**IDS 561 Analytics for Big Data**

**Fall 2019**

****

**MyTunes : Music Recommendation Engine**

**The App that sings You**

**Project Report**

**Talish Barmare: 662161711**

***The Problem:***

People’s music taste and preference change often. Thus, standardized music playlists are not as appealing as they might seem. It can also be tedious to create good playlists on existing applications like Apple Music & Spotify.

***Our Objective:***

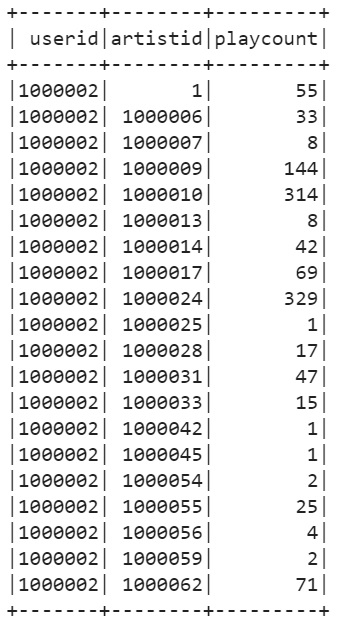
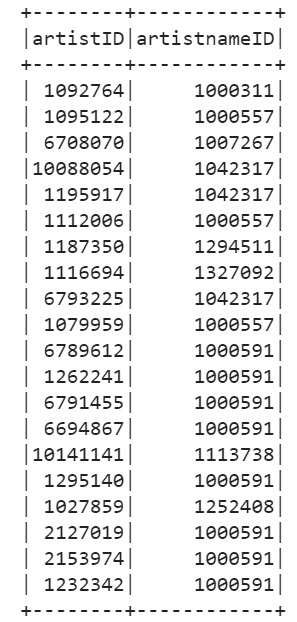
Our objective is to create a music recommendation system that will recommend a particular user new artist based on their listening history.

***Data Description:***

* To create a music recommendation system, I used a publicly available dataset from Audioscrobbler: <http://www.iro.umontreal.ca/~lisa/datasets/>
* This dataset is 484 MB in size.

This dataset contains music profiles for around 150,000 real people and 1.6 million unique artists.

* It lists the artists each person listens to, and a counter indicating how many times each user played each artist.
* There are 3 parts it consists of:
  + A **playlist dataset** which consists of a user ID, artist ID and a play count
  + An **artist dataset** which contains numeric IDs associated with artist names
  + An **artist alias dataset** which maps artist IDs that may be misspelled or nonstandard to the ID of the artist’s name

Playlist Dataset Artist Dataset Artist Alias Dataset

***Techniques:***

* Used MLlib and PySpark to build the music recommendation system using the collaborative filtering algorithm.

***Execution:***

The steps of execution taken to build the music recommendation system is as follows:

***Step 1 – Initial Step:***

* In this step, set up the Spark environment in Google Colab and load the dataset required to be used for this project.

***Step 2 – Pre Processing:***

* First obtain a list of raw artist data with each ID and name stored in tuples.
* Then use artist alias dataset to map “bad” artist IDs to “good” ones, instead of just using it as raw data pairs of artist IDs.

E.g.: Mapping ID 6803336 to 1000010, which means it maps “Aerosmith (unplugged)” to “Aerosmith.”

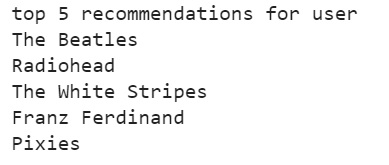
***Step 3 – Model Building:***

* Use collaborative filtering to build the model.
* The model is trained on the playlist dataset, using the Alternating Least Squares Method.
* The model runs for nearly 10 minutes.
* The output of this model is a list of artists, in order, that a particular user listens to.

***Results:***

***Applying the Model:***

For a given user id (e.g. user id: 2079004), you should get the corresponding recommendations based on the artists that the user listen to.



