de desarrollo puede decidir añadir nuevas funcionalidades tratándolo previamente con el equipo académico. De esta forma, ¡los propietarios del producto sois también vosotros!

¿Qué meetings se realizarán para gestionar el proyecto?

Para un correcto desarrollo del proyecto hace falta una muy buena comunicación de equipo. Sin realizar reuniones ni disponer de buenas herramientas de comunicación, cada integrante del equipo iría desarrollando por su cuenta sin tener en consideración lo que estén realizando sus compañeros de equipo. La metodología Scrum propone especiales momentos en los que reunirse y tratar diversos temas. Veámoslos a continuación. El proyecto se gestionará realizando los siguientes meetings, en los cuales el equipo académico también estará presente:

- Una *reunión semanal* de **1h** en la que se cerrará el sprint anterior y se iniciará un nuevo sprint
 - **(30 min) Sprint Review:** En este punto, el equipo debería haber finalizado todas las tareas propuestas para la semana. Se realizará una revisión de lo que se ha ido desarrollando durante la semana, el equipo irá mostrando *demos* de la parte del producto que haya ido trabajando para ver el progreso semanal. Primeramente se iniciará la revisión de las tareas marcadas como completadas, luego en progreso, y finalmente de las tareas pendientes.
 - **(30 min) Sprint Planning:** En este punto, el equipo ya ha acabado de revisar las tareas completadas, y empezará a valorar las nuevas tareas que se realizarán durante la siguiente semana. Aquí se añadirán tanto tareas que no se hayan completado en el Sprint anterior, como tareas que no han pasado el proceso de validación, como tareas nuevas que provienen de la propia planificación del producto (véase la tabla de historias de usuario al final de este documento).
- Una *reunión diaria* de **15 min** en la que se iniciará el día en equipo
 - **(15 min) Daily Standup** al inicio de la jornada: en este punto todos los integrantes, por orden, irán hablando de su situación durante el desarrollo del proyecto. El objetivo es encontrar que todo el equipo está alineado y que no están habiendo problemas de comunicación o de distribución de tareas. Para ello, cada uno de los integrantes irá respondiendo a cada una de estas tres preguntas:
 - ¿Qué hiciste ayer?
 - ¿Qué harás hoy?
 - ¿Tienes algún impedimento?

¿Cómo se distribuirán los sprints para el proyecto Apollofy?

El equipo académico ha estimado y repartido las diferentes tareas del proyecto en Sprints semanales. La duración de cada Sprint, entonces, será de **una semana**. Estas pueden visualizarse en el apartado *Epics* y *User Stories* de este documento. Pero la **estimación de las tareas** con respecto a cada semana de trabajo es relativa, y puede irse modificando mientras se vaya desarrollando el proyecto. Por ejemplo, si el equipo acabara todas las tareas del Sprint 1 a mitad de la semana, se podría revisar y añadir tareas del Sprint 2. Esto no suele ser propio de la metodología Scrum, pero añadiremos valores del propio sistema Agile, como es el de la adaptación, para mejorar la situación dependiendo de las necesidades de cada semana.

La **configuración de los equipos** habrá sido previamente propuesta y aprobada. Esta propuesta deberá estar bien equilibrada en cuanto a las habilidades en el desarrollo de software. Por ejemplo, un equipo bien equilibrado dispondría de 2 front-ends, 2 back-ends y 1 full-stack. Cada alumno decidirá su perfil antes de empezar el proyecto. El hecho de escoger un perfil especializado no implica que no pueda escoger algunas tareas de otros perfiles. La elección del perfil ayudará principalmente a que los equipos estén bien equilibrados, y a una adecuada asignación de tareas.

En el proyecto Apollofy se trabajará mediante los siguientes Sprints en las siguientes semanas:

Week 1	Week 2	Week 3	Week 4	Week 5
Project Planning + Sprint 1	Sprint 2	Sprint 3	Sprint 4	Presentación del proyecto

En primer lugar, los 2 primeros días del proyecto se trabajará en el Project Planning e ir arrancando el Sprint 1. Durante las siguientes semanas, se irá trabajando en cada uno de los sprints, realizando las correspondientes reuniones de revisión y planificación. Finalmente, el último día del proyecto se realizará una presentación y demostración de este.

Tareas a realizar durante el Project Planning

En los primeros días del proyecto, se realizarán una serie de actividades por parte de cada equipo para disponer de una buena planificación del desarrollo del proyecto. Como ya se ha trabajado con la herramienta Github Projects, seguiremos empleando esta plataforma. Todos los equipos tendrán que realizar durante estos días las siguientes actividades:

- Sesión de producto: Reunión en la que trataremos cuál es la visión general del producto a desarrollar, y un vistazo general de las funcionalidades principales. Apollofy incluye funcionalidades de producto muy relacionadas con funcionalidades que actualmente ofrecen Soundcloud o Spotify, por lo que no será muy difícil entenderlo. También se realizará una explicación de las vistas y pantallas que incluirá la aplicación.
- Sesión de gestión de proyecto: Puntos principales para cómo gestionar el desarrollo del proyecto.
 - Metodología SCRUM: explicación de conceptos como User Stories, Epics, Sprints, Sprint Planning, Sprint Review, Daily Standups, etc.
 - Github: revisión para cómo crear un repositorio, añadir colaboradores, etc.
 - Github Projects: revisión de la herramienta para aprender cómo se crean User Stories (issues), Sprints (milestones), vinculación de issues con los milestones, cómo asignar participantes a las issues, etc.

A partir de estas reuniones iniciales, el *equipo* estará listo para **volcar** en su proyecto de *Github Projects* el listado de *User Stories* que aparece en el [proyecto template](#), y que también están disponibles al final de este documento. Este listado de tareas está dividido mediante tres tipos de prioridades: “must” (obligatorias), “should” (deberían) y “could” (podrían). Durante cada Sprint, el equipo irá decidiendo en qué tareas se va centrando a partir de esta definición de prioridades. El equipo irá trabajando primeramente en las tareas “must”, luego “should” y finalmente, si dispusiera de tiempo, de las “could”. El equipo decidirá si ir volcando a *Github Projects* únicamente las user stories con prioridad “Must Have” o ir volcando también las user stories con prioridad de “Should have” o “Could have”. Se recomienda que readactéis las User Stories en inglés para potenciar vuestro portfolio de cara a oportunidades internacionales.

Enlace: <https://github.com/assembler-school/apollofy-music-project/milestones>

		Sort ▾
4 Open	0 Closed	
Sprint 2	0% complete 12 open 0 closed	
Due by April 23, 2021 Last updated 8 minutes ago		Edit Close Delete
Funcionalidades:		
· Subida, reproducción y gestión de canciones		
Sprint 4	0% complete 10 open 0 closed	
Due by May 07, 2021 Last updated 13 minutes ago		Edit Close Delete
Funcionalidades:		
· Visualización y seguimiento de usuarios		
· Estadísticas de las entidades del sistema		
Sprint 3	0% complete 10 open 0 closed	
Due by April 30, 2021 Last updated 18 minutes ago		Edit Close Delete
Funcionalidades:		
· Creación y administración de playlists		
Sprint 1	0% complete 8 open 0 closed	
Due by April 16, 2021 Last updated 7 days ago		Edit Close Delete
Funcionalidades:		
· Registro y autenticación de usuarios		
· Gestión del perfil del usuario		

Tareas a realizar durante los Sprints

Una vez finalizado el volcado de historias de usuario, los equipos podrán iniciar mediante una reunión de **Sprint Planning** el Sprint semanal. Como se ha comentado anteriormente, durante cada uno de los Sprints el equipo irá trabajando en las tareas asignadas para esa semana. Primeramente, se centrarán en las tareas “Must Have”. Si las acabaran, podrán avanzar entonces con las tareas de “Should Have” y finalmente de “Could Have”. Si no se llegara a resolver las tareas de “Should Have” o de “Could Have” en el sprint en curso y este finalizara, estas tareas quedarán congeladas y se decidirá dar prioridad a las tareas entrantes del siguiente Sprint en vez de centrarse en las tareas menos prioritarias.

Cada día realizaréis un meeting por equipo de 15 minutos llamado **Daily Standup** para poner en común con todo el equipo las tareas que se van realizando y tratar en caso necesario cualquier problema que se esté afrontando. Al finalizar la semana, volveréis a realizar una reunión, en la que se realizará un **Sprint Review** de las tareas finalizadas la semana anterior, y un **Sprint Planning** de nuevo para planificar las tareas de la próxima semana. Esta metodología se irá aplicando durante los 4 sprints que dura el proyecto de desarrollo.

A lo largo del proyecto se realizarán un total de 23 user stories “Must Have”, distribuidas a lo largo de los 4 sprints.

Sprint	1																		
Referencia	https://github.com/assembler-school/apollofy-music-project/milestone/1																		
<h2>Sprint 1</h2>																			
📅 Due by April 16, 2021 0% complete																			
Funcionalidades: <ul style="list-style-type: none"> · Registro y autenticación de usuarios · Gestión del perfil del usuario 																			
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="padding: 5px;"><input type="checkbox"/> ⓘ 8 Open</th> <th style="padding: 5px;"><input checked="" type="checkbox"/> 0 Closed</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;"> <input type="checkbox"/> ⓘ Autentificación de Usuario live-coding-session must-have user-story #4 opened 7 days ago by JCarri14 </td> <td style="padding: 5px;"></td> </tr> <tr> <td style="padding: 5px;"> <input type="checkbox"/> ⓘ Guardado de credenciales must-have user-story #6 opened 7 days ago by JCarri14 </td> <td style="padding: 5px;"></td> </tr> <tr> <td style="padding: 5px;"> <input type="checkbox"/> ⓘ Cierre de sesión de usuario live-coding-session must-have user-story #7 opened 7 days ago by JCarri14 </td> <td style="padding: 5px;"></td> </tr> <tr> <td style="padding: 5px;"> <input type="checkbox"/> ⓘ Registro de Usuarios live-coding-session must-have user-story #8 opened 7 days ago by JCarri14 </td> <td style="padding: 5px;"></td> </tr> <tr> <td style="padding: 5px;"> <input type="checkbox"/> ⓘ Recuperación de password live-coding-session must-have user-story #9 opened 7 days ago by JCarri14 </td> <td style="padding: 5px;"></td> </tr> <tr> <td style="padding: 5px;"> <input type="checkbox"/> ⓘ Visualización de información de usuario must-have user-story #16 opened 7 days ago by JCarri14 </td> <td style="padding: 5px;"></td> </tr> <tr> <td style="padding: 5px;"> <input type="checkbox"/> ⓘ Edición de información de usuario must-have user-story #17 opened 7 days ago by JCarri14 </td> <td style="padding: 5px;"></td> </tr> <tr> <td style="padding: 5px;"> <input type="checkbox"/> ⓘ Cambio de contraseña de usuario must-have user-story #18 opened 7 days ago by JCarri14 </td> <td style="padding: 5px;"></td> </tr> </tbody> </table>		<input type="checkbox"/> ⓘ 8 Open	<input checked="" type="checkbox"/> 0 Closed	<input type="checkbox"/> ⓘ Autentificación de Usuario live-coding-session must-have user-story #4 opened 7 days ago by JCarri14		<input type="checkbox"/> ⓘ Guardado de credenciales must-have user-story #6 opened 7 days ago by JCarri14		<input type="checkbox"/> ⓘ Cierre de sesión de usuario live-coding-session must-have user-story #7 opened 7 days ago by JCarri14		<input type="checkbox"/> ⓘ Registro de Usuarios live-coding-session must-have user-story #8 opened 7 days ago by JCarri14		<input type="checkbox"/> ⓘ Recuperación de password live-coding-session must-have user-story #9 opened 7 days ago by JCarri14		<input type="checkbox"/> ⓘ Visualización de información de usuario must-have user-story #16 opened 7 days ago by JCarri14		<input type="checkbox"/> ⓘ Edición de información de usuario must-have user-story #17 opened 7 days ago by JCarri14		<input type="checkbox"/> ⓘ Cambio de contraseña de usuario must-have user-story #18 opened 7 days ago by JCarri14	
<input type="checkbox"/> ⓘ 8 Open	<input checked="" type="checkbox"/> 0 Closed																		
<input type="checkbox"/> ⓘ Autentificación de Usuario live-coding-session must-have user-story #4 opened 7 days ago by JCarri14																			
<input type="checkbox"/> ⓘ Guardado de credenciales must-have user-story #6 opened 7 days ago by JCarri14																			
<input type="checkbox"/> ⓘ Cierre de sesión de usuario live-coding-session must-have user-story #7 opened 7 days ago by JCarri14																			
<input type="checkbox"/> ⓘ Registro de Usuarios live-coding-session must-have user-story #8 opened 7 days ago by JCarri14																			
<input type="checkbox"/> ⓘ Recuperación de password live-coding-session must-have user-story #9 opened 7 days ago by JCarri14																			
<input type="checkbox"/> ⓘ Visualización de información de usuario must-have user-story #16 opened 7 days ago by JCarri14																			
<input type="checkbox"/> ⓘ Edición de información de usuario must-have user-story #17 opened 7 days ago by JCarri14																			
<input type="checkbox"/> ⓘ Cambio de contraseña de usuario must-have user-story #18 opened 7 days ago by JCarri14																			

Sprint	2							
Referencia	https://github.com/assembler-school/apollofy-music-project/milestone/2							
<h2>Sprint 2</h2>								
<p>📅 Due by April 23, 2021 0% complete</p> <p>Funcionalidades:</p> <ul style="list-style-type: none"> · Subida, reproducción y gestión de canciones <table border="1"> <thead> <tr> <th>Open</th> <th>Closed</th> </tr> </thead> <tbody> <tr> <td>12</td> <td>0</td> </tr> <tr> <td colspan="2"> <ul style="list-style-type: none"> ① Pantalla home básica must-have user-story #11 opened 7 days ago by JCarri14 ① Subida de canciones live-coding-session must-have user-story #20 opened 7 days ago by JCarri14 ① Visualización de "Mis canciones" must-have user-story #21 opened 7 days ago by JCarri14 ① Dar "Me gusta" a una canción must-have user-story #22 opened 7 days ago by JCarri14 ① Eliminar canción dentro del diálogo de información de canción must-have user-story #23 opened 7 days ago by JCarri14 ① Reproducción de canciones must-have user-story #25 opened 2 hours ago by JCarri14 ① Visualización del reproductor must-have user-story #26 opened 2 hours ago by JCarri14 ① Pausar canción must-have user-story #27 opened 2 hours ago by JCarri14 ① Canción atrás should-have user-story #28 opened 2 hours ago by JCarri14 ① Canción hacia adelante should-have user-story #29 opened 2 hours ago by JCarri14 ① Me gusta a una canción dentro del diálogo de información de canción must-have user-story #52 opened 1 hour ago by JCarri14 ① Opción de editar canción dentro de diálogo de información de canción must-have user-story #53 opened 1 hour ago by JCarri14 </td><td></td></tr> </tbody> </table>		Open	Closed	12	0	<ul style="list-style-type: none"> ① Pantalla home básica must-have user-story #11 opened 7 days ago by JCarri14 ① Subida de canciones live-coding-session must-have user-story #20 opened 7 days ago by JCarri14 ① Visualización de "Mis canciones" must-have user-story #21 opened 7 days ago by JCarri14 ① Dar "Me gusta" a una canción must-have user-story #22 opened 7 days ago by JCarri14 ① Eliminar canción dentro del diálogo de información de canción must-have user-story #23 opened 7 days ago by JCarri14 ① Reproducción de canciones must-have user-story #25 opened 2 hours ago by JCarri14 ① Visualización del reproductor must-have user-story #26 opened 2 hours ago by JCarri14 ① Pausar canción must-have user-story #27 opened 2 hours ago by JCarri14 ① Canción atrás should-have user-story #28 opened 2 hours ago by JCarri14 ① Canción hacia adelante should-have user-story #29 opened 2 hours ago by JCarri14 ① Me gusta a una canción dentro del diálogo de información de canción must-have user-story #52 opened 1 hour ago by JCarri14 ① Opción de editar canción dentro de diálogo de información de canción must-have user-story #53 opened 1 hour ago by JCarri14 		
Open	Closed							
12	0							
<ul style="list-style-type: none"> ① Pantalla home básica must-have user-story #11 opened 7 days ago by JCarri14 ① Subida de canciones live-coding-session must-have user-story #20 opened 7 days ago by JCarri14 ① Visualización de "Mis canciones" must-have user-story #21 opened 7 days ago by JCarri14 ① Dar "Me gusta" a una canción must-have user-story #22 opened 7 days ago by JCarri14 ① Eliminar canción dentro del diálogo de información de canción must-have user-story #23 opened 7 days ago by JCarri14 ① Reproducción de canciones must-have user-story #25 opened 2 hours ago by JCarri14 ① Visualización del reproductor must-have user-story #26 opened 2 hours ago by JCarri14 ① Pausar canción must-have user-story #27 opened 2 hours ago by JCarri14 ① Canción atrás should-have user-story #28 opened 2 hours ago by JCarri14 ① Canción hacia adelante should-have user-story #29 opened 2 hours ago by JCarri14 ① Me gusta a una canción dentro del diálogo de información de canción must-have user-story #52 opened 1 hour ago by JCarri14 ① Opción de editar canción dentro de diálogo de información de canción must-have user-story #53 opened 1 hour ago by JCarri14 								

