Proyecto Final Spotify– Etapa 1

Spotify es una plataforma de streaming de música digital que ofrece acceso gratuito o de pago a sus usuarios, conectándolos con millones de canciones, videos y podcast de artista de todo el mundo. Actualmente, supera los 400 millones de usuarios registrados, este dato convierte a la plataforma como la gran referencia del sector.

De acuerdo con la solicitud realizada por parte de Spotify, a continuación, se resumirá a sus accionistas una descripción general, análisis de calidad de datos, procesos de limpieza y un análisis exploratorio de los datos de cada uno de los datasets. Esto, con el fin de que al momento de ingresar el usuario y reproducir una canción, la aplicación le recomiende un conjunto de canciones que potencialmente sean de su agrado.

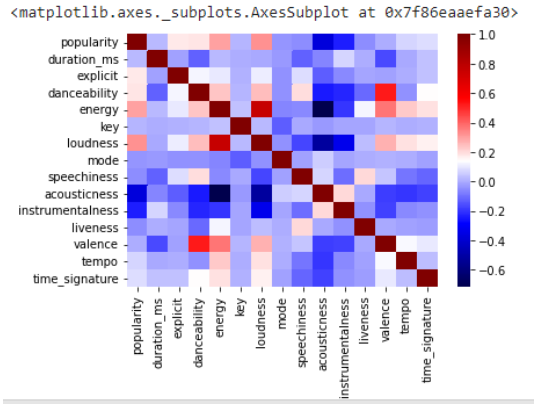
Dataset Tracks_mod

1) Descripción General:

- Dimensiones: El dataset tiene un tamaño de 586.672 filas y 20 columnas
- Tipo de datos iniciales de las variables:

```
id                 object
name               object
popularity        float64
duration_ms         int64
explicit            int64
artists            object
id_artists         object
release_date       object
danceability      float64
energy            float64
key               float64
loudness          float64
mode              float64
speechiness       float64
acousticness      float64
instrumentalness  float64
liveness          float64
valence           float64
tempo             float64
time_signature    float64
dtype: object
```

- Correlación de variables: Se realizo un gráfico de correlación para identificar la relación entre las variables y de este modo tomar aquellas que brinden mayor información para la recomendación de canciones.



- Variables relevantes en el data frame:

```
relevant
```

```
{'danceability': dtype('float64'),
 'energy': dtype('float64'),
 'loudness': dtype('float64'),
 'acousticness': dtype('float64'),
 'valence': dtype('float64')}
```

2) Análisis de calidad de datos y procesos de limpieza implementado

Con el fin de identificar las variables en las cuales hace falta información o hay campos vacíos, se utilizó la siguiente función:

```
track_df.isnull().sum()
```

La cual permite identificar el Id, duration_ms, explicit, artists, id_artist y release_date como las variables cuya información debe ser obtenida dentro de la recopilación de datos. De las variables restantes, se puede inferir que si se saltó una columna puede que no sea tan importante la información.

Se realiza una limpieza para la variable release_date YYYY-MM-DD, dado que hay datos cuyos valores contienen solo el año, la palabra year + año, año y mes. Por medio del siguiente código, se identifican cuantos datos están de forma incorrecta para la variable

```
#creacion de variable para identificar los datos que no cumplen con la condicion YYYY-MM-DD
date_malformed = track_df.loc[track_df['release_date'].apply(lambda x: (re.match('\d{4}-\d{2}-\d{2}', x) is None))]
```

```
#cantidad de datos con error en fecha
date_malformed.shape
```

```
(138594, 20)
```

Se unifican los datos de tal forma que se puedan ajustar como datatime, cambiando el tipo de dato de la variable original.

```
id                        object
name                      object
popularity               float64
duration_ms                int64
explicit                   int64
artists                   object
id_artists                object
release_date      datetime64[ns]
```

