A

Project Report on

**Spotify Data Analysis**

*Submitted in Partial Fulfilment of the requirement for the Award of Degree of*

**Bachelor of Computer Application**

Session 2023-24

*Submitted to*

**Er. Vinay Kumar Vishwas**

**(Assistant Professor)**

*Submitted by*

**ANANYA SHUKLA**

**(202110101130019)**

*Under the guidance of*

**Mr Amit Kumar Singh**



Department of Computer Science and Information System

**Shri Ramswaroop Memorial University Barabanki, Uttar Pradesh**

**ACKNOWLEDGEMENT**

I would like to express my profound to **Dr. Bineet Kumar Gupta** , Head of Department of Computer Science and information Systems and to my faculty **Er. Vinay Kumar Vishwas** who gave me this opportunity to work on the project Spotify Data Analysis

I would like to express my special thanks to my mentor **Mr Amit Kumar Singh** who gave me useful advice and suggestions which were really helpful to me during this project’s completion.

Also, I would like to thank my teammate Namita Singh and her effort towards this project.

I would like to acknowledge that this project was completed entirely by me and my teammates not by someone else.

Name- Namita Singh Name- Ananya Shukla

Roll No.- 202110101130020 Roll No.-202110101130019

Course- BCA Course-BCA

Group- 51 Group-51

**DECLARATION**

This is to certify that the project report entitled **"Spotify Data Analysis"**, which is submitted by me in the partial fulfilment for the award of the degree of Bachelor of computer application, Shri Ramswaroop Memorial University Barabanki, (Uttar Pradesh) comprises the original work carried out by me.

I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full for the award of any other degree or diploma in this institute or any other institute or university.

**Place: Name of Student:**

Barabanki (Uttar Pradesh) Ananya Shukla

**CERTIFICATE**

This is to certify that the report entitled Spotify Data Analysis being submitted by Ananya Shukla towards partial fulfilment for the award of the Degree of Bachelor of Computer Application from the Department of Computer Science and Information Systems to SHRI RAMSWAROOP MEMORIAL UNIVERSITY is a record of benefited work carried out by them under my guidance and supervision.

The result embodied in this project report have not been submitted to any other University or Institute for the award of any Degree or Diploma.

**Dr. Bineet Kumar Gupta Er. Vinay Kumar Viswas Mr Amit Kumar Singh**

(Head of the Department) (Project Coordinator) (Supervisor)

Department of Computer Department of Computer Department of Computer

Science and Information Science and Information Science and Information

Systems. Systems. Systems.

Shri Ramswaroop Memorial Shri Ramswaroop Memorial Shri Ramswaroop Memorial

University. University. University.

# TABLE OF CONTENT

|  |  |  |
| --- | --- | --- |
| **S. NO.** | **CONTENT** | **PAGE NO.** |
| 1 | Introduction | 1 |
| 2 | Objective | 2 |
| 3 | Features | 3 |
| 4 | Scope | 4 |
| 5 | Purpose | 5 |
| 6 | Requirement Analysis | 6 |
| 7 | Software Requirement | 7 |
| 8 | Hardware Requirement | 8 |
| 9 | Project Team | 9 |
| 10 | Module Description | 10 |
| 11 | Data Flow Diagram | 11 |
| 12 | Pictures Section | 12-31 |
| 13 | Coding Section | 31-40 |
| 14 | Conclusion | 41 |
| 15 | Reference | 42 |

1. **INTRODUCTION**

The "Spotify Data Analysis " represents a sophisticated solution designed to navigate and analyze the extensive Spotify dataset seamlessly. In contrast to conventional manual data exploration methods, this application serves to eliminate the challenges associated with extracting insights from the Spotify music data. The software is crafted with precision to optimize operations, reduce errors, and enhance the overall effectiveness of Spotify data analysis.

The application provides users with an intuitive interface to explore various facets of the Spotify dataset, including genre distributions, artist trends, track details, and popularity dynamics. Leveraging the capabilities of data visualization libraries like Matplotlib, Seaborn, and Plotly, the app delivers a user-friendly experience for navigating the complexities of Spotify's rich musical landscape.

It is tailored for users seeking a comprehensive and user-friendly tool to explore and interpret Spotify music data. By providing a range of analytical options and visualizations, the application empowers both casual users and data enthusiasts to uncover patterns and trends within the diverse world of Spotify music. It is positioned as a valuable resource for those interested in gaining deeper insights into Spotify's extensive music collection. With its interactive features and user-friendly design, the application offers a powerful platform for exploring, understanding, and appreciating the diverse attributes of the Spotify dataset.

# OBJECTIVE

The core objective of the "Spotify Data Analysis " is to offer users a user-friendly platform for exploring and analyzing Spotify's extensive music dataset. Through clear visualizations, the application enables users to delve into key numerical columns, such as popularity and danceability. It further facilitates in-depth genre analysis, artist-driven exploration, and detailed insights into individual tracks, including popularity trends over time. The project aims to streamline data analysis, empowering users to uncover patterns and trends within the rich landscape of Spotify's music collection. The dynamic visualizations and user-friendly interface make this app a valuable resource for anyone seeking meaningful insights into Spotify data.