Sprint	3
Referencia	https://github.com/assembler-school/apollofy-music-project/milestone/3
<h3>Sprint 3</h3>	
[] Due by April 30, 2021 0% complete Funcionalidades: · Creación y administración de playlists	
<input type="checkbox"/> ⓘ 10 Open ✓ 0 Closed <ul style="list-style-type: none"> <input type="checkbox"/> ⓘ Listado de playlists must-have user-story #33 opened 2 hours ago by JCarri14 <input type="checkbox"/> ⓘ Orden manual en listado de playlist must-have user-story #34 opened 2 hours ago by JCarri14 <input type="checkbox"/> ⓘ Creación de playlists live-coding-session must-have user-story #35 opened 2 hours ago by JCarri14 <input type="checkbox"/> ⓘ Vista de Playlists must-have user-story #36 opened 2 hours ago by JCarri14 <input type="checkbox"/> ⓘ Seguir Playlist en Visualización de Playlists must-have user-story #37 opened 2 hours ago by JCarri14 <input type="checkbox"/> ⓘ Cambio de privacidad en vista de Playlist should-have user-story #38 opened 2 hours ago by JCarri14 <input type="checkbox"/> ⓘ Añadir canciones dentro de Playlist desde vista Playlist live-coding-session must-have user-story #39 opened 2 hours ago by JCarri14 <input type="checkbox"/> ⓘ Menú "Opciones" dentro de Playlist must-have user-story #40 opened 2 hours ago by JCarri14 ⋮ <input type="checkbox"/> ⓘ Añadir canción a playlist dentro del diálogo de información de canción must-have user-story #42 opened 2 hours ago by JCarri14 <input type="checkbox"/> ⓘ Botón eliminar canción de la playlist dentro del diálogo de información de canción must-have user-story #43 opened 2 hours ago by JCarri14 	

Sprint	4				
Referencia	https://github.com/assembler-school/apollofy-music-project/milestone/4				
<h2>Sprint 4</h2>					
<p>📅 Due by May 07, 2021 0% complete</p> <p>Funcionalidades:</p> <ul style="list-style-type: none"> - Visualización y seguimiento de usuarios - Estadísticas de las entidades del sistema <table border="1"> <thead> <tr> <th>Open</th> <th>Closed</th> </tr> </thead> <tbody> <tr> <td>10</td> <td>0</td> </tr> </tbody> </table> <ul style="list-style-type: none"> Sección "Playlists populares" en pantalla home should-have user-story #12 opened 7 days ago by JCarri14 Sección "Artistas populares" en pantalla home should-have user-story #13 opened 7 days ago by JCarri14 Sección "Playlists que sigo" en pantalla home should-have user-story #14 opened 7 days ago by JCarri14 Búsqueda por texto must-have user-story #31 opened 2 hours ago by JCarri14 Vista de usuario must-have user-story #45 opened 2 hours ago by JCarri14 Seguir a otros usuarios should-have user-story #46 opened 2 hours ago by JCarri14 Visualización de seguidores should-have user-story #47 opened 2 hours ago by JCarri14 Visualización de siguiendo should-have user-story #48 opened 2 hours ago by JCarri14 Visualización de top géneros reproducidos en la app must-have user-story #50 opened 2 hours ago by JCarri14 Visualización de las canciones más reproducidas en la app live-coding-session must-have user-story #51 opened 2 hours ago by JCarri14 		Open	Closed	10	0
Open	Closed				
10	0				

Kanban Board

Implementaréis vuestra plataforma mediante GitHub, aprovechando la potencia de Git como sistema de control de versiones, y también para la gestión del proyecto (GitHub Projects) y CI/CD (Github Actions). Realizaréis Code Reviews y las mejores prácticas Agile para asegurar un buen desarrollo en equipo.

Apollofy
Updated 1 hour ago

Filter cards

To do	In progress	Review in progress	Reviewer approved	Done
<p>Autenticación de Usuario</p> <p>#4 opened by JCarr14</p> <p>live-coding-session must-have user-story</p> <p>Sprint 1</p>	<p>Registro y Autenticación</p> <p>0 of 5</p> <p>#5 opened by JCarr14</p> <p>epic</p>	<p>Guardado de credenciales</p> <p>#6 opened by JCarr14</p> <p>must-have user-story</p> <p>Sprint 1</p>	<p>Cierre de sesión de usuario</p> <p>#7 opened by JCarr14</p> <p>live-coding-session must-have user-story</p> <p>Sprint 1</p>	<p>Registro de Usuarios</p> <p>#8 opened by JCarr14</p> <p>live-coding-session must-have user-story</p> <p>Sprint 1</p>
<p>Automated as To do</p> <p>Manage</p>	<p>Automated as In progress</p> <p>Manage</p>	<p>Automated as In progress</p> <p>Manage</p>	<p>Automated as In progress</p> <p>Manage</p>	<p>Automated as Done</p> <p>Manage</p>

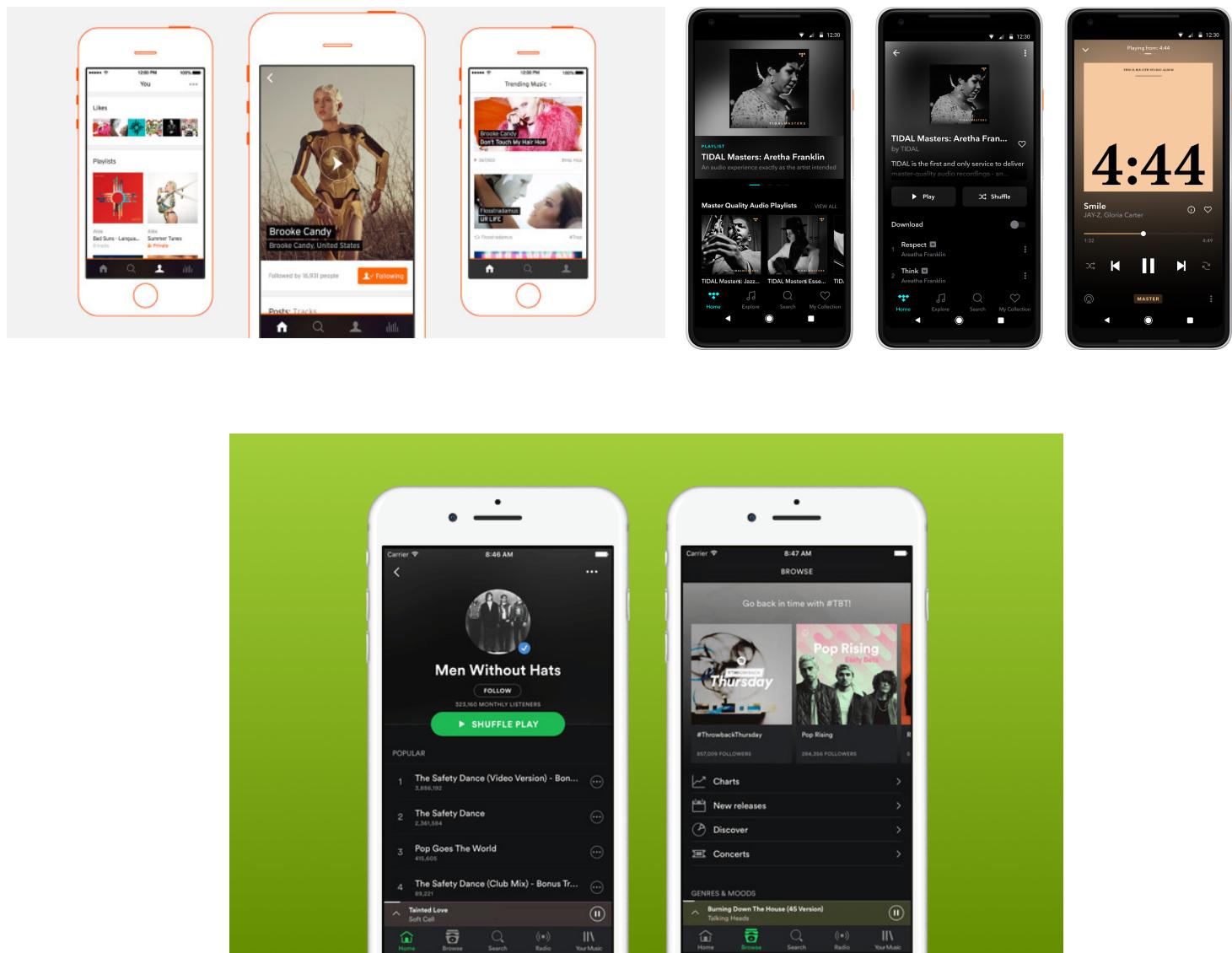
Tareas a realizar durante la presentación

Las habilidades comunicativas, de “vender” un proyecto y de presentarlo son muy importantes de cara a las empresas. La fase final del proyecto consta de una presentación del proyecto al equipo académico. Se dispondrán de **45 minutos** para presentar el proyecto, sus objetivos, las funcionalidades que ha decidido asumir, la posibilidad de alguna funcionalidad adicional, además de una demo del propio proyecto y de mostrar alguna estructura de código que pudieran parecer importantes.

Descripción del Producto

State of the Art

Existen muchas aplicaciones en el mercado relacionadas con la reproducción de música. Algunas de ellas son Soundcloud, Tidal o Spotify. **Apollofy** se encuentra dentro del sector de este tipo de aplicaciones, con mayor similitud por Soundcloud en vez que por Spotify, por el hecho de que este permite la publicación de canciones por parte del usuario y no existe una diferenciación clara entre usuario y artista. A continuación se adjuntan algunas capturas de pantalla de las actuales aplicaciones del mercado relacionadas con la reproducción de música.



Objetivo del Producto

¿Qué es Apollofy?

El objetivo del producto *Apollofy* es realizar una **fullstack webapp** empleando tecnología MERN y PHP para la publicación y reproducción de canciones entre una comunidad de usuarios. Los usuarios podrán acceder a la aplicación para reproducir canciones subidas por ellos mismos o canciones subidas por otros usuarios.

¿Por qué el nombre Apollofy?

Había dos opciones: o llamarle Apollofy o **Assemblerfy**. Y pareció mucho más artístico llamarle Apollofy. Apolo es el dios griego de las artes, de la verdad, de la belleza, de la armonía y de la razón. ¡Cuántas veces necesitaremos el favor del dios Apolo para conseguir orden y equilibrio en el desarrollo del proyecto!

¿Cuál es el futuro de Apollofy?

El proyecto inicia en Assembler, pero el futuro de este depende completamente de vosotros. El valor principal del proyecto es que se trata de un proyecto *open source* publicado en Github y que servirá de portfolio de los participantes para cualquier entrevista de trabajo. Será vuestro proyecto estelar a mostrar en todas las entrevistas como una prueba evidente de vuestras habilidades como desarrolladores.

Nos encantaría que este proyecto no terminara al finalizar vuestra experiencia en Assembler. Por ello, se ha preparado un listado muy amplio de funcionalidades opcionales que pueden implementarse posteriormente para poder darle vida al proyecto más allá del máster.

¿Qué funcionalidades incluirá Apollofy?

En términos generales, se trata de una aplicación web multi-idioma que permite:

- Registro y autenticación de usuarios
- Acceso a una pantalla home básica
- Apartado de gestión de cuenta

Las funcionalidades principales serán:

- Publicación, visualización, reproducción y gestión de canciones
- Dar like a una canción
- Buscador de canciones
- Publicación, visualización y gestión de playlists
- Poder seguir una playlist
- Visualizador y gestor de usuarios/artistas, con opciones de seguimiento
- Visualizador de estadísticas en las que poder ver por número de reproducciones los géneros, artistas o canciones más escuchados (backend de Laravel con MySQL)

Como funcionalidades opcionales:

- Búsqueda completa en todas las entidades del sistema (playlists, usuarios y canciones)
- Pantalla de home avanzada
- Publicación y gestión de álbumes
- Sistema avanzado de cola de reproducción
- Posibilidad de castear una canción a chromecast
- Posibilidad de subir y reproducir vídeos además de canciones
- Opciones de gestión de contenido offline
- Sistemas de notificaciones

Las funcionalidades se han distribuido aproximadamente en los siguientes Sprints:

- Sprint 1: Registro y autenticación de usuarios. Gestión del perfil del usuario
- Sprint 2: Subida, reproducción y gestión de canciones
- Sprint 3: Creación y administración de playlists
- Sprint 4: Visualización y seguimiento de usuarios. Estadísticas de las entidades del sistema

En el siguiente apartado del documento se encontrará detallado un listado completo de Epics e Historias de Usuario del producto Apollofy con la designación original de su prioridad (must, should y could), número de sprint (1-4) y story points (1-21). **Let's Do This!! May your Soul take flight on the breath of the Great Spirit!**



May your Soul take flight on the breath of the Great Spirit.

Epics e Historias de Usuario MUST HAVE

Epics

#	Epic title	Epic description	Priority	Sprint
1	Registro y Autenticación	Funcionalidades básicas de autenticación y recuperación de password	MUST HAVE	Sprint 1
2	Pantalla Home principal básica	Pantalla de bienvenida, con funciones básicas de acceso a los datos de la aplicación	MUST HAVE	Sprint 2
3	Gestión de la cuenta	Gestión de la cuenta de usuario, fotografía, password, datos de cuenta	MUST HAVE	Sprint 1
4	Publicación, visualización y gestión de canciones	Apartado de “My Songs” en el menú principal donde aparecen canciones publicadas y con “me gusta”, publicación de canciones y visualización de canciones	MUST HAVE	Sprint 2
5	Reproductor de canciones	Reproducción de canciones	MUST HAVE	Sprint 2
6	Búsqueda de canciones	Buscador de canciones	MUST HAVE	Sprint 4
7	Publicación, visualización y gestión de playlists	Apartado de “My Playlists” en mi menú principal con propias playlists y playlists con “me gusta”, creación, gestión y visualización de playlists de otros usuarios	MUST HAVE	Sprint 3
8	Visualización de estadísticas	Visualización de estadísticas de uso de mis canciones, etc.	MUST HAVE	Sprint 4

Historias de Usuario

Podéis consultar las user stories organizadas según los diversos sprints, directamente en el [repositorio de GitHub](#).