3) Análisis exploratorio de los datos

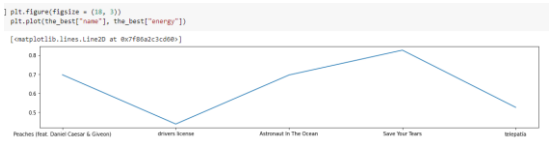
Análisis univariado: Se hace uso de la función Profiling para revisar cada una de las variables, adicionalmente se crea una lista de las variables numéricas y categóricas para poder obtener la media de los datos numéricos y graficarlos mediante histogramas. Evidenciando que la media para la duración de las canciones en minutos es de aproximadamente 3.8. Adicionalmente, se puede deducir que el artista cuya canción es la más popular es Justin Bieber.

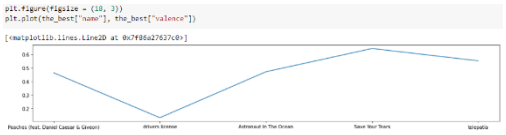
```
popularity          27.564493
duration_ms         230051.167286
explicit            0.027571
danceability        0.563608
energy              0.542160
key                 5.222037
loudness            -10.206486
mode                0.658609
speechiness         0.104757
acousticness        0.449750
instrumentalness    0.113352
liveness            0.213931
valence             0.552261
tempo               118.457192
time_signature      3.873484
dtype: float64
```

```
#Identificar cual es la cancion más popular
pop_max = track_df["popularity"].max()
track_df[track_df["popularity"] == pop_max]["name"]
```

```
93802    Peaches (feat. Daniel Caesar & Giveon)
Name: name, dtype: object
```

Análisis bivariado: Se realizo una gráfica de correlación para identificar que, entre más positividad transmita una pista más bailable será, y mayor energía tendrá. No siempre la canción con mayor energía será la más popular y la más popular no siempre será la más bailable. Para ello, se creó una variable que muestra las 5 canciones más populares y de ellas se analizó su energía y valencia. Evidenciando que, para ser de las más populares no siempre deben ser canciones extremadamente alegres, su energía en 3 de ellas es superior al 0.6 mientras que dos de ellas tienen una energía baja, en cuanto a su valencia 4 de ellas están por debajo de 0.5 es decir que no transmiten demasiada alegría. Esto permite concluir que el estigma que se tenía en cuanto a que las canciones más populares serán las más bailables no siempre es verdadero.





Dataset Artist_mod

1) Descripción General:

- Dimensiones: El dataset tiene un tamaño de 1.162.095 filas y 5 columnas
- Tipo de datos iniciales de las variables:

```
id             object
followers     float64
genres         object
name           object
popularity      int64
dtype: object
```

- Información general de los datos numéricos: La popularidad se maneja o puntúa en una escala del 0 al 100

	followers	popularity
count	1.162084e+06	1.162095e+06
mean	1.022070e+04	8.795961e+00
std	2.543995e+05	1.355777e+01
min	0.000000e+00	0.000000e+00
25%	1.000000e+01	0.000000e+00
50%	5.700000e+01	2.000000e+00
75%	4.170000e+02	1.300000e+01
max	7.890023e+07	1.000000e+02

```
artist_df['popularity'].unique()
```

```
array([  0,   8,   6,   5,   7,   3,   2,   9,  16,  19,  15,  27,  26,
        22,  34,  53,  51,  52,  56,  57,  32,  37,  41,  31,  49,  47,
        66,  59,  63,  12,  11,  40,   1,  58,  24,  23,  48,  14,  13,
        25,  18,  20,  17,  28,  30,  21,  36,  29,  35,  33,  38,  39,
        44,  42,  43,  45,  55,  46,   4,  10,  60,  65,  50,  54,  62,
        64,  61,  89,  67,  83,  72,  69,  70,  78,  68,  77,  71,  84,
        80,  76,  75,  81,  79,  73,  74,  90,  86,  85,  82,  87,  92,
        98,  96,  95,  91,  88,  93, 100,  94])
```