***Consolidating the Results:***

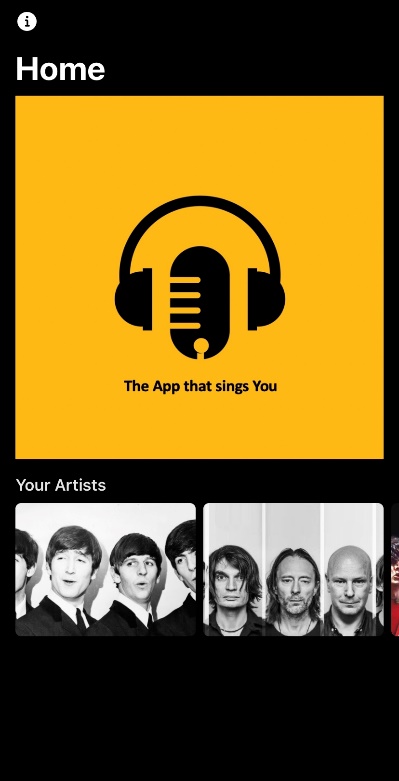
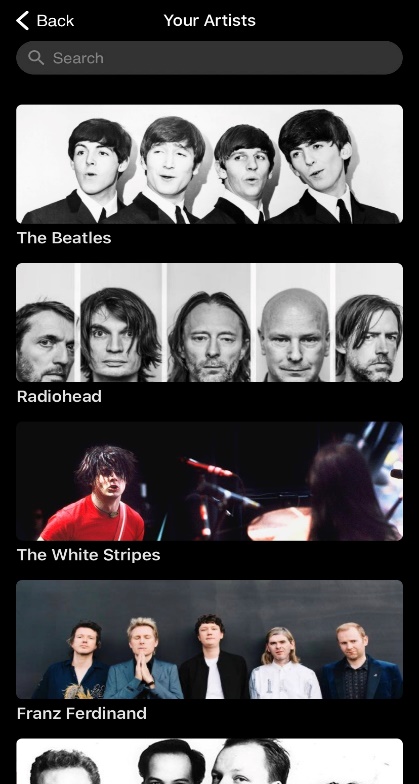
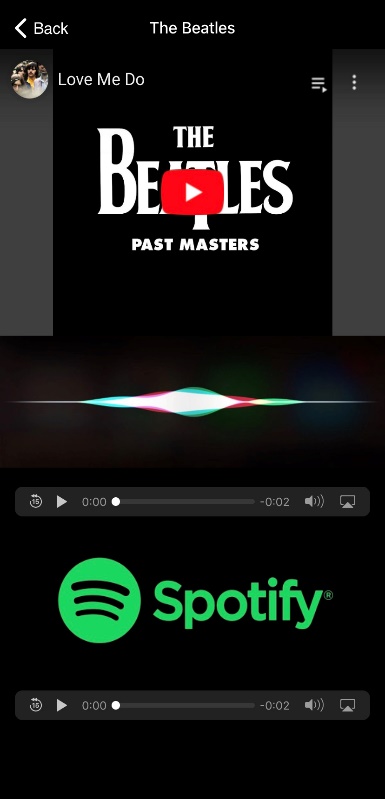
* From the recommendations obtained in the previous step, add YouTube playlists link, Siri audio and Google Assistant file links (files present in a Google drive) of the respective artist into a consolidated datasheet, which we will then import into Glide to build our mobile application (front end). With a nice web scrapping tool a list of YouTube URLs can be generated for the corresponding artists.

View the consolidated[***Datasheet link***](https://docs.google.com/spreadsheets/d/1VuW6MVvDDNX0WnmPIRyHvPzZf7VrxB19lEt1EksL9bI/edit?usp=sharing)*here*

***Datasheet snapshot***

***Front End:***

* Load the datasheet that we created earlier into the platform Glide to build the UI of our mobile application.

The app UI can be explored here: <https://u5cdo.glideapp.io> or you can scan the barcode below to download the app.