#	Epic	User story title	User story description	SP	Priority	Sprint
-	(1) Registro y Autenticación	Autentificación de Usuario	El usuario podrá iniciar sesión introduciendo sus credenciales de acceso al sistema: Username y password	8	Seed Project	Sprint 1
-	(1) Registro y Autenticación	Cierre de sesión de usuario	El usuario podrá cerrar la sesión clicando en el botón de cierre de sesión	5	Seed Project	Sprint 1
-	(1) Registro y Autenticación	Registro de Usuarios	El usuario podrá registrarse en el sistema suministrando los datos: Nombre, Apellido, E-mail, Username y Password	13	Seed Project	Sprint 1
-	(1) Registro y Autenticación	Recuperación de password	El usuario podrá cambiar el password gracias a una URL de recuperación de password ofrecida por Firebase enviada a su correo electrónico	8	Seed Project	Sprint 1
1	(3) Gestión de la cuenta	Visualización de información de usuario	El usuario podrá acceder a visualizar la información de su propio usuario, es decir, username, email, foto de perfil	5	MUST HAVE	Sprint 1
2	(3) Gestión de la cuenta	Edición de información de usuario	El usuario podrá modificar la información de username, email y foto de perfil	5	MUST HAVE	Sprint 1
3	(3) Gestión de la cuenta	Cambio de contraseña de usuario	El usuario podrá acceder a modificar su contraseña si este introduce la suya previa y reintroduce correctamente la nueva contraseña con una validación	3	MUST HAVE	Sprint 1
4	(2) Pantalla Home principal	Pantalla home básica	Pantalla básica de Home principal	5	MUST HAVE	Sprint 2
-	(4) Publicación, visualización y gestión de canciones	Subida de canciones	El usuario podrá subir canciones en la plataforma, siendo estas disponibles para todos los usuarios, y gestionada por el propietario, visualizada en el apartado "Mis canciones"	21	Seed Project	Sprint 2

5	(4) Publicación, visualización y gestión de canciones	Visualización de "Mis canciones"	El usuario podrá acceder a todas las canciones subidas dentro del apartado "Mis canciones", además, este apartado mostrará las canciones que el usuario le ha dado a "Me gusta"	8	MUST HAVE	Sprint 2
6	(4) Publicación, visualización y gestión de canciones	Dar "Me gusta" a una canción	El usuario podrá dar "Me gusta" a las canciones, guardándolas automáticamente en el apartado "Mis canciones"	8	MUST HAVE	Sprint 2
7	(4) Publicación, visualización y gestión de canciones	Me gusta a una canción dentro del diálogo de información de canción	El usuario podrá dar "me gusta" a una canción mediante el diálogo de información de canción	2	MUST HAVE	Sprint 2
8	(4) Publicación, visualización y gestión de canciones	Opción de editar canción dentro de diálogo de información de canción	El usuario podrá editar la información de una canción (nombre, portada, género) dentro del diálogo de información de canción	8	MUST HAVE	Sprint 2
9	(4) Publicación, visualización y gestión de canciones	Eliminar canción dentro del diálogo de información de canción	El usuario podrá eliminar una canción accediendo al diálogo de información de canción	3	MUST HAVE	Sprint 2
10	(5) Reproductor de canciones	Reproducción de canciones	El usuario podrá reproducir canciones seleccionando la canción deseada, y mostrando en la barra de reproducción la información correspondiente	5	MUST HAVE	Sprint 2
11	(5) Reproductor de canciones	Visualización del reproductor	El usuario dispondrá de una vista donde visualizará el detalle de reproducción, la foto de la canción, y las distintas opciones de reproducción.	8	MUST HAVE	Sprint 2
12	(5) Reproductor de canciones	Pausar canción	El usuario dispondrá del control de pausar la canción	1	MUST HAVE	Sprint 2
13	(7) Publicación, visualización y gestión de playlists	Listado de playlists	El usuario podrá acceder a "Mis Playlists", un listado de playlists en la que aparecerán todas las que son de su propiedad o que el usuario está siguiendo	2	MUST HAVE	Sprint 3
14	(7) Publicación, visualización y	Orden manual en listado de playlist	El usuario podrá definir su propio orden manual en el listado de "Mis Playlists"	8	MUST HAVE	Sprint 3

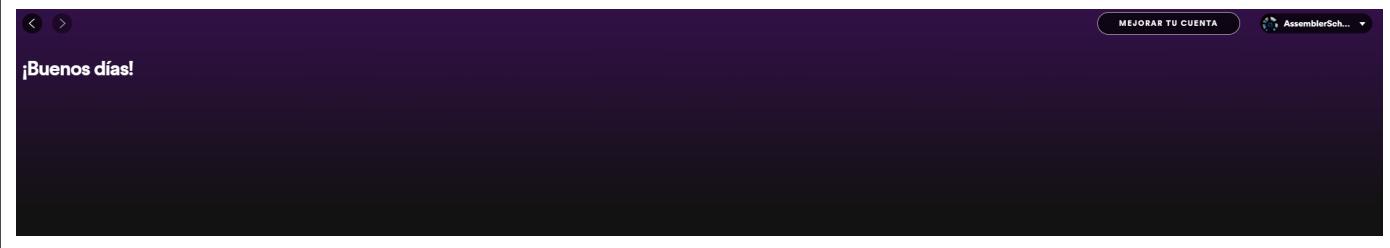
	gestión de playlists					
15	(7) Publicación, visualización y gestión de playlists	Creación de playlists	El usuario podrá crear sus propias playlists, solicitando nombre, fotografía, privacidad y descripción de esta	5	MUST HAVE	Sprint 3
16	(7) Publicación, visualización y gestión de playlists	Vista de Playlists	El usuario podrá acceder a la vista de una playlist, en la que se mostrará el listado de canciones que incluye esta playlist	5	MUST HAVE	Sprint 3
16	(7) Publicación, visualización y gestión de playlists	Seguir Playlist en Visualización de Playlists	En la vista de playlist, el usuario (si no es propietario) podrá seguir una Playlist, guardándose dentro de su listado de playlists	8	MUST HAVE	Sprint 3
-	(7) Publicación, visualización y gestión de playlists	Añadir canciones dentro de Playlist desde vista Playlist	En la vista de playlist, el usuario podrá añadir canciones (si es propietario) mediante un botón, mostrando una pantalla con su listado de canciones (propietarias o me gusta)	8	MUST HAVE	Sprint 3
17	(7) Publicación, visualización y gestión de playlists	Menú “Opciones” dentro de Playlist	En la vista de playlist, accediendo al botón “Opciones”, podrá editar los datos de la playlist (si es propietario), ver más información del creador, compartirla y eliminarla (si es propietario)	5	MUST HAVE	Sprint 3
18	(4) Publicación, visualización y gestión de canciones	Añadir a la playlist dentro del diálogo de información de canción	El usuario podrá añadir una canción a una playlist suya desde el diálogo de información de canción	5	MUST HAVE	Sprint 3
19	(4) Publicación, visualización y gestión de canciones	Botón eliminar de la playlist dentro del diálogo de información de canción	El usuario podrá eliminar una canción de una playlist suya desde el diálogo de información de canción	2	MUST HAVE	Sprint 3
20	(6) Búsqueda de canciones, usuarios, playlists y álbumes	Búsqueda por texto	El usuario podrá buscar canciones, playlists y artistas introduciendo en el campo de búsqueda lo buscado, uniendo la búsqueda de texto en estas tres secciones visuales	13	MUST HAVE	Sprint 4

21	(8) Visualización y gestión de usuarios/artistas	Vista de usuario	El usuario podrá acceder a visualizar otros usuarios, donde se mostrará al usuario con nombre apellido y username, número de seguidores, un apartado con el listado de playlists suyos, sus canciones subidas, y sus álbumes	8	MUST HAVE	Sprint 4
22	(13) Visualización de estadísticas	Visualización de top géneros reproducidos en la app	El usuario podrá visualizar los top géneros reproducidos en la aplicación en un pie chart	13	MUST HAVE	Sprint 4
-	(13) Visualización de estadísticas	Visualización de las canciones más reproducidas en la app	El usuario podrá visualizar un gráfico de barras de las canciones más reproducidas en la aplicación (Chart.js or d3.js)	13	MUST HAVE	Sprint 4

Mockups-Stories Orientation

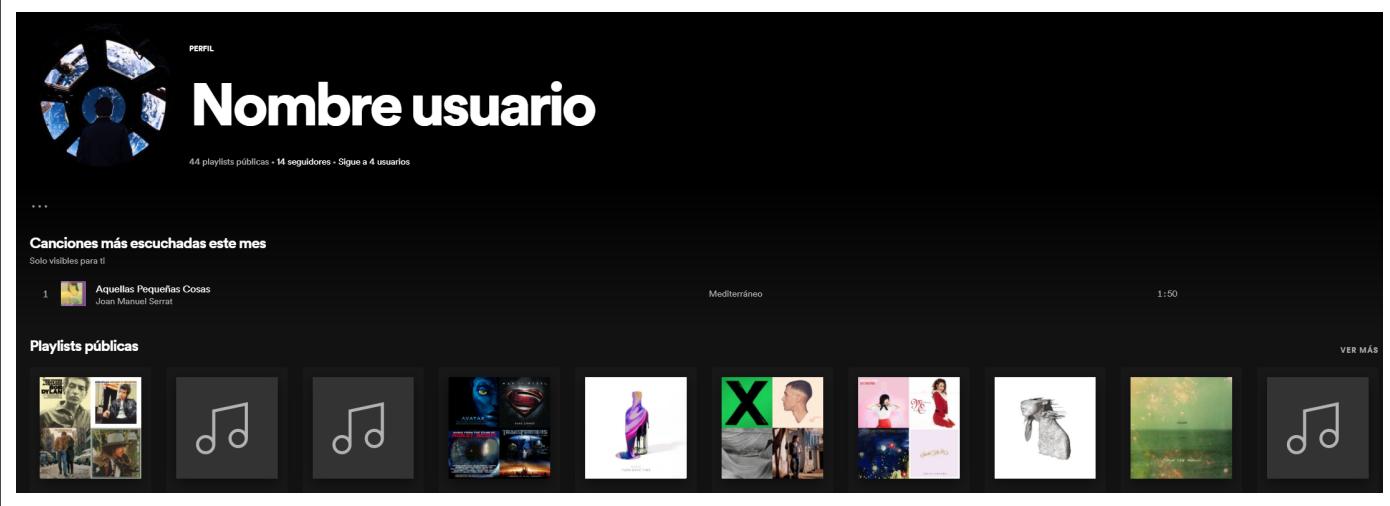
Pantalla Home principal

Epic	Pantalla Home principal	MUST HAVE
User Story Title	Pantalla home básica	
API Endpoint	/	



Gestión de la cuenta

Epic	Gestión de la cuenta	MUST HAVE
User Story Title	Visualización de información de usuario	
API Endpoint	/api/account	



Epic	Gestión de la cuenta	
User Story Title	Edición de información de usuario	MUST HAVE
API Endpoint	/api/account	

Editar perfil

Correo electrónico

Sexo

Masculino

Fecha de nacimiento

 / /

País o región

España

Compartir mis datos de registro con los proveedores de contenidos de Spotify para fines de marketing. Ten en cuenta que tus datos pueden ser transferidos a un país fuera del EEE, tal y como se describe en nuestra política de privacidad.

CANCELAR

GUARDAR PERFIL

Epic	Gestión de la cuenta	MUST HAVE
User Story Title	Cambio de contraseña de usuario	
API Endpoint	/api/authenticate	

Publicación, visualización y gestión de canciones

Epic	Publicación, visualización y gestión de canciones	MUST HAVE
User Story Title	Visualización de "Mis canciones"	
API Endpoint	/api/me/tracks	



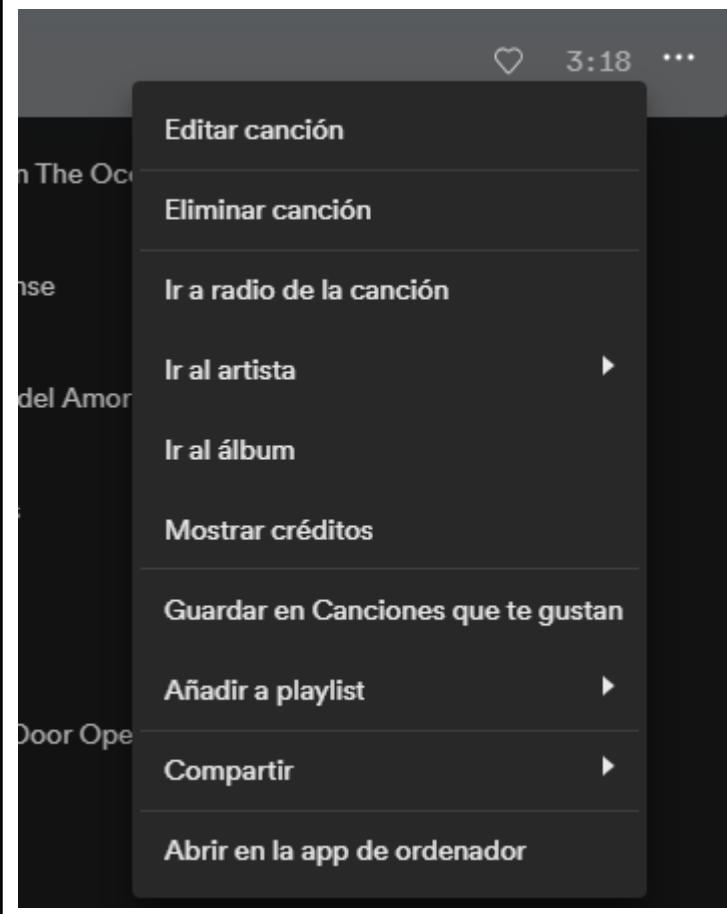
Epic	Publicación, visualización y gestión de canciones	MUST HAVE
User Story Title	Dar “Me gusta” a una canción	
API Endpoint	/api/tracks/{id}/like	



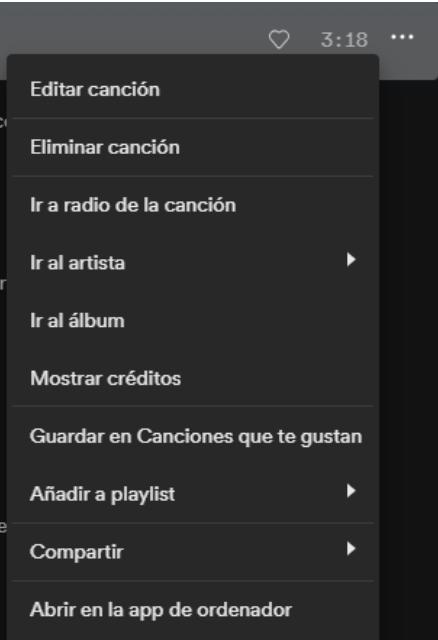
Epic	Publicación, visualización y gestión de canciones	MUST HAVE
User Story Title	Me gusta a una canción dentro del diálogo de información de canción	
API Endpoint	/api/tracks/{id}/like	



Epic	Publicación, visualización y gestión de canciones	
User Story Title	Opción de editar canción dentro de diálogo de información de canción	MUST HAVE
API Endpoint	/api/tracks/{id}	

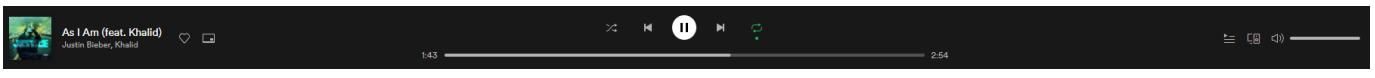


Epic	Publicación, visualización y gestión de canciones	MUST HAVE
User Story Title	Eliminar canción dentro del diálogo de información de canción	
API Endpoint	/api/tracks/{id}	

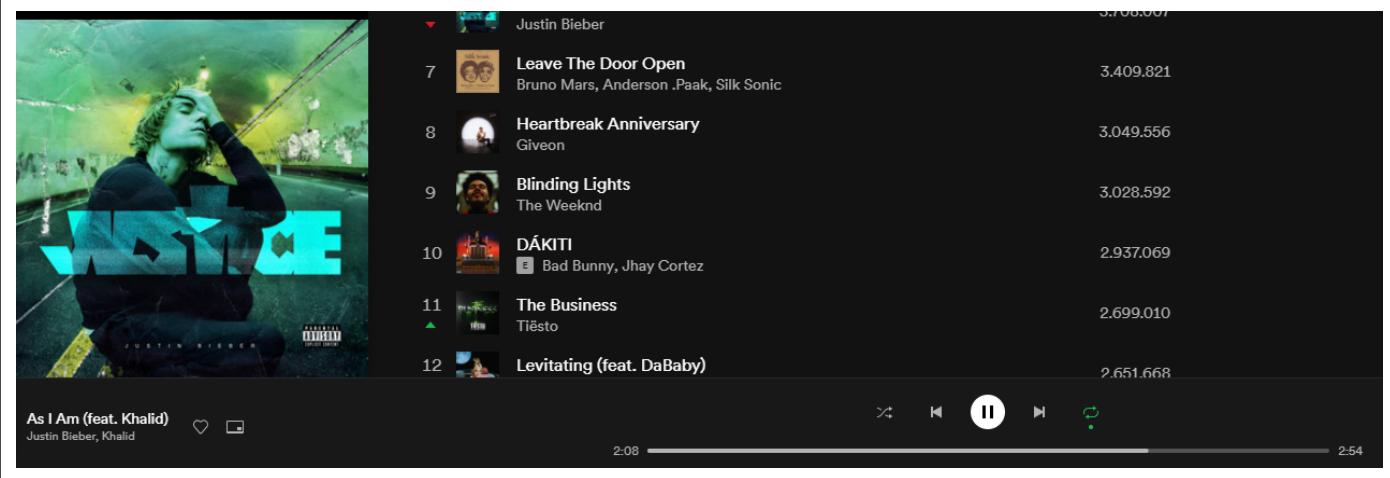


Reproductor de Canciones

Epic	Reproductor de canciones	MUST HAVE
User Story Title	Reproducción de canciones	
API Endpoint	/api/tracks/{id}/play	



Epic	Reproductor de canciones	
User Story Title	Visualización del reproductor	MUST HAVE
API Endpoint	n/a	

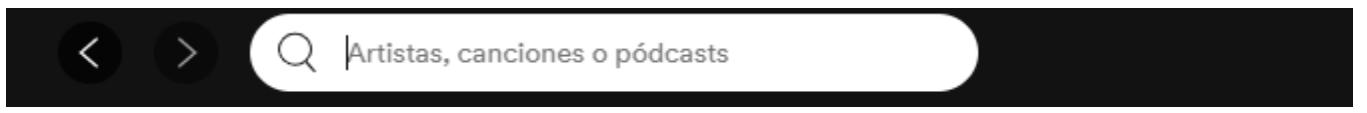


Epic	Reproductor de canciones	
User Story Title	Pausar canción	MUST HAVE
API Endpoint	n/a	

A screenshot of a music player interface showing a song playing. The progress bar is at 2:34 of 2:54. The playback controls include a play/pause button, a shuffle button, and a repeat button.