2) Análisis de calidad de datos y limpieza

- Se valida que en seguidores hay 11 datos nulos, sin embargo, al tener una popularidad baja este valor no afectara la conclusión de los datos, para ello se opta por filtrar los 11 registros.

```
# se valida si hay datos nulos
artist_df.isnull().sum()
```

```
id            0
followers    11
genres        0
name          0
popularity    0
dtype: int64
```

```
#mostrar el ajuste de los nulos
artist_df.isnull().sum()
```

```
id            0
followers     0
genres        0
name          0
popularity    0
dtype: int64
```

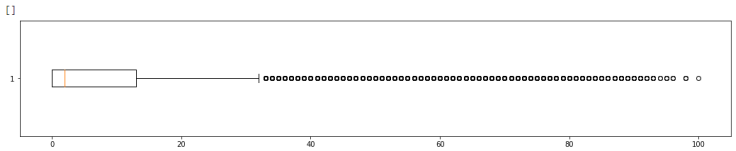
- Se decide no manipular el género, dado que ningún género representa ser mayor o menor a otro.

3) Análisis exploratorio de los datos

Análisis univariado: La popularidad entre los artistas esta entre 0 y 16, el rango de lo normal está en 36 sin embargo hay varios artistas con una puntuación superior a 40. Esto, debido a que a pesar de haber varios artistas en la plataforma no todos logran alcanzar una alta popularidad.
Cada uno de los artistas más populares pueden tener o desempeñarse en diferentes géneros musicales.

```
#Comportamiento de la popularidad
plt.figure(figsize = (18, 3))
plt.boxplot(artist_df['popularity'], vert= False)
plt.plot()
```

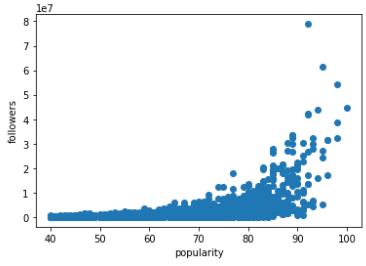
```
[]
```



Análisis bibariado: Se puede concluir que los seguidores no determinan el artista más popular, esto justificado en que Justin Bieber es el artista más popular, pero quien tiene mayores seguidores es Ed Sheeran.

```
#comportamiento de los seguidores frente a la popularidad
plt.scatter(most_popular["popularity"], most_popular["followers"])
plt.xlabel("popularity")
plt.ylabel("followers")
```

```
Text(0, 0.5, 'followers')
```



Enlace a GitHub https://github.com/Rosemary-99/Taller-1-

Proyecto Final Spotify– Etapa 2

Base de datos relacional

Inicialmente se realizaron algunos ajustes en Colab con los datasets, dado que los datos de artists y id_artists eran reconocidos como un string; cuando realmente eran una lista con varios elementos. Es decir, una canción le podía pertenecer a varios artistas generando un problema al cruzar Tracks_mod con la tabla de Artist_mod, dado que si o si se debía reconocer la cantidad de artistas para lograr cruzar la información.

Para ello, antes de exportar el track_df se realizó una nueva limpieza de datos retirando los paréntesis cuadrados [] y las comillas simples, para que dentro de la base de datos de Postgress se pudiera realizar una conversión y así los datos no fueran recibidos solo como un texto si no como un Array, guardándolo como un archivo csv con la siguiente función: track_df.**to_cs.** Lo mismo ocurrió con artis_mod, con la diferencia que en este; para géneros se dejó la información como un texto plano sin mucha modificación.

Una vez normalizados los 2 dos archivos se realizó la importación hacia la base de datos de Postgress, para ello se utilizó Jupyter como bloc de notas para realizar la conexión, se adjunta código por cada una de las tablas (Anexo 1-import_songs.py y Anexo 2 import_artist.py).

