# **Objetivos**

En este proyecto he desarrollado una aplicación web que utiliza datos de álbumes y reseñas de Amazon Music y AOTY para permitir búsquedas y recomendaciones de álbumes a los usuarios de estos servicios. Para ello he utilizado Django, BeautifulSoup, Whoosh y Sistemas de Recomendación. Asimismo, he realizado este trabajo de forma autónoma para afianzar los conocimientos sobre estas herramientas que se han utilizado en las clases prácticas de la asignatura. También se puede acceder al proyecto en GitHub a través del siguiente enlace: <https://github.com/darkgigi/proyectoAII>

# **Descripción**

Para la página web he utilizado Django y Bootstrap 5 para los componentes y estilos html. Asimismo, se ha usado una base de datos Sqlite3.

La página tiene un menú con tres partes: Cargar, Buscar y Recomendar. En la parte de la carga se puede cargar la base de datos y se puede cargar el sistema de recomendación. Para la carga de datos he seleccionado los datos de Amazon Music del siguiente enlace:

<https://github.com/caserec/Datasets-for-Recommender-Systems/blob/master/Processed%20Datasets/AmazonMusic.tar.xz>

Se ha realizado la carga de álbumes, géneros, usuarios y reseñas de los archivos con formato csv y json del archivo .tar mencionado.

Debido a la longitud del json (unas 60.000 reseñas) decidí crear un segundo json con la mitad de longitud para permitir realizar una carga completa o parcial de dichos datos, y así permitir la opción de que la carga tarde algo menos.

Además de cargar los datos he seleccionado datos de la página de AOTY (<https://www.albumoftheyear.org/releases/this-week/> ) haciendo scraping con BeautifulSoup. Solo he hecho scraping a la primera página de lanzamientos de álbumes debido a la longitud de los datos de Amazon Music para no aumentar demasiado el tiempo de carga. Además, para cada álbum, también selecciono los datos de las reseñas de la primera página de reseñas por el mismo motivo que expliqué anteriormente. Asimismo, tuve que normalizar las puntuaciones de los usuarios de AOTY para que en vez de estar entre 0 y 100 estuvieran entre 0 y 5.

Toda la carga de datos, el scraping y la creación de los esquemas de Whoosh está encapsulada en una función llamada populateDB, que llama a las funciones de las clases populate.py, scraping.py y whoosh.py.

Para la carga del Sistema de Recomendación utilicé los métodos que se habían enseñado en clases de prácticas.

En cuanto a la búsqueda de datos he implementado una búsqueda de todos los álbumes mostrando el nombre, ID y los géneros de forma paginada, mostrando 100 álbumes por página.

A continuación, las siguientes búsquedas se han realizado con Whoosh.

Asimismo, también se puede buscar un álbum a partir del nombre. Con esta funcionalidad de puede buscar todos los álbumes que contengan alguna palabra en concreto en el nombre.

También se puede buscar álbumes a partir del género. En está parte de la página se puede seleccionar un género de un botón desplegable con todos los géneros de la base de datos.

Finalmente, se puede realizar una búsqueda de las reseñas a partir del usuario. Introduciendo el nombre o el ID del usuario se puede obtener todas las reseñas que corresponden a los usuarios seleccionados. Además, se puede filtrar la búsqueda para que solo se muestren aquellas reseñas que puntúen al álbum con una puntuación dada o superior.

En cuanto al Sistema de recomendación, he utilizado un sistema colaborativo. Se pueden acceder a tres funciones, la primera es recomendar álbumes con un filtrado colaborativo basado en usuarios. Así, introduciendo el ID de un usuario es posible obtener hasta 5 álbumes recomendados basados en las preferencias del resto de usuarios, es decir, se hace la búsqueda basándose en la similitud de los usuarios.

También se puede recomendar álbumes usando un filtrado colaborativo basado en ítems. Al introducir el ID de un usuario, se mostrarán hasta 5 álbumes basándose en la similitud entre ítems y en las puntuaciones de los usuarios.

Además, se puede obtener los 3 álbumes más similares a uno dado al introducir su ID. Para calcular la similitud entre los álbumes se ha utilizado el coeficiente de Pearson, que estaba por defecto en los métodos empleados.

# **Manual de uso**

Para poder ejecutar la página se tendrá que ejecutar los siguientes comandos en la carpeta exterior llamada “ProyectoAII”:

python manage.py makemigrations main

Este comando creará las migraciones de los modelos que se encuentran en la carpeta main.

python manage.py migrate

Con este comando ejecutamos las migraciones y así tendremos la base de datos vacía.

python manage.py createsuperuser

Opcionalmente, se puede ejecutar este comando para crear un superusuario con permisos de admin para poder visualizar los datos de la base de datos de forma más cómoda si se quisiera.

python manage.py runserver localhost:8000

Finalmente, con este comando se puede ejecutar la aplicación en local en el puerto 8000.