Búsqueda de canciones, usuarios, playlists y álbumes

Epic	Búsqueda de canciones, usuarios, playlists y álbumes	MUST HAVE
User Story Title	Búsqueda por texto	
API Endpoint	/api/search	

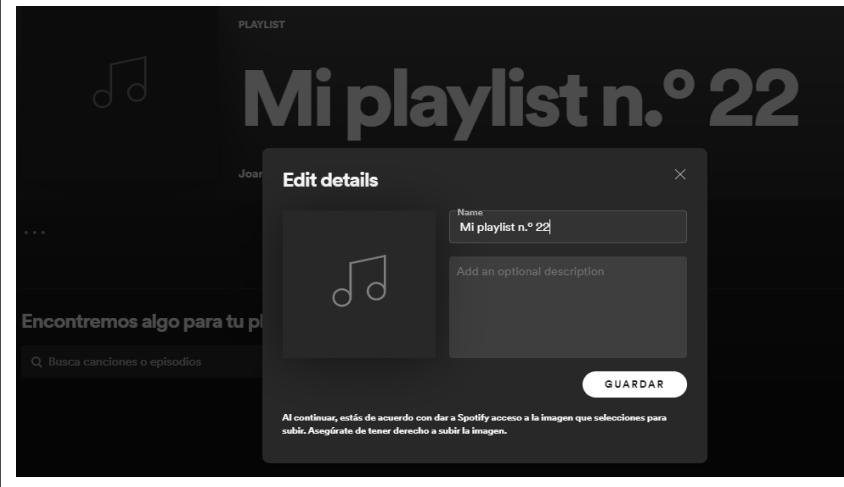


Publicación, visualización y gestión de playlists

Epic	Publicación, visualización y gestión de playlists	MUST HAVE
User Story Title	Listado de playlists	
API Endpoint	/api/authenticate	

Epic	Publicación, visualización y gestión de playlists	MUST HAVE
User Story Title	Orden manual en listado de playlist	
API Endpoint	n/a	

Epic	Publicación, visualización y gestión de playlists	
User Story Title	Creación de playlists	MUST HAVE
API Endpoint	/api/playlists	

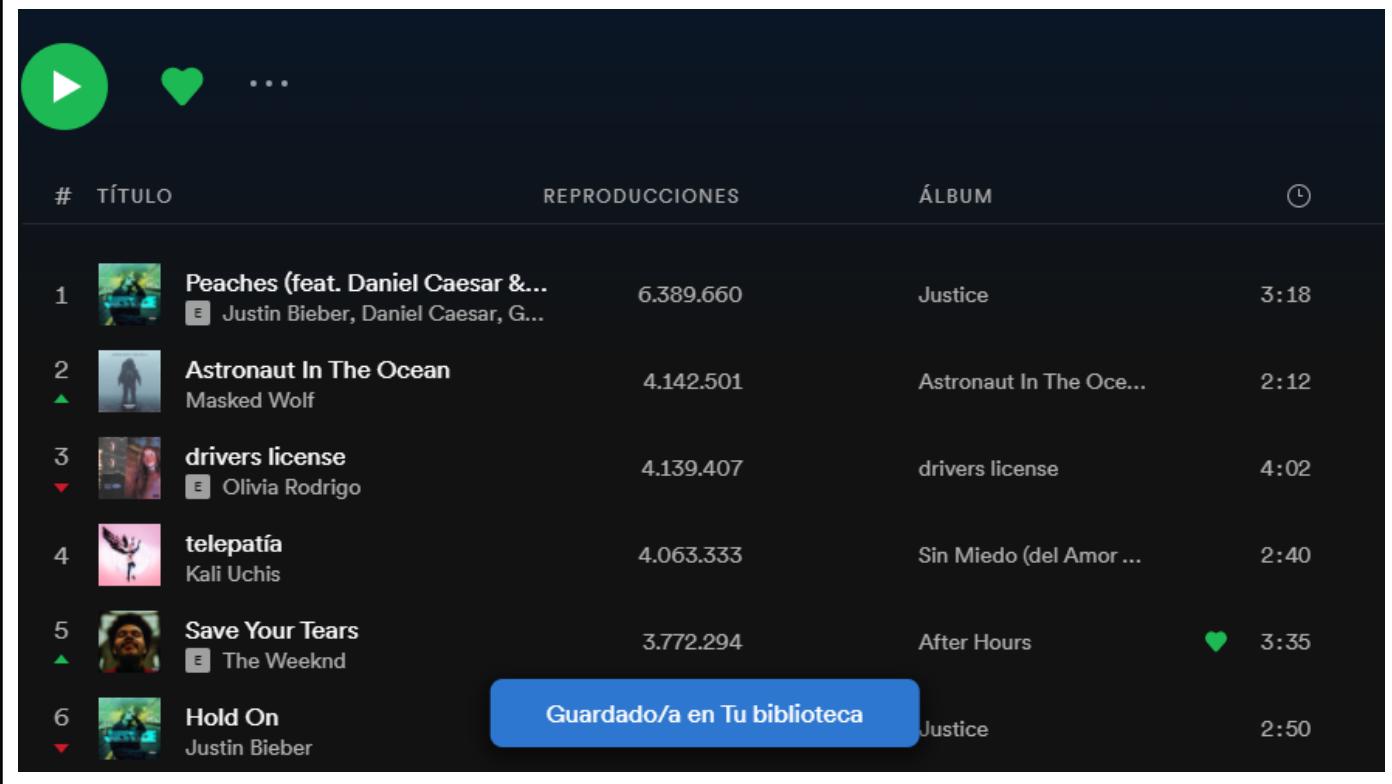


Epic	Publicación, visualización y gestión de playlists	
User Story Title	Vista de Playlists	MUST HAVE
API Endpoint	/api/playlists/{id}	

The screenshot shows the Spotify Global Top 50 chart. At the top left is a sidebar with 'Top 50' and 'GLOBAL'. The main title 'Top 50: Global' is displayed prominently. Below it, a subtitle reads 'Tu actualización diaria de las canciones más escuchadas ahora mismo en Global.' and 'spotifycharts - 15.983.225 "me gusta" - 50 canciones, 2 hr 35 min • 2 entradas nuevas'. On the left, there are three circular icons: a play button, a heart, and three dots. The main area lists the top 50 songs with columns for rank (#), title, reproducciones (plays), álbum (album), and duration. The first few songs are:

#	TÍTULO	REPRODUCCIONES	ÁLBUM	DURACIÓN
1	Peaches (feat. Daniel Caesar & Giveon) Justin Bieber, Daniel Caesar, Giveon	6.389.660	Justice	3:18
2	Astronaut In The Ocean Masked Wolf	4.142.501	Astronaut In The Ocean	2:12
3	drivers license Olivia Rodrigo	4.139.407	drivers license	4:02
4	telepatía Kali Uchis	4.063.333	Sin Miedo (del Amor y Otros Demonios) ~	2:40
5	Save Your Tears The Weeknd	3.772.294	After Hours	3:35
6	Hold On Justin Bieber	3.708.007	Justice	2:50

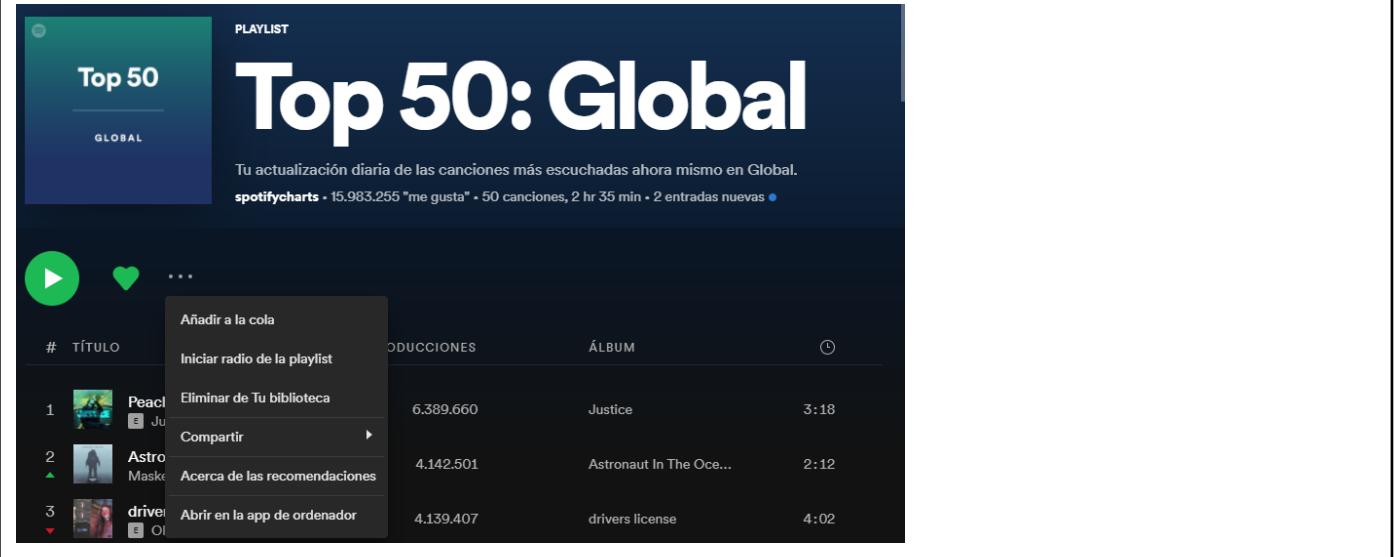
Epic	Publicación, visualización y gestión de playlists	MUST HAVE
User Story Title	Seguir Playlist en Visualización de Playlists	
API Endpoint	/api/playlist/{id}/follow	



Epic	Publicación, visualización y gestión de playlists	MUST HAVE
User Story Title	Añadir canciones dentro de Playlist desde vista Playlist	
API Endpoint	/api/playlists	



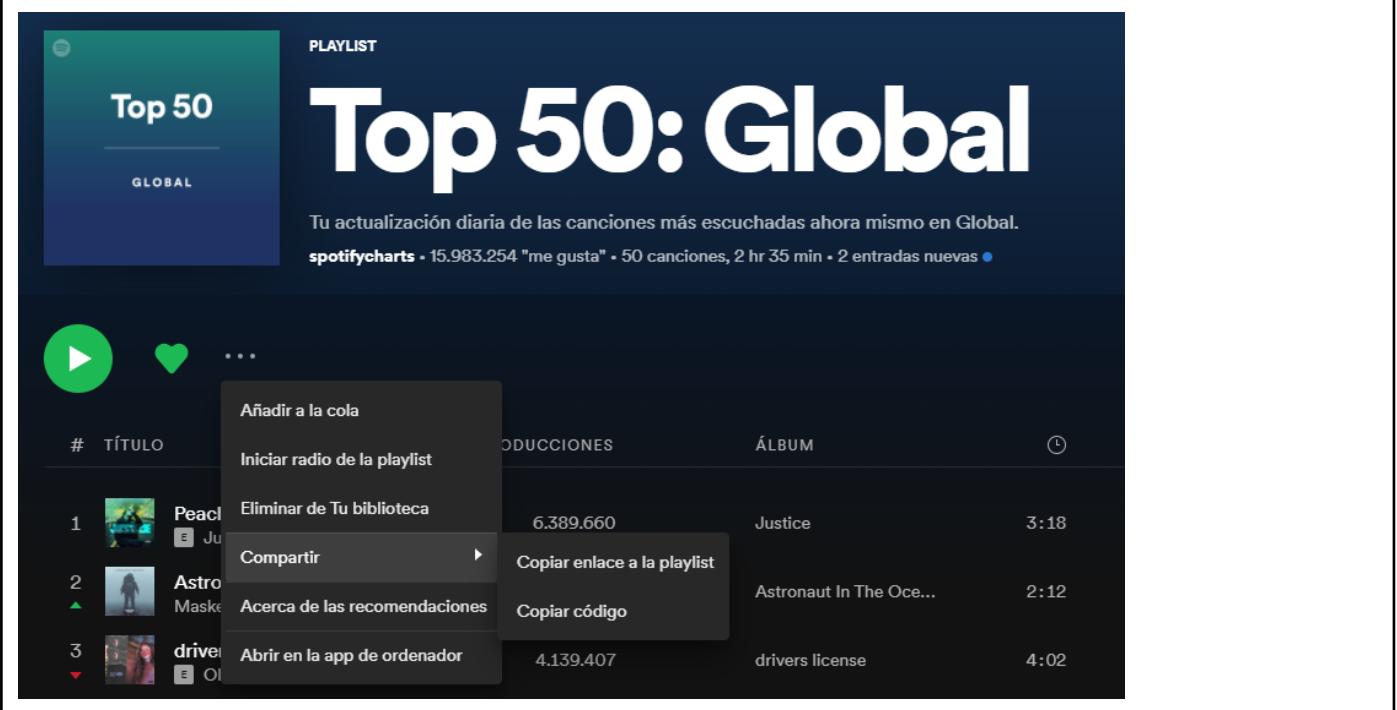
Epic	Publicación, visualización y gestión de playlists	MUST HAVE
User Story Title	Menú “Opciones” dentro de Playlist	
API Endpoint	n/a	



Visualización y gestión de usuarios/artistas

Epic	Visualización y gestión de usuarios/artistas	MUST HAVE
User Story Title	Vista de usuario	
API Endpoint	/api/users/{id}	

Epic	Visualización y gestión de usuarios/artistas	MUST HAVE
User Story Title	Compartición de perfil	
API Endpoint	n/a	