Para ejecutar correctamente los códigos, en cada una de las importaciones es necesario que quien quiera correr el programa cambie en la conexión el password y la ruta en read_csv, de este modo al momento de comprobar su funcionamiento lo podrá realizar desde su equipo. De igual forma, para no generar un error en la instalación se debe ejecutar en el CMD de Windows lo siguiente:

```
pip install pandas
pip install psycopg2-binary
pip install SQLAlchemy
```

Una vez importados los datos a Postgress, fue necesario crear una nueva columna en la tabla de songs (tracks_mod) dado que al importar la columna artist y id_artist esta no era tomada como un arreglo; para ello se utilizó el siguiente código:

```
ALERT TABLE songs ADD COLUMN artist1 TEXT [];
UPDATE songs SET artist1 = string_to_array(artists, ',');

ALERT TABLE songs ADD COLUMN id_artist1 TEXT [];
UPDATE songs SET id_artist1 = string_to_array(id_artists, ',');
```

Esta función permitió encontrar la llave que conectaría las tablas ver anexo 3.

A continuación, se mostrará el modelo relacional implementado para la conexión entre las tablas, identificando que efectivamente id_artist y Id contenían la misma información facilitando el cruce entre las tablas.

Tracks = songs

```
id
name
popularity
duration_ms
explicit
artists
id_artists
release_date
danceability
energy
key
loudness
mode
speechiness
acousticness
instrumentalness
liveness
valence
tempo
time_signature
dtype: object
```

Artist

```
id
followers
genres
name
popularity
dtype: int64
```

Obteniendo como resultado las siguientes tres tablas:
Tabla 1. Representa la relación entre el nombre del artista y las canciones más populares.

	Nombre de la Canción text	Popularidad de la Canción double precision	Nombre del artista character varying
1	Peaches (feat. Daniel Caesar & Giveon)	100	Justin Bieber
2	drivers license	99	Olivia Rodrigo
3	Astronaut In The Ocean	98	Masked Wolf
4	Save Your Tears	97	The Weeknd
5	telepatía	97	Kali Uchis

Tabla 2. Representa el genero del artista que tiene la canción más popular.

	Nombre de la Canción text	Popularidad de la Canción double precision	Género del artista character varying
1	Peaches (feat. Daniel Caesar & Giveon)	100	['canadian pop', 'pop', 'post-teen pop']

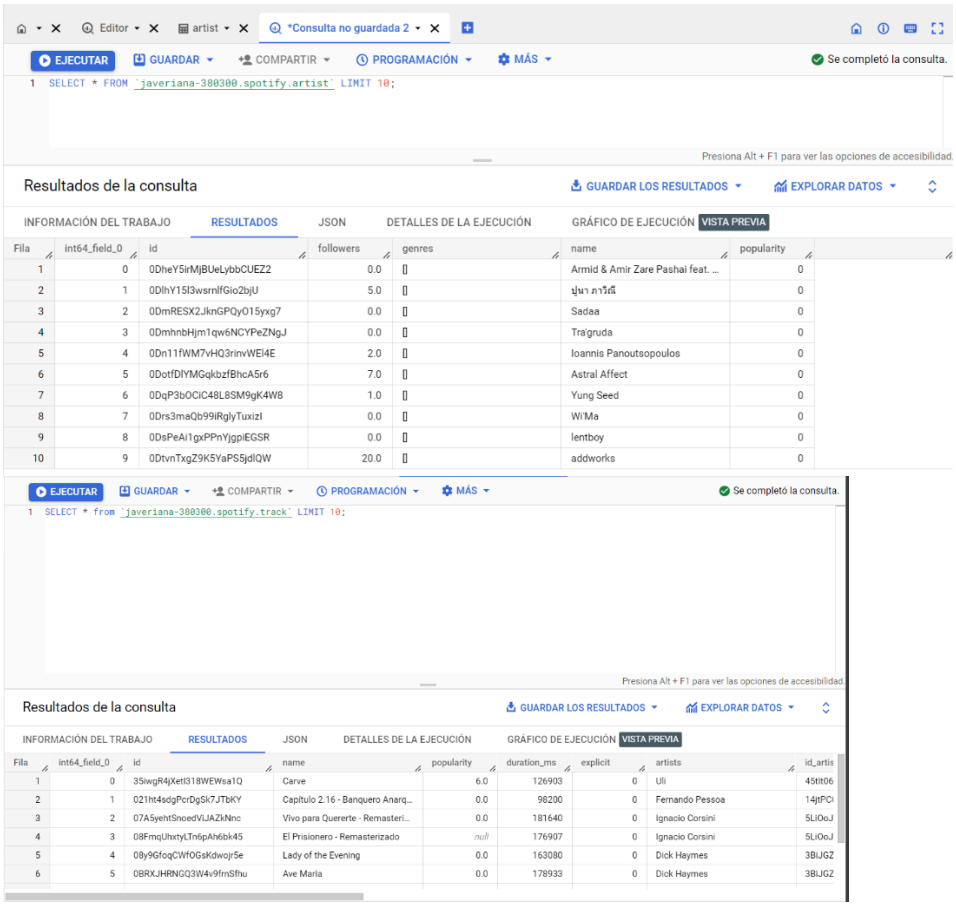
Tabla 3. Representa la información de las canciones con el artista con mayores seguidores.