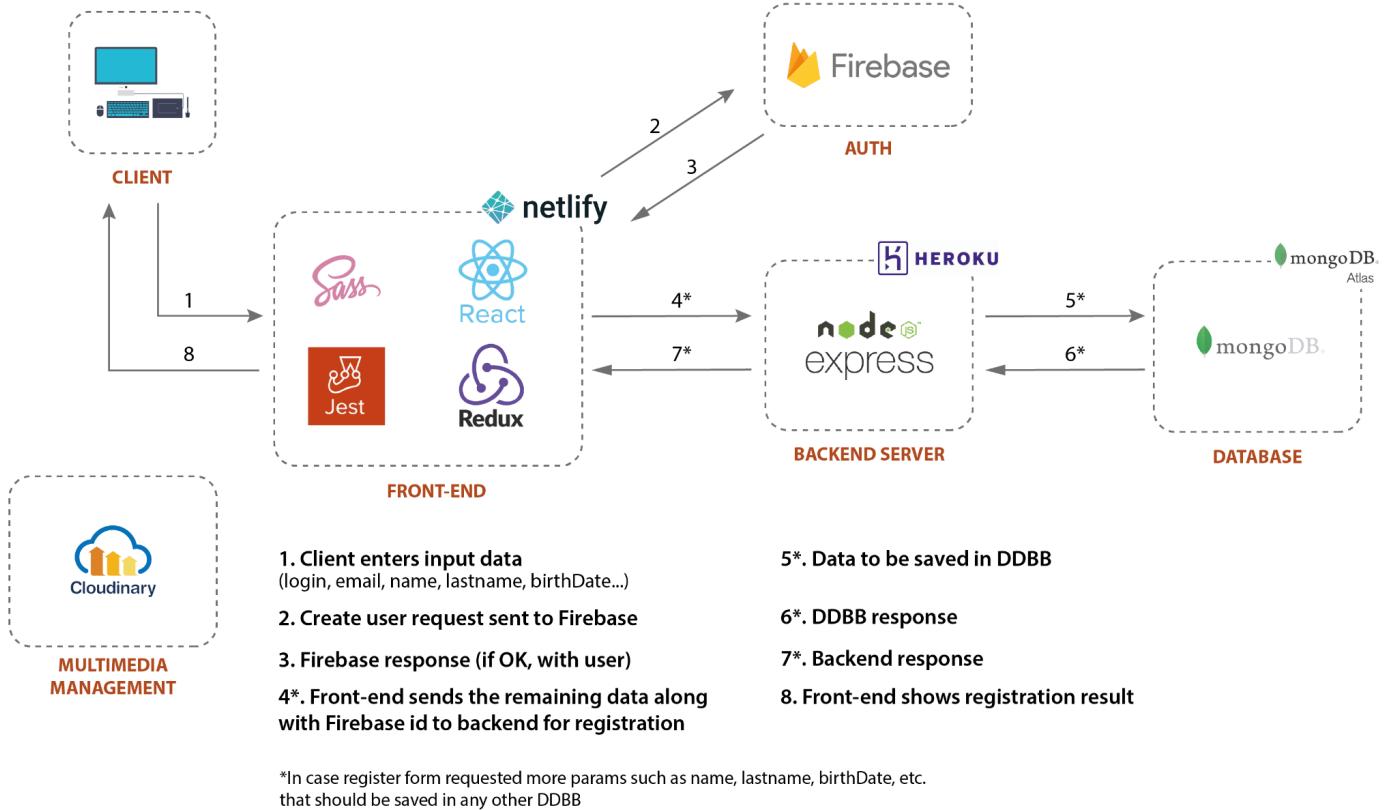


Visualización de estadísticas

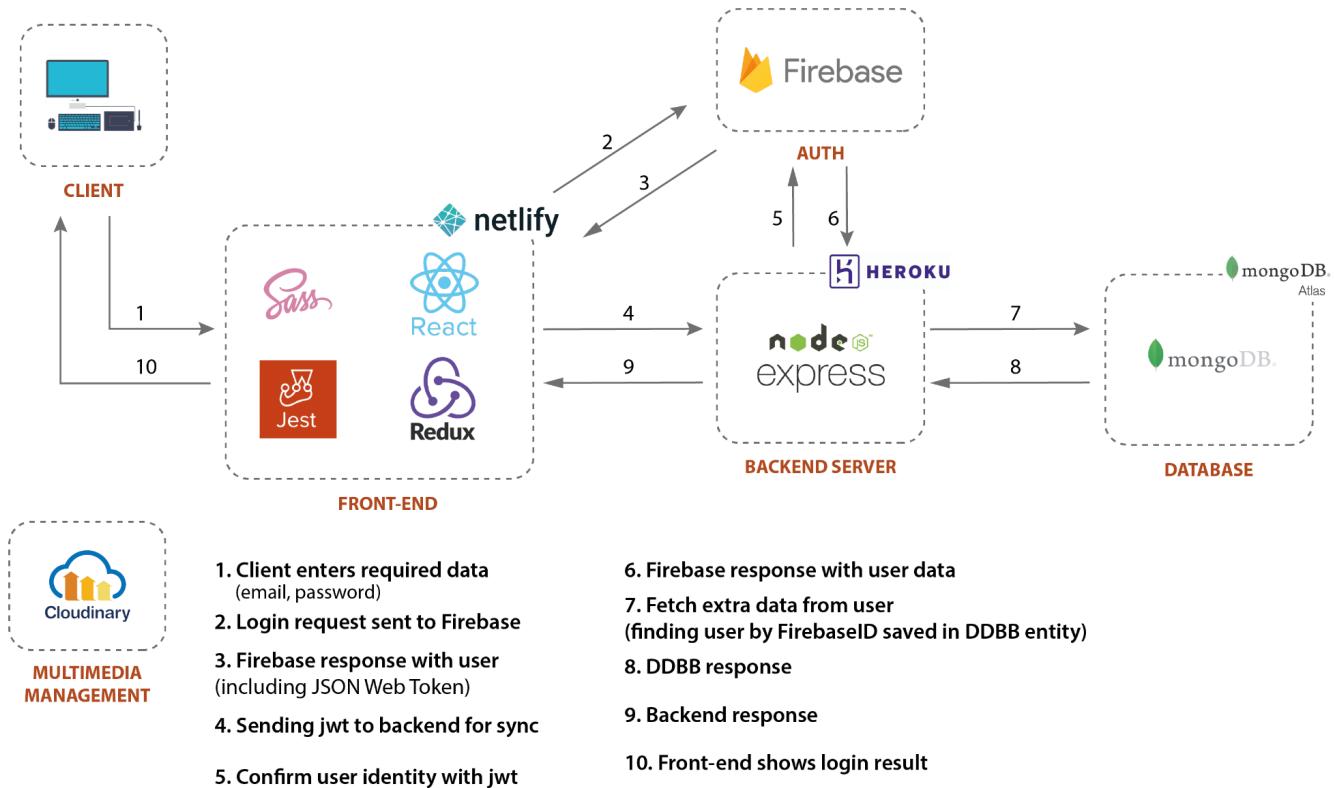
Epic	Visualización de estadísticas	MUST HAVE
User Story Title	Visualización de top géneros reproducidos en la app	
API Endpoint	/api/stats/genres/most-listened	
https://blog.soundcloud.com/2015/03/11/how-to-use-your-soundcloud-stats/		

Dataflow Diagrams

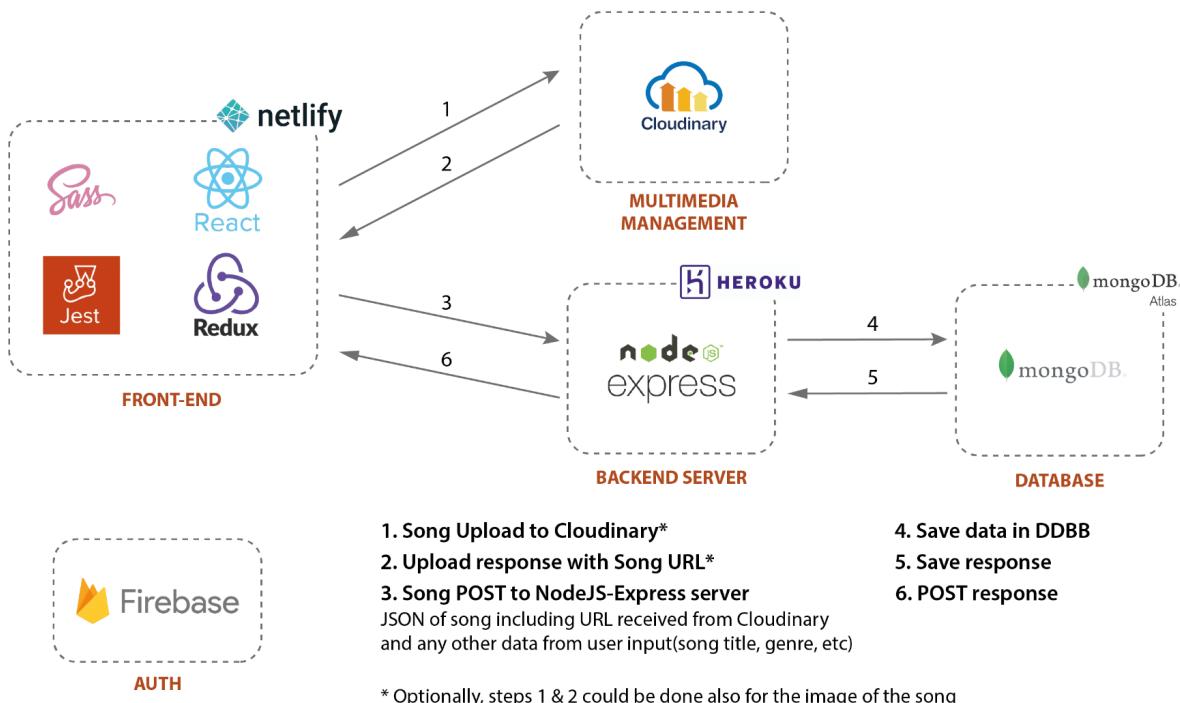
Registro usuario



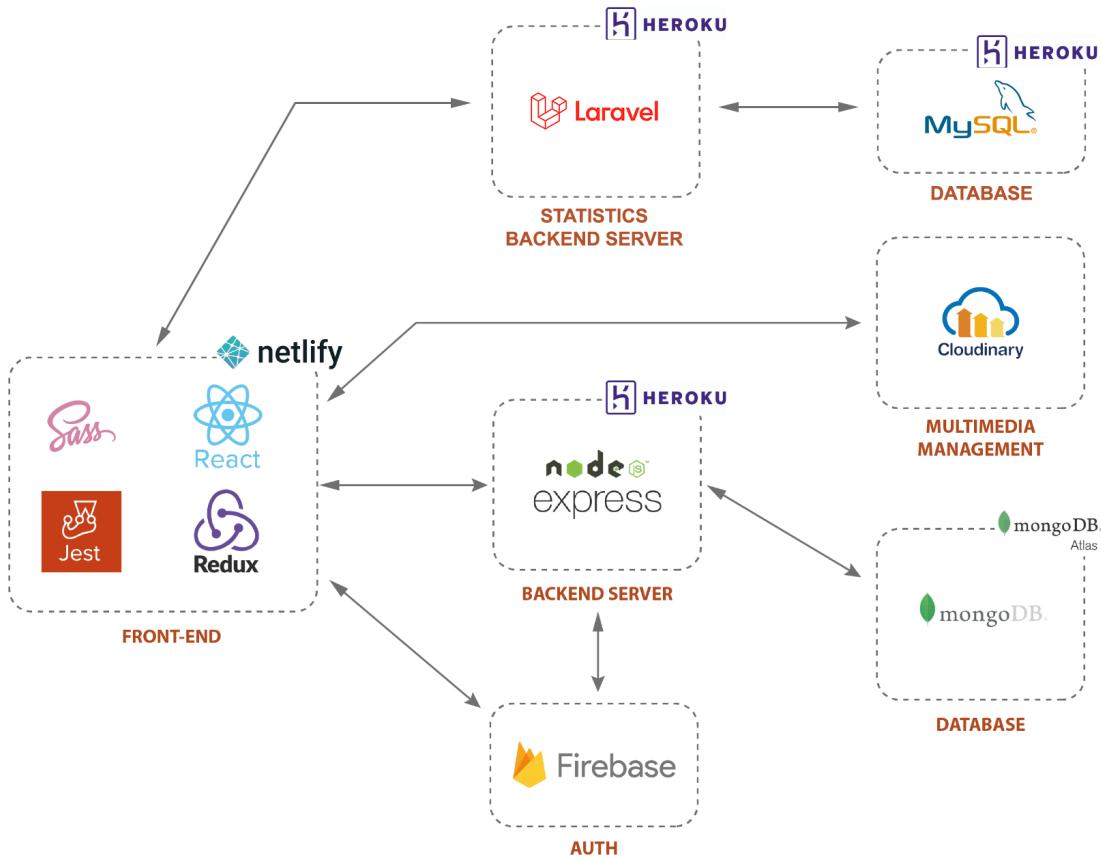
Login Usuario



Subida de canción

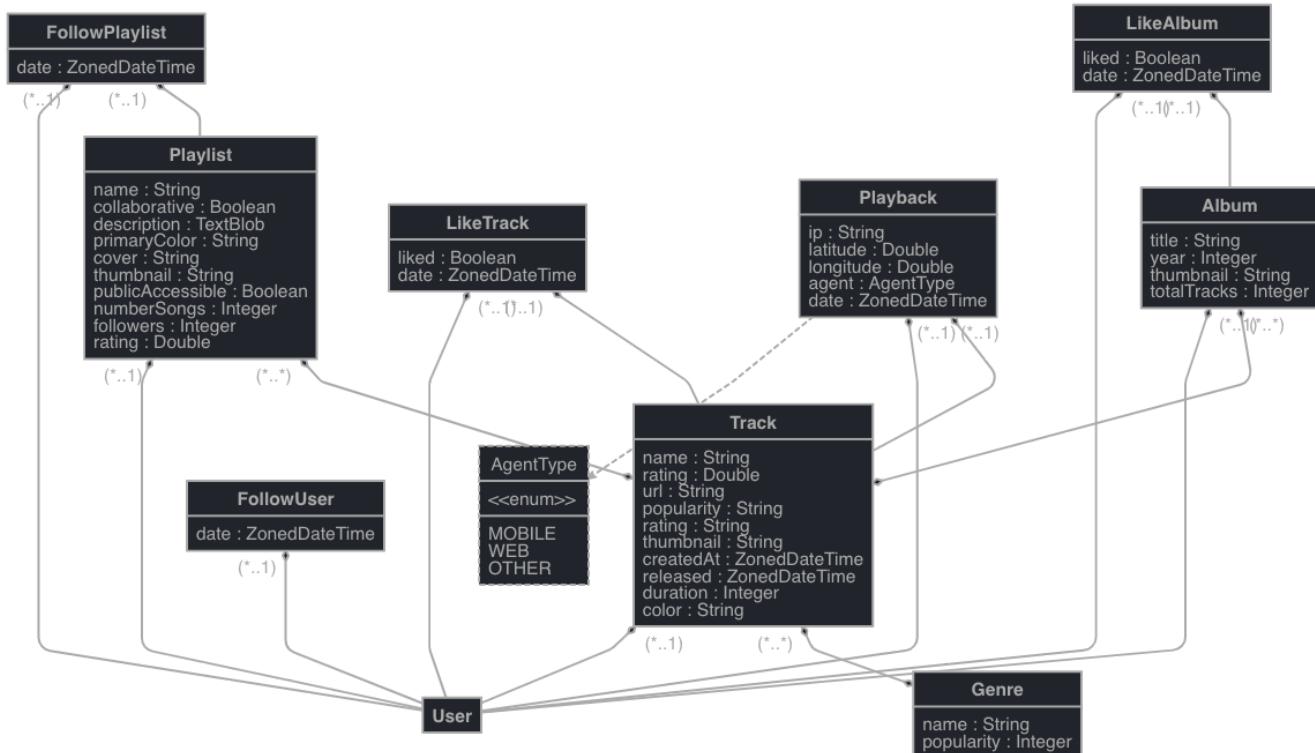


Apéndice I. Arquitectura de Software



<https://ngrok.com/> para trabajo en remoto y conexión entre FrontEnd y BackEnd

Apéndice II. Arquitectura de Base de Datos



<pre> entity Track { name String, rating Double, url String, popularity String, rating String, thumbnail String, createdAt ZonedDateTime, released ZonedDateTime, duration Integer, color String } entity Playlist { name String, collaborative Boolean, description TextBlob, primaryColor String, cover String, thumbnail String, publicAccessible Boolean, numberSongs Integer, followers Integer, rating Double } entity Album { title String, year Integer, thumbnail String, totalTracks Integer } entity LikeTrack { liked Boolean, date ZonedDateTime } entity FollowUser { date ZonedDateTime } entity FollowPlaylist { date ZonedDateTime } entity LikeAlbum { liked Boolean, date ZonedDateTime } </pre>	<pre> entity Genre { name String, popularity Integer } entity Playback { ip String, latitude Double, longitude Double, agent AgentType, date ZonedDateTime } enum AgentType { MOBILE, WEB, OTHER } relationship ManyToOne { Playlist{user(login) required} to User, Album{user(login) required} to User, Track{user(login) required} to User, FollowUser{followed(login) required} to User, FollowUser{user(login) required} to User, LikeTrack{user(login) required} to User, LikeAlbum {user(login) required} to User, Playback{user(login) required} to User, Playback{track(name) required} to Track, LikeTrack{track(name) required} to Track, LikeAlbum{album(title) required} to Album, FollowPlaylist{user(login) required} to User, FollowPlaylist{playlist(name) required} to Playlist } relationship ManyToMany { Playlist{track(name)} to Track{playlist(name)}, Album{track(name)} to Track{album(title)}, Track{genre(name)} to Genre{track(name)} } </pre>
---	---