Data Output | Messages | Graph Visualiser | Notifications

	Nombre de la Canción text	Popularidad de la Canción double precision	Seguidores del artista double precision
26	Dive	75	78900234
27	Perfect Duet (Ed Sheeran & Beyoncé)	75	78900234
28	I Don't Care (with Justin Bieber)	83	78900234
29	Beautiful People (feat. Khalid)	81	78900234
30	South of the Border (feat. Camila Cabello & Cardi B)	80	78900234

Bodega de Datos

La bodega de datos se realizó en BigQuery, allí se almacenaron los datasets tracks_mod y artists_mod lo más preparados posibles para el futuro entrenamiento del modelo de recomendación.



Finalmente, el proceso de ETL se realizó de la siguiente forma:

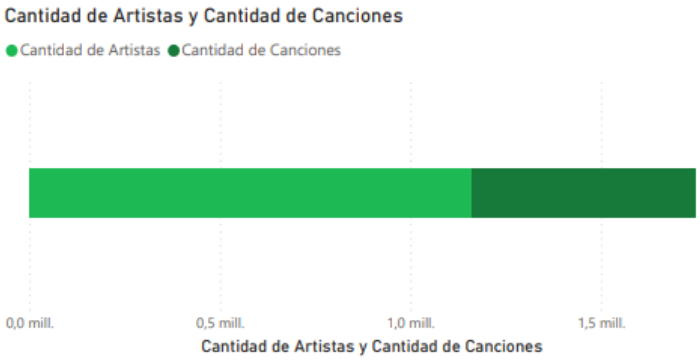
Enlace a GitHub https://github.com/Rosemary-99/Taller-1-

Proyecto Final Spotify– Etapa 3

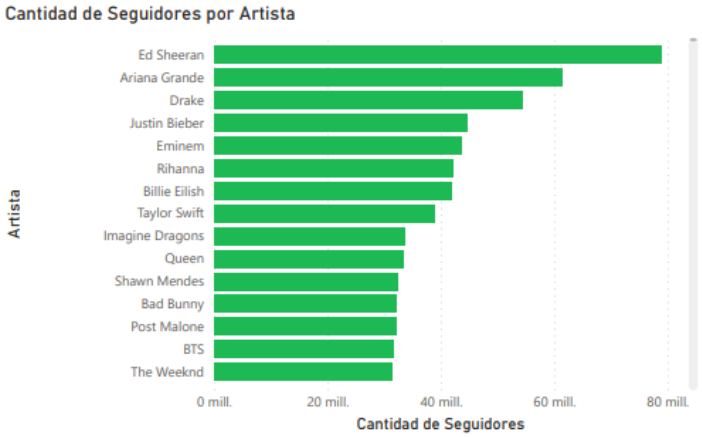
Con los datos de canciones y artistas limpios y almacenados en PostgreSQL y BigQuery, se construyeron dos productos de datos con las siguientes características:

1) Un dashboard en powerBI el cual le permite a la unidad de negocio conocer el estado actual de la plataforma en términos de:

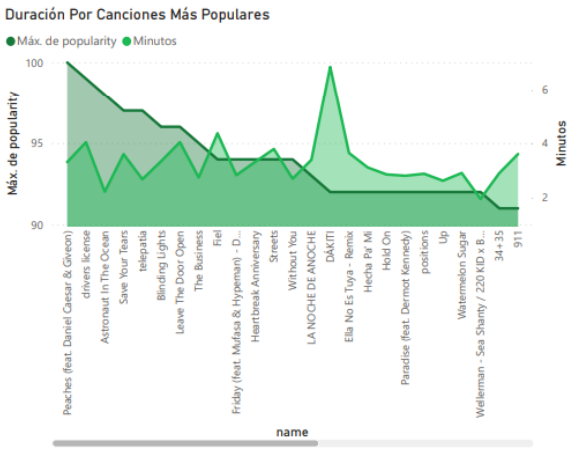
- Cantidad de artistas y cantidad de canciones, de la cual se puede concluir que a un artista le pueden pertenecer varias canciones y pueden realizar colaboraciones entre ellos, adicionalmente existen más de 1.0 mill de artistas en esta plataforma.



- Cantidad de seguidores por artistas, allí se reflejan los artistas con mayor cantidad de seguidores en la plataforma, lo cual pude motivar a los accionistas a dirigir sus comerciales para aquellos que no pagan plan premium una vez escuchan uno de los artistas más seguidos, ya que su raiting es mayor y pueden cobrar por una excelente publicidad en base a esta información.



- Duración por canciones más populares, esta información es demasiado útil para los artistas, ya que pueden ver la relación entre duración y canciones Top, con el fin de orientar sus creaciones a un tiempo estándar en el cual cautiven a la audiencia para incrementar su popularidad y seguidores.



Adicionalmente, se crearon indicadores clave los cuales permiten medir:

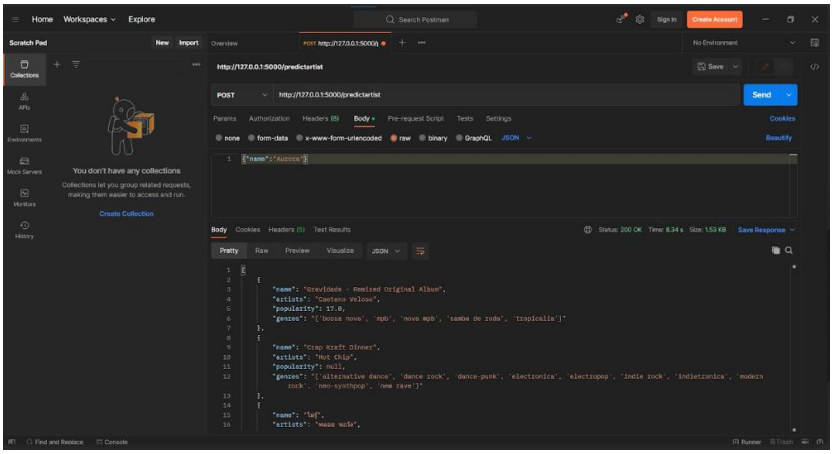
- Porcentaje de artistas con popularidad superior a 90, con este indicador se puede medir la importancia de mantener dentro de la plataforma a los artista más importantes, ya que gracias a ellos se unen cada vez más usuarios en la plataforma, esto también permite a los accionista corroborar o plantearse un objetivo de negocio que permita la continuidad de la plataforma, ya que al medir a los artistas más populares pueden garantizar que dentro de esta están los artistas de moda y que efectivamente disponen en Spotify a los mejores iconos de la música. Mientras estén los artistas más populares se garantiza la permanencia de la audiencia dentro de la plataforma.