Apéndice III. Arquitectura de Base de Datos no Relacional

```
entity Track {
    name String,
    rating Double,
    url String,
    popularity String,
    thumbnail String,
    createdAt ZonedDateTime,
    released ZonedDateTime,
    duration Integer,
    color String,
    userId: String,
    genre: { id: String, name: String },
    albums: [albumId: String],
    likedBy: [userId: String]
}
```

```
entity Playlist {
    name String,
    collaborative Boolean,
    description TextBlob,
    primaryColor String,
    cover String,
    thumbnail String,
    publicAccessible Boolean,
    numberSongs Integer,
    followers Integer,
    rating Double,
    userId: String,
    tracks: [trackId: String],
    followedBy: [userId: String]
}
```

```
entity Album {
    title String,
    year Integer,
    thumbnail String,
    totalTracks Integer,
    userId: String,
    likedBy: [userId: String]
}
```

```
entity User {
    ...
    following: [userId: String]
    followedBy: [userId: String]
}
```

```
entity Genre {
    name String,
    popularity Integer
}

entity Playback {
    ip String,
    latitude Double,
    longitude Double,
    agent AgentType,
    date ZonedDateTime
    trackId: String
}

enum AgentType {
    MOBILE, WEB, OTHER
}
```

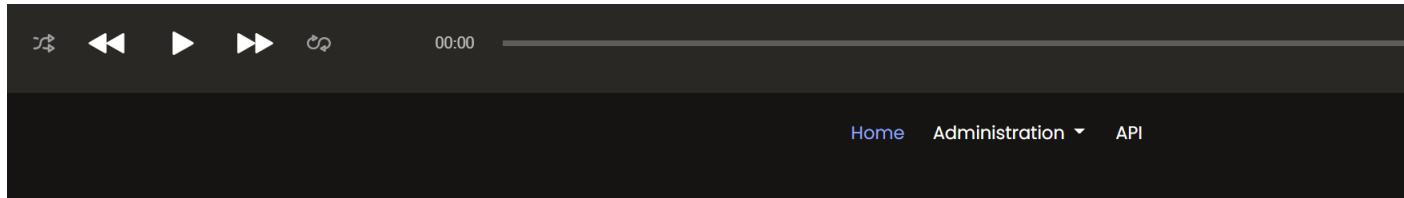
Apéndice IV. Arquitectura de la API

Con el objetivo de que podáis trabajar de una manera muy bien enfocada a la hora de desarrollar vuestro backend y frontend, os proporcionamos un diseño y una implementación de referencia de la API de Apollofy.

Esta implementación de la API se ha desarrollado con Java y Spring Boot. La API que comunica el frontend y el backend se basa en HTTP y JSON. Dado que HTTP es un protocolo de comunicaciones estándar y JSON es un formato de datos estándar, la interfaz es independiente de la implementación interna específica que se ha utilizado en el backend.

Os proporcionamos esta API de referencia, ya que os guiará en el desarrollo de vuestro proyecto. Podéis hacer todas las pruebas necesarias con Swagger, Postman o cualquier otro software para desarrollar/testear APIs.

[URL de la API:](#)



Hacer click en API para ir directos a la herramienta Swagger

Se ruega hacer un uso responsable, ya que es un recurso compartido para toda la promoción. Muchas gracias.

Registro de usuario

Para la creación de usuarios deberemos dirigirnos al apartado de "account-resource", que al pulsar sobre él nos aparecerán las peticiones HTTP que podremos realizar:

Si observamos, crear una nueva cuenta solo requiere realizar una petición POST sobre el endpoint de **/api/account**. Ya que en muchas de estas llamadas es necesario enviar un objeto de datos, para conocer qué parámetros lo conforman será necesario pulsar sobre el endpoint que deseemos realizar la petición, en este caso, sobre **saveAccount**.

Una vez se haya abierto el desplegable, aparecerán los parámetros a añadir sobre el *body* de la llamada para que esta se pueda realizar correctamente.

GET	<code>/api/account getAccount</code>
POST	<code>/api/account saveAccount</code>
POST	<code>/api/account/change-password changePassword</code>
POST	<code>/api/account/reset-password/finish finishPasswordReset</code>
POST	<code>/api/account/reset-password/init requestPasswordReset</code>
GET	<code>/api/activate activateAccount</code>
GET	<code>/api/authenticate isAuthenticated</code>
POST	<code>/api/register registerAccount</code>

POST `/api/account saveAccount`

Parameters Try it out

Name	Description
userDTO * required object (body)	userDTO Example Value Model

```
{
  "activated": true,
  "authorities": [
    "string"
  ],
  "createdBy": "string",
  "createdDate": "2021-04-08T11:00:44.037Z",
  "email": "string",
  "firstName": "string",
  "followers": 0,
  "following": 0,
  "id": 0,
  "imageUrl": "string",
  "langKey": "string",
  "lastModifiedBy": "string",
  "lastModifiedDate": "2021-04-08T11:00:44.037Z",
  "lastName": "string",
  "login": "string",
  "playlists": 0,
  "tracks": 0
}
```

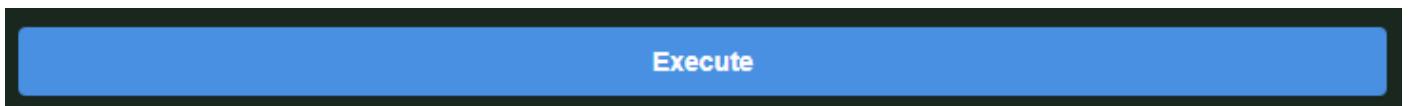
Parameter content type application/json

Para desbloquear la escritura de los datos y poder ejecutar la petición deberemos pulsar sobre el botón de **Try it out**.

A pesar de que el modelo de usuario que aparece como ejemplo contiene una gran cantidad de parámetros, es posible, en este, caso reducirlo al objeto que se muestra a continuación:

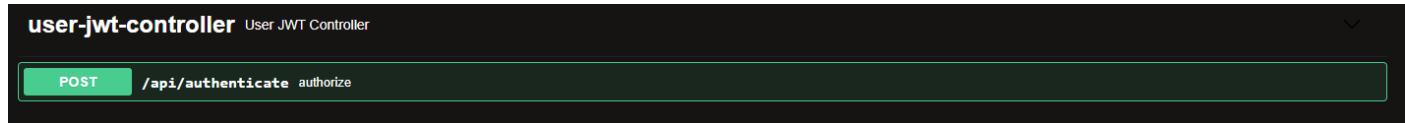
```
{
  "login": "username",
  "email": "username@gmail.com",
  "password": "userpassword",
  "langKey": "en"
}
```

De esta manera, cuando hayamos introducido la información de un nuevo usuario, pulsaremos sobre el botón de **Execute**.



Inicio de sesión

Una vez se haya realizado el registro de manera correcta, ya podremos probar de autenticarnos con el usuario previamente creado. Para ello, nos deberemos dirigir al apartado de **user-jwt-controller**, donde se encuentra el endpoint destinado al inicio de sesión.

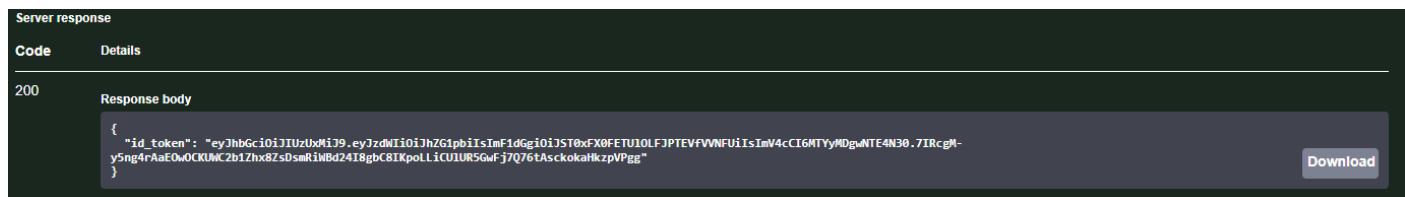


The screenshot shows a dark-themed API documentation interface. At the top, it says "user-jwt-controller User JWT Controller". Below that, there's a button labeled "POST /api/authenticate authorize".

De la misma manera que se ha visto en el apartado anterior, pulsamos sobre el recuadro de la petición para que se despliegue todo su contenido, seguidamente sobre el botón de “Try it out” para desbloquear la escritura y, finalmente, introducimos los datos.

En este caso, el resultado es importante no solo para ver que la petición se ha realizado de manera correcta, sino también porque en el cuerpo de la respuesta se devolverá el token necesario para realizar cualquier futura llamada al backend.

```
{
  "password": "string",
  "rememberMe": true,
  "username": "string"
}
```



The screenshot shows a "Server response" section. It has tabs for "Code" (set to 200) and "Details". Under "Code", it says "Response body". The response body is a JSON object with one key, "id_token", containing a long string of characters. There is also a "Download" button.

Creación de canciones

Al no haber una diferenciación explícita a nivel de backend entre artistas y usuarios, cualquier persona registrada tiene la posibilidad de subir canciones. Si fuera esa nuestra intención, simplemente nos tendríamos que dirigir al apartado de “Tracks” y hacer uso del endpoint de **/api/tracks (POST)**. Así, una vez introducidos los datos requeridos (ciertos parámetros que se generan automáticamente durante la creación se pueden obviar, por ejemplo, el id, fechas de creación...) ya habríamos creado nuestra primera canción.



The screenshot shows a dark-themed API documentation interface. At the top, it says "POST /api/tracks Creates a track".

Creación de playlists

Sin muchos cambios respecto al caso anterior, crear listas de reproducción solo requiere de un añadido más, especificar qué canciones queremos que estén en ella. Si bien en el body de ejemplo aparecen gran parte de los parámetros para cada canción seleccionada, se puede simplificar incorporando únicamente el identificador de la canción correspondiente (la url también pero sin tener que saber su valor).

```
{
  "cover": null,
  "description": "This is a simple description",
  "name": "Admin Test List 2",
  "publicAccessible": true,
  "thumbnail": null,
  "tracks": [
    {
      "id": 1,
      "url": "string"
    }
  ]
}
```

account-resource Account Resource

GET	/api/account	getAccount
POST	/api/account	saveAccount
POST	/api/account/change-password	changePassword
POST	/api/account/reset-password/finish	finishPasswordReset
POST	/api/account/reset-password/init	requestPasswordReset
GET	/api/activate	activateAccount
GET	/api/authenticate	isAuthenticated
POST	/api/register	registerAccount

genre-resource Genre Resource

GET	/api/genres	getAllGenres
POST	/api/genres	Creates a Genre
PUT	/api/genres	updateGenre
GET	/api/genres/{id}	getGenre
DELETE	/api/genres/{id}	deleteGenre
GET	/api/genres/{id}/tracks	getTracksByGenreId

me-resource Me Resource

GET	/api/me/followers	Shows all the current user followers
GET	/api/me/followings	Shows all the following users by the current user
GET	/api/me/playlists	Shows own playlists
GET	/api/me/playlists/following	Shows all the following playlists by the current user
GET	/api/me/playlists/{id}	Shows the desired playlist by the requested id
GET	/api/me/tracks	Shows own tracks
GET	/api/me/tracks/liked	Shows all the liked tracks by the current user
GET	/api/me/tracks/{id}	Shows the desired track by the requested id

playback-resource Playback Resource

GET /api/playbacks Finds playbacks by criteria

playlist-resource Playlist Resource

GET /api/playlists Shows playlists

POST /api/playlists createPlaylist

PUT /api/playlists updatePlaylist

GET /api/playlists/{id} getPlaylist

DELETE /api/playlists/{id} deletePlaylist

GET /api/playlists/{id}/follow Check if current user follows the playlist

PUT /api/playlists/{id}/follow Follows the playlist by id

search-resource Search Resource

GET /api/search Search globally

track-resource Track Resource

GET /api/tracks Shows tracks

POST /api/tracks Creates a track

PUT /api/tracks Updates a track

GET /api/tracks/{id} Shows the desired track by id param

DELETE /api/tracks/{id} Deletes a track by id

GET /api/tracks/{id}/like Check if current user liked a track

PUT /api/tracks/{id}/like Likes the track by id

PUT /api/tracks/{id}/play Plays the track by id

Apéndice V. Backend - Frontend development dependency

La preparación y organización de las tareas de un proyecto es de vital importancia para su correcta realización, más aún cuando las partes involucradas pueden llegar a depender entre ellas para seguir avanzando en su desarrollo. Si bien en el momento de establecer el orden de las funcionalidades a realizar, se puede tratar de prevenir las dependencias a lo largo de la duración del proyecto, son muchos los condicionantes que pueden (y llegan) a aparecer y que obligan a los equipos de trabajo a hacer cambios en su planificación.

Para el proyecto que vais a realizar, la dependencia más notable se encuentra en la comunicación entre back-end y front-end, sobre todo al inicio del desarrollo, pues el primero todavía no tendrá implementados los **endpoints** que la plataforma va a llamar para obtener los datos. Ante este contexto, son diversas las opciones que se podrían tomar para evitar que el equipo de frontend se viera impedido a avanzar.

En el momento de empezar la implementación, es probable que los componentes UI que se van a emplear para dar forma a las distintas páginas todavía no estén creados. Esto siempre puede dar un margen de tiempo para que el equipo de back-end codifique algunos de los **endpoints** y estos estén disponibles al finalizar la creación de los componentes. Incluso haciendo uso de librerías con componentes ya creados, siempre es necesario adaptarlos a las necesidades que presenta el problema en cuestión.

Por otra parte, también se ha puesto a disposición de los desarrolladores un **backend** totalmente funcional cuyas funcionalidades han sido de ejemplo para el proyecto en cuestión. Si bien el retorno de los datos de los **endpoint** disponibles (a nivel de nomenclatura de los parámetros devueltos, por ejemplo) podrá diferir con el que devuelva el **backend** creado por el equipo de trabajo asignado, esto permitiría al equipo de **frontend** asignar y hacer las llamadas necesarias para cada página de la plataforma.

Otra aproximación que se puede adoptar pasa por asentar al inicio del proyecto qué entidades conformarán la base de datos, qué llamadas estarán disponibles y qué tipo de objetos se devolverán. Esto, a nivel de **frontend**, ofrece la posibilidad de poder generar ficheros de ejemplo de resultados de llamadas para los **endpoints** existentes. Así, la dependencia entre los equipos se libera y permite el correcto desarrollo de ambas partes.

Para este último caso, una biblioteca de gran utilidad es [**Mock Service Worker**](#). Su funcionamiento se basa en capturar peticiones HTTP y proporcionar una respuesta definida por el desarrollador, normalmente similar o igual a la que se devolvería al hacer la petición real.

[Ejemplos con React para mockear una REST API](#)

Apéndice VI. Epics y User Stories optionales

#	Epic title	Epic description	Priority
1	Pantalla Home principal avanzada	Pantalla de bienvenida, donde unifica distintas visualizaciones de top playlists, top usuarios, top álbumes y top tracks	COULD HAVE
2	Búsqueda de canciones, usuarios, playlists y álbumes	Buscador avanzado recuperando canciones, usuarios, playlists y álbumes	COULD HAVE
3	Visualización y gestión de usuarios/artistas	Visualización de usuarios/artistas y opciones de seguimiento	SHOULD HAVE
4	Relación de géneros	Relación de géneros con canciones	SHOULD HAVE
5	Publicación, visualización y gestión de géneros	Gestión y creación de nuevos géneros para ser relacionados con las canciones y con los álbumes	COULD HAVE
6	Publicación y gestión de álbumes	Gestión, creación y seguimiento de álbumes a partir de un conjunto de canciones	COULD HAVE
7	Publicación, visualización y gestión de vídeos	Permitir publicación de vídeos empleando el mismo sistema de gestión que las canciones	COULD HAVE
8	Reproducción de vídeos	Reproducción de vídeos mostrando el vídeo en vez de la portada de la canción	COULD HAVE
9	Sistema de notificaciones	Distintas notificaciones a partir de los seguimientos de playlists y artistas, así como notificaciones relacionadas con marketing	COULD HAVE
10	Multi-idioma	Aplicación disponible en varios idiomas	SHOULD HAVE

Historias de Usuario opcionales

Este [conjunto de historias de usuarioopcionales](#) representa una [guía o roadmap](#) que os permitirá seguir desarrollando vuestra plataforma de música más allá del Máster. Las historias de usuario MUST HAVE os guiarán para desarrollar un [MVP](#) que podréis incluir con éxito en vuestro portfolio personal. Las historias de usuario opcionales también pueden ser útiles, en el caso de que finalicéis antes de tiempo las funcionalidades obligatorias.

#	Epic	User story title	User story description	SP	Priority	
1	(1) Registro y Autenticación	Guardado de credenciales	El usuario podrá permitir que se recuerden los credenciales en la pantalla de inicio de sesión	5	SHOULD HAVE	
2	(2) Pantalla Home principal	Sección “Trending” en pantalla home	La sección “Trending” (antigua “By Sallyfy”) mostrará 4 perfiles aleatorios sobre los 20 más reproducidos que dispongan al menos de imagen de perfil y de 5 canciones en su cuenta	13	COULD HAVE	
3	(2) Pantalla Home principal	Sección “Playlists populares” en pantalla home	La sección “Playlists populares” deberá mostrar un top de las playlists con más seguidores.	8	SHOULD HAVE	
4	(2) Pantalla Home principal	Sección “Artistas populares” en pantalla home	La sección “Artistas populares” deberá mostrar un top de l@s artistas con más seguidores.	8	SHOULD HAVE	
5	(2) Pantalla Home principal	Sección “Playlists que sigo” en pantalla home	La sección “Playlists que sigo” mostrará el listado de playlists que el usuario sigue	8	SHOULD HAVE	
6	(4) Publicación, visualización y gestión de canciones	Subida de canciones avanzada	El usuario podrá subir canciones con URL privada (signed URL de Cloudinary). La subida de la gestión pasa a ser gestionada por el Backend.	21	COULD HAVE	
7	(4) Publicación, visualización y gestión de canciones	Lectura automática de género de canción	El usuario, al subir una canción, podrá recuperar el género de la canción automáticamente a partir de una integración con una IA que detecte géneros de canciones	21	COULD HAVE	

SÓLO INFORMATIVO

8	(4) Publicación, visualización y gestión de canciones	Lectura automática de metadatos de canción	El usuario, al subir una canción, podrá introducir automáticamente los datos de la canción a partir de los metadatos del mp3	21	COULD HAVE	
9	(4) Publicación, visualización y gestión de canciones	Orden manual en "Mis canciones"	El usuario podrá definir su propio orden manual en el listado de "Mis canciones"	13	COULD HAVE	
10	(4) Publicación, visualización y gestión de canciones	Ordenar en listado de "Mis canciones"	En "Mis canciones" el usuario podrá ordenar estas canciones por orden alfabético, ascendente o descendente, o mantener el orden manual	5	COULD HAVE	
11	(4) Publicación, visualización y gestión de canciones	Búsqueda de canciones en "Mis canciones"	En el listado de "Mis canciones" el usuario podrá realizar una búsqueda por nombre	5	COULD HAVE	
12	(4) Publicación, visualización y gestión de canciones	Compartir canción dentro del diálogo de información de canción	El usuario podrá compartir una canción accediendo al diálogo de información de canción	5	COULD HAVE	
13	(5) Reproductor de canciones	Canción atrás	El usuario dispondrá del control de ir una canción atrás	13	SHOULD HAVE	
14	(5) Reproductor de canciones	Canción hacia adelante	El usuario dispondrá de un control de ir una canción hacia adelante	5	SHOULD HAVE	
15	(5) Reproductor de canciones	Castear reproducción	El usuario dispondrá de opción de castear a un Chromecast la reproducción actual	21	COULD HAVE	
16	(5) Reproductor de canciones	Reproducción en bucle de canción	El usuario podrá seleccionar que la reproducción se realice en modo bucle de canción, haciendo que cada vez que finaliza la canción esta vuelva a reproducirse	13	COULD HAVE	
17	(5) Reproductor de canciones	Reproducción en bucle de playlist/álbum	El usuario podrá seleccionar que la reproducción se realice en modo bucle de playlist/álbum, haciendo que cada vez que finalice la playlist o álbum seleccionada este vuelva a reproducirse	13	COULD HAVE	
18	(5) Reproductor de canciones	Reproducción en modo aleatorio	El usuario podrá seleccionar que la reproducción se realice en modo aleatorio, haciendo que la	13	COULD HAVE	

			reproducción de una playlist o álbum seleccionada se reproduzca de manera aleatoria			
19	(5) Reproductor de canciones	Sistema de cola de reproducción	El usuario podrá disponer y visualizar una cola de reproducción, que se generará automáticamente cada vez que el usuario decida reproducir una canción	21	COULD HAVE	
20	(5) Reproductor de canciones	Añadir canción a la cola de reproducción	El usuario podrá añadir canciones a la cola de reproducción, añadiendo estas justo a continuación de la actual	5	COULD HAVE	✓ ✓ ✓
21	(5) Reproductor de canciones	Cambiar posición de canción en la cola de reproducción	El usuario podrá modificar la posición de una canción dentro de la cola de reproducción	5	COULD HAVE	✓ ✓ ✓
22	(5) Reproductor de canciones	Guardado de cola de reproducción tras reabrir la aplicación	El usuario podrá disponer de la misma cola de reproducción que tenía en el último uso de la aplicación, siendo esta guardada automáticamente en caso de que el usuario cierre la aplicación	5	COULD HAVE	✓ ✓ ✓
23	(6) Búsqueda de canciones, usuarios, playlists y álbumes	Búsqueda por géneros	El usuario en caso de que no introduzca ninguna información en el panel de búsqueda, en este por defecto aparecerá un listado de géneros de música de la aplicación	13	COULD HAVE	✓ ✓
24	(7) Publicación, visualización y gestión de playlists	Ordenar en listado de playlists	En el listado de playlists el usuario podrá ordenar estas por orden alfabético, ascendente o descendente, o mantener el orden manual	5	COULD HAVE	✓
25	(7) Publicación, visualización y gestión de playlists	Búsqueda en listado de playlists	En el listado de playlists el usuario podrá realizar una búsqueda por nombre, filtrando desde el listado de playlists	5	COULD HAVE	✓ ✓ ✓
26	(7) Publicación, visualización y gestión de playlists	Panel "Más información" en visualización de Playlist	En la vista de playlist el usuario podrá ver la vista "Más información", visualizando el creador y la descripción de la playlist	3	COULD HAVE	✓
27	(7) Publicación, visualización y gestión de playlists	Cambio de privacidad en vista de Playlist	En la vista de playlist, el usuario (si es propietario) podrá cambiar la privacidad de pública a privada y a la inversa, afectando a su visibilidad	5	SHOULD HAVE	✓ ✓ ✓

28	(5) Reproductor de canciones	Reproducción Shuffle dentro de Playlist	En la vista de playlist, el usuario podrá seleccionar el botón Shuffle para reproducir la lista en orden aleatorio	8	COULD HAVE	
29	(7) Publicación, visualización y gestión de playlists	Subir automáticamente una canción al añadirla dentro de Playlist	En la vista de playlist, añadir canción, el usuario podrá también subir una track y que esta se añada automáticamente a la playlist	5	COULD HAVE	✓ ✓ ✓
30	(7) Publicación, visualización y gestión de playlists	Ordenación de tracks dentro de Playlist	En la vista de playlist, el usuario podrá ordenar las tracks por orden alfabético, duración y alfabéticamente por nombre de artista	5	COULD HAVE	✓ ✓ ✓ ✓
31	(7) Publicación, visualización y gestión de playlists	Ordenación de tracks propia dentro de Playlist	En la vista de playlist, el usuario podrá ordenar tracks por orden propio del usuario	8	COULD HAVE	✓ ✓ ✓
32	(4) Publicación, visualización y gestión de canciones	Acceder a vista de artista dentro del diálogo de información de canción	El usuario podrá acceder a la vista de un artista desde el diálogo de información de canción	1	COULD HAVE	
33	(8) Visualización y gestión de usuarios/artistas	Top 5 canciones de artista	El usuario podrá visualizar un top de 5 canciones más populares en la vista del artista	8	COULD HAVE	
34	(8) Visualización y gestión de usuarios/artistas	Seguir a otros usuarios	El usuario podrá seguir a otros usuarios presionando el botón de seguir	13	SHOULD HAVE	
35	(8) Visualización y gestión de usuarios/artistas	Visualización de seguidores	El usuario podrá acceder a un listado para visualización de seguidores de un usuario o su propio usuario, pudiendo acceder también a estos usuarios	13	SHOULD HAVE	
36	(8) Visualización y gestión de usuarios/artistas	Visualización de siguiendo	El usuario podrá acceder a un listado para visualización de seguidores de un usuario o de su propio usuario, pudiendo acceder también a estos usuarios	8	SHOULD HAVE	
37	(8) Visualización y gestión de usuarios/artistas	Compartición de perfil	El usuario podrá “compartir” un perfil de usuario, generando un link propio	5	COULD HAVE	

38	(9) Publicación, visualización y gestión de géneros	Gestión de géneros	El usuario administrador podrá editar, añadir y eliminar géneros	21	COULD HAVE	
39	(10) Publicación y gestión de álbumes	[Opcional]	[Opcional]	30	COULD HAVE	
40	(11) Publicación, visualización y gestión de vídeos	[Opcional]	[Opcional]	40	COULD HAVE	X
41	(12) Reproducción de vídeos	[Opcional]	[Opcional]	21	COULD HAVE	X
42	(13) Visualización de estadísticas	Visualización de top géneros reproducidos por usuario	El usuario podrá visualizar los top géneros reproducidos de sus canciones en un pie chart	13	COULD HAVE	
43	(13) Visualización de estadísticas	Visualización de las canciones más reproducidas por el usuario	El usuario podrá visualizar un gráfico de barras de las canciones más reproducidas por el usuario		COULD HAVE	
44	(13) Visualización de estadísticas	Visualización de reproducción de canciones del usuario en un mapa de calor en tiempo real	El usuario podrá acceder a un mapa interactivo en el que a partir de la selección de una de sus canciones visualizar en formato de mapa de calor la geolocalización de reproducción de estas canciones	21	COULD HAVE	
45	(15) Sistema de notificaciones	Notificaciones de nuevas canciones	El usuario recibirá notificaciones de nuevas canciones de los usuarios que siga, haciendo click reproducirá la canción	13	COULD HAVE	
46	(15) Sistema de notificaciones	Notificaciones de nuevas playlists	El usuario recibirá notificaciones de nuevas playlists de los usuarios que siga, haciendo click abrirá la lista y la reproducirá	13	COULD HAVE	
47	(15) Sistema de notificaciones	Notificaciones de nuevos álbumes	El usuario recibirá notificaciones de nuevos álbumes de los usuarios que siga, haciendo click abrirá y reproducirá el álbum	13	COULD HAVE	

48	(15) Sistema de notificaciones	Notificación de seguimiento	El usuario recibirá una notificación cada vez que un usuario nuevo le siga	13	COULD HAVE	
49	(16) Multi-idioma		El usuario podrá decidir el idioma de la aplicación, cambiando automáticamente el idioma de esta	5	COULD HAVE	

Mockups-Stories Orientation

Registro y Autenticación

Epic	Registro y Autenticación	SHOULD HAVE
User Story Title	Guardado de credenciales	
API Endpoint	/api/authenticate	

Recuérdame **INICIAR SESIÓN**

Epic	Registro y Autenticación	COULD HAVE
User Story Title	Recuperación de password	
API Endpoint	/api/account/change-password	

Restablecer contraseña

Introduce tu **nombre de usuario de Spotify** o la **dirección de correo electrónico** que usaste para registrarte. Te enviaremos un correo electrónico con tu nombre de usuario y un enlace para restablecer tu contraseña.

Dirección de correo o nombre de usuario

No soy un robot



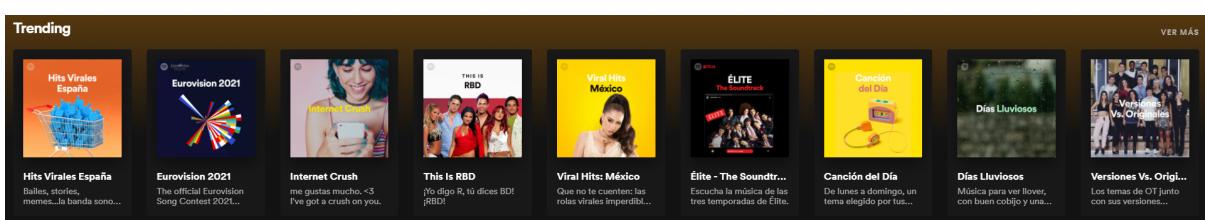
reCAPTCHA
Privacidad - Términos

ENVIAR

Si necesitas más ayuda, contacta con [el servicio de soporte de Spotify](#).

Pantalla Home principal

Epic	Pantalla Home principal	COULD HAVE
User Story Title	Sección “Trending” en pantalla home	
API Endpoint	/api/playlists/trending	



Epic	Pantalla Home principal	COULD HAVE
User Story Title	Sección “Playlists populares” en pantalla home	
API Endpoint	/api/playlists/popular	

Playlists populares

Women of Classical From the sacred works... Hildegarde von Bingen...	Chill Singer-Songwriter A fresh & mellow mix from some of today's...	Women of Pop Kick back with the queens of pop.	Legendary Women... The Women who put Country Music on the...	Young Nashville Music from Nashville's emerging Singers and...	fem. A new generation making their own rules...	Singer Songwriter ... Take a good cup of coffee and enjoy some...	This Is Taylor Swift The essential tracks, all in one playlist.

Epic	Pantalla Home principal	COULD HAVE
User Story Title	Sección "Artistas populares" en pantalla home	
API Endpoint	/api/users/popular	

Top Artistas

							VER MÁS
KG0516 KAROL G	Warm Up Bad Gyal	Justice Justin Bieber	Mis Manos Camilo	El Madrileño C. Tangana	Los Dioses Anuel AA, Ozuna	11 RAZONES Aitana	

Epic	Pantalla Home principal	COULD HAVE
User Story Title	Sección "Playlists que sigo" en pantalla home	
API Endpoint	/api/me/playlists/following	

Playlists que sigues

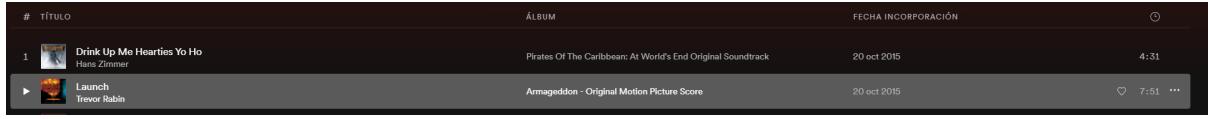
									VER MÁS
Legendary Great songs by indisputable music...	Best of Rock: 1972 Big rock tracks from 1972.	Classic Acoustic Kick back with a chill collection of iconic...	70s Rock Anthems These songs rocked the 70s. Cover: Queen	60s Rock Anthems These songs rocked the '60s. Cover: Jim...	Raised on Rock A soundtrack that both you and your little one...	Best of Rock: 1970 Big rock tracks from 1970. Cover: Creedenc...	Best of Rock: 1971 Big rock tracks from 1971. Cover: Led...	This Is Bob Dylan The man, the myth, the legend.	