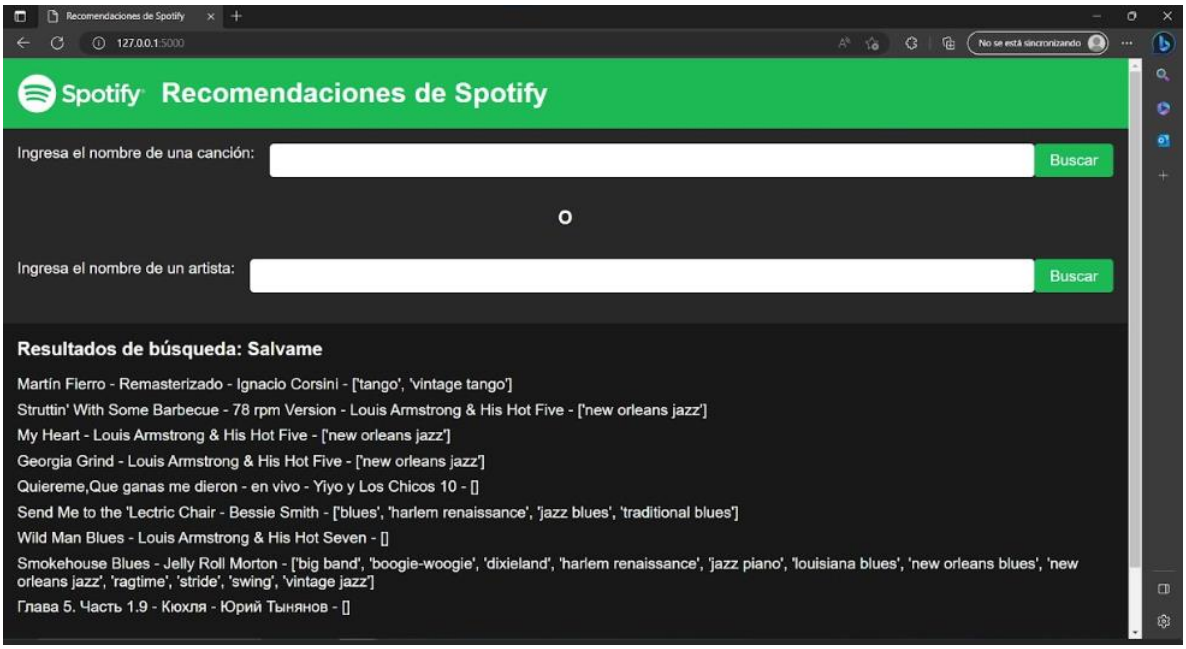
- Porcentaje de las canciones con popularidad mayor a 90, este indicador permite medir la calidad y la aceptación de los usuarios con respecto al contenido disponible en la plataforma de diferentes artistas, con esta información se podrían censurar aquellos contenidos que tal vez no generen una buena imagen dentro de la plataforma, garantizando a los usuarios un excelente servicio.

De este modo y con toda la información recopilada dentro de las 3 etapas, se construyó un simulador de recomendación de canciones para la plataforma, en la cual al reproducirse una canción se le recomienda al usuario de acuerdo con variables como canción, nombre del artista y género un listado de 10 canciones, personalizando el servicio para lograr ser la plataforma número uno de servicios de multimedia a nivel mundial.

Para ello se utilizo el framework Flask el cual permitió la creación de la aplicación web, inicialmente se utilizo la consola de visual studio, sin embargo, en ella no fue posible descargar la librería annoy, por tal motivo se selecciono pycharm en el cual se pudo ejecutar adecuadamente el código, el cual se conecto con los datasets ajustados de los pasos anteriores.

De igual forma, por medio de postman se realizaron pruebas para validar el funcionamiento del código, allí se presentaron diferentes inconvenientes, pero por medio de prueba y error se logro llegar al mejor modelo. El cual se puede evidenciar a continuación y con el anexo de los códigos implementados en pycharm quien desee lo podrá ejecutar.





Enlace a GitHub https://github.com/Rosemary-99/Taller-1-