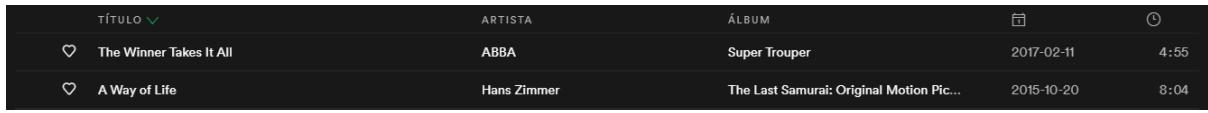
Publicación, visualización y gestión de canciones

Epic	Publicación, visualización y gestión de canciones	COULD HAVE
User Story Title	Lectura automática de género de canción	

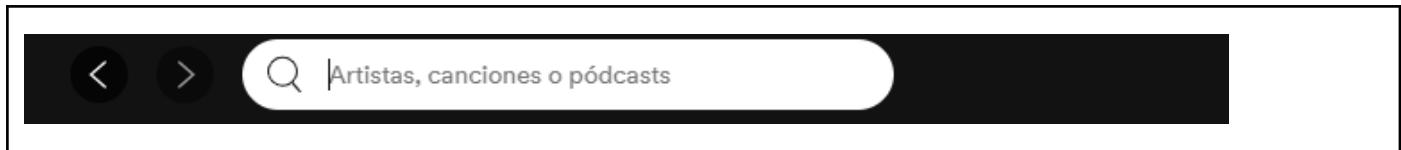
API Endpoint	n/a

Epic	Publicación, visualización y gestión de canciones	COULD HAVE
User Story Title	Lectura automática de metadatos de canción	
API Endpoint	n/a	

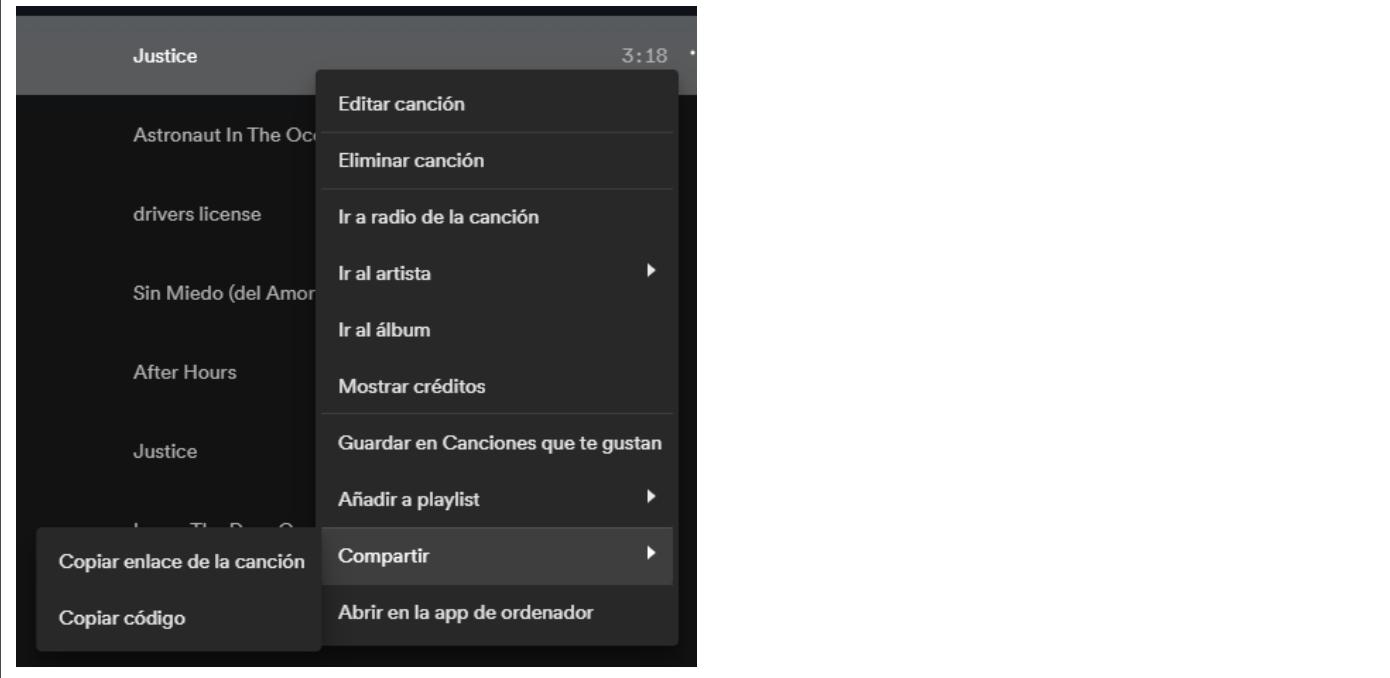
Epic	Publicación, visualización y gestión de canciones	COULD HAVE	
User Story Title	Orden manual en “Mis canciones”		
API Endpoint	n/a		
			

Epic	Publicación, visualización y gestión de canciones	COULD HAVE	
User Story Title	Ordenar en listado de “Mis canciones”		
API Endpoint	n/a		
			

Epic	Publicación, visualización y gestión de canciones	COULD HAVE
User Story Title	Búsqueda de canciones en “Mis canciones”	
API Endpoint	api/search	



Epic	Publicación, visualización y gestión de canciones	COULD HAVE
User Story Title	Compartir canción dentro del diálogo de información de canción	
API Endpoint	n/a	



Reproductor de Canciones

Epic	Reproductor de canciones	SHOULD HAVE
User Story Title	Canción atrás	
API Endpoint	n/a	

Epic	Reproductor de canciones	SHOULD HAVE
User Story Title	Canción hacia adelante	
API Endpoint	n/a	

Epic	Reproductor de canciones	COULD HAVE
User Story Title	Castear reproducción	
API Endpoint	n/a	

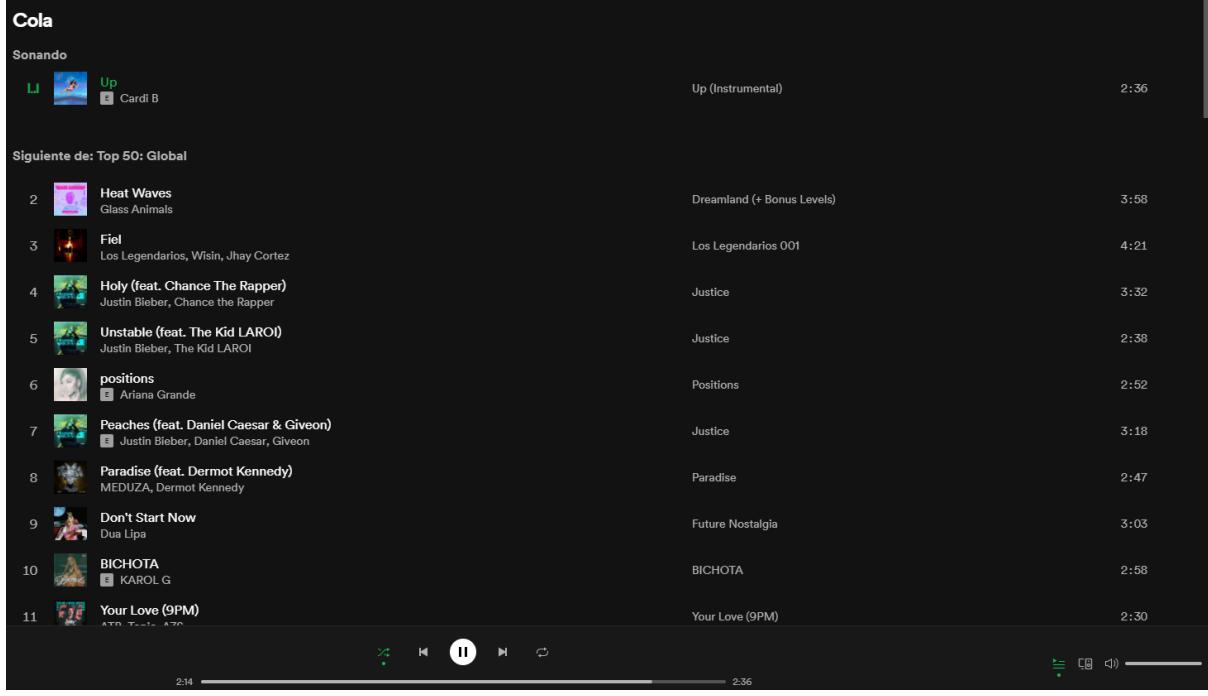
Epic	Reproductor de canciones	COULD HAVE
User Story Title	Reproducción en bucle de canción	
API Endpoint	n/a	

Epic	Publicación, visualización y gestión de canciones	COULD HAVE
User Story Title	Reproducción en bucle de playlist/álbum	
API Endpoint	n/a	

Epic	Publicación, visualización y gestión de canciones	COULD HAVE
User Story Title	Reproducción en modo aleatorio	
API Endpoint	n/a	



Epic	Publicación, visualización y gestión de canciones	COULD HAVE
User Story Title	Sistema de cola de reproducción	
API Endpoint	n/a	

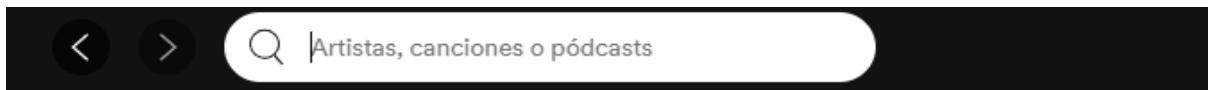


Epic	Reproductor de canciones	COULD HAVE
User Story Title	Añadir canción a la cola de reproducción	
API Endpoint	n/a	

Epic	Reproductor de canciones	COULD HAVE
User Story Title	Cambiar posición de canción en la cola de reproducción	
API Endpoint	n/a	

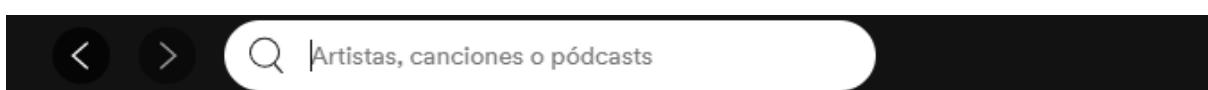
Epic	Reproductor de canciones	COULD HAVE
User Story Title	Guardado de cola de reproducción tras reabrir la aplicación	
API Endpoint	n/a	

Búsqueda de canciones, usuarios, playlists y álbumes

Epic	Búsqueda de canciones, usuarios, playlists y álbumes	COULD HAVE
User Story Title	Búsqueda por géneros	
API Endpoint	/api/search -- or -- /api/genres/{id}/tracks	
		

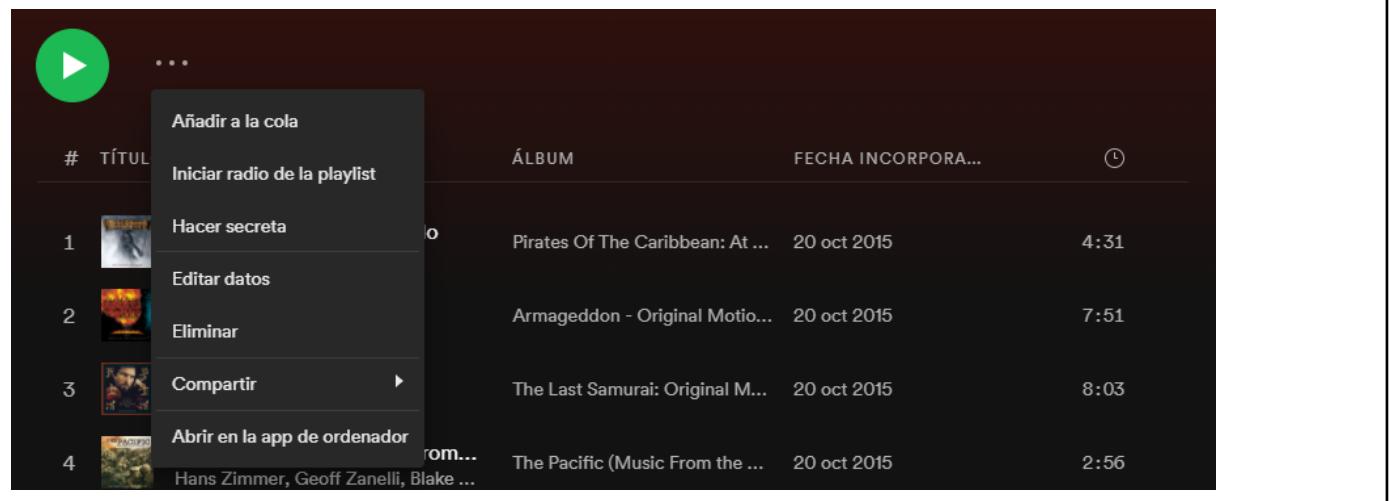
Publicación, visualización y gestión de playlists

Epic	Publicación, visualización y gestión de playlists	COULD HAVE
User Story Title	Ordenar en listado de playlists	
API Endpoint	n/a	

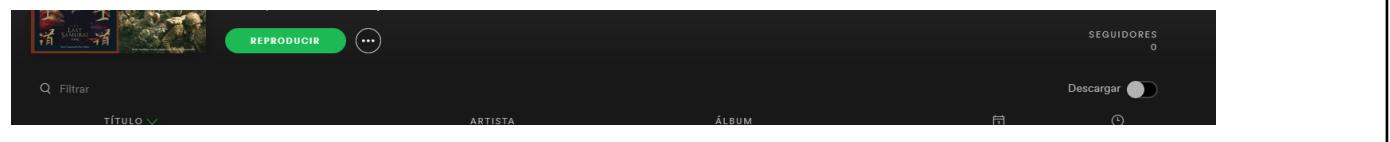
Epic	Publicación, visualización y gestión de playlists	COULD HAVE
User Story Title	Búsqueda en listado de playlists	
API Endpoint	/api/search -- or -- /api/playlists (with filter params)	
		

Epic	Publicación, visualización y gestión de playlists	COULD HAVE
User Story Title	Panel “Más información” en visualización de Playlist	
API Endpoint	/api/playlists/{id}	

Epic	Publicación, visualización y gestión de playlists	SHOULD HAVE
User Story Title	Cambio de privacidad en vista de Playlist	
API Endpoint	/api/playlists/{id}	



Epic	Publicación, visualización y gestión de playlists	COULD HAVE
User Story Title	Reproducción Shuffle dentro de Playlist	
API Endpoint	/api/authenticate	



Epic	Publicación, visualización y gestión de playlists	COULD HAVE
User Story Title	Subir automáticamente canción al añadirla dentro de Playlist	
API Endpoint	/api/playlists	

Epic	Publicación, visualización y gestión de playlists	COULD HAVE
User Story Title	Ordenación de tracks dentro de Playlist	
API Endpoint	n/a	

Epic	Publicación, visualización y gestión de playlists	COULD HAVE
User Story Title	Ordenación de tracks propia dentro de Playlist	
API Endpoint	n/a	

Visualización y gestión de usuarios/artistas

Epic	Visualización y gestión de usuarios/artistas	COULD HAVE
User Story Title	Top 5 canciones de artista	
API Endpoint	/api/users	

Epic	Visualización y gestión de usuarios/artistas	SHOULD HAVE
User Story Title	Seguir a otros usuarios	
API Endpoint	/api/users/{id}/follow	



Epic	Visualización y gestión de usuarios/artistas	SHOULD HAVE
User Story Title	Visualización de seguidores	
API Endpoint	/api/users/{id}	



Epic	Visualización y gestión de usuarios/artistas	SHOULD HAVE
User Story Title	Visualización de siguiendo	
API Endpoint	/api/me/playlists/following	