# FEATURES

* User-friendly interface for seamless navigation and accessibility.
* Interactive visualizations, such as histograms, facilitating analysis of numerical columns like popularity and danceability.
* In-depth exploration of music genres, including distribution, top genres based on features, and visualizations of average attributes.
* Artist-centric analysis, enabling users to search for specific artists and access detailed information about associated tracks.
* Comprehensive track analysis, offering insights into attributes like danceability, energy, and valence, along with temporal popularity trends.
* Thorough popularity analysis, featuring descriptive statistics, popularity distribution histograms, and tables displaying top and bottom tracks.
* Dynamic genre graphs, providing visual representations of average popularity, energy, and danceability by genre.
* 3D scatter plots for an immersive exploration of complex relationships within the dataset.
* Remote access capabilities, allowing users, including busy executives, to manage and explore Spotify data at their convenience.
* Designed to cater to users with diverse levels of expertise in data analysis, ensuring a universally accessible and enriching experience

1. **SCOPE**

* Holistic Data Exploration: Uncover patterns and trends within the extensive Spotify dataset.
* Genre-Specific Insights: Gain a deeper understanding of different music genres and their unique attributes.ruko

* Track Details Exploration: Delve into detailed information about specific tracks, including key attributes.
* Popularity Dynamics Analysis: Thoroughly analyze track popularity and identify trends over time.
* Remote Accessibility: Manage and explore Spotify data conveniently with remote access capabilities.
* Visual Data Representation: Utilize dynamic visualizations for effective representation of complex data relationships.
* User-Friendly Interface: Ensure accessibility for users with varying levels of expertise for a seamless experience.
* Error-Free Operation: Implement error checks for smooth app operation and minimal data exploration errors.

# PURPOSE

The purpose of the "Spotify Data Analysis " is multifaceted, aiming to provide users with a comprehensive platform for exploring and understanding the extensive Spotify music dataset. Through various analyses, the app seeks to unearth valuable insights into different facets of the music data, including genres, artist contributions, track details, and popularity dynamics. The primary goal is to empower users with informed decision-making capabilities, whether for personal enjoyment, strategic planning, or business considerations. The user-friendly interface ensures accessibility for a broad audience, fostering an enjoyable and seamless data exploration experience. Additionally, the app serves as an educational resource, promoting data literacy by presenting complex data relationships through dynamic visualizations. With remote accessibility features, including error prevention measures, the app caters to users' diverse needs, allowing them to conveniently manage and explore Spotify data. Ultimately, the purpose is to enrich the user experience, making the exploration, analysis, and interpretation of Spotify's music collection both powerful and accessible.

# REQUIREMENT ANALYSIS

The "Spotify Data Analysis " is designed to load and analyze data from the Spotify dataset. It offers functionalities such as data display, distribution analysis, genre insights, artist and track exploration, and popularity trends. The app prioritizes user friendliness and customization, allowing users to interact with the data in an intuitive manner. Non-functional requirements include performance efficiency, accessibility, and data security, ensuring a seamless and secure user experience. Overall, the app aims to provide accessible and insightful music data analysis to a diverse user base, from music enthusiasts to data analysts, with a focus on user interaction and data exploration.

# 

**The proposed system has the following requirements:**

* The app should offer a user-friendly interface for easy navigation and exploration of the Spotify dataset.
* Implement numerical column analysis with interactive visualizations, covering features like popularity, danceability, and energy.
* Provide genre analysis features, including distribution, top genres, and visualizations of average attributes.
* Support artist name analysis, enabling users to search for artists and retrieve detailed information about associated tracks.
* Enable track analysis, offering insights into attributes like danceability, energy, and popularity trends over time.
* Ensure remote access capabilities, allowing users, including busy executives, to conveniently manage and explore Spotify data.

# SOFTWARE REQUIREMENT

OPERATING SYSTEM Windows 10, 11, Linux, MacOS

LANGUAGES Python - Version 3.7 or higher

DATA ANALYSIS LIBRARIES Pandas - Version 1.3.3 or higher

Matplotlib - Version 3.4.3 or higher

Seaborn - Version 0.11.2 or higher

Plotly - Version 5.3.1 or higher

Streamlit - Version 1.10.0 or higher

BROWSER Any browser, i.e. chrome -Version 4.0, 3.2

IDE Visual Studio Code - Version 1.83

PACKAGE MANAGEMENT pip - Version 21.2.4 or higher

# HARDWARE REQUIREMENT

PROCESSOR Intel i5 or above.

RAM 1 GB

HARD DISK 20 GB

MONITOR 15 inch Colour Monitor

# PROJECT TEAM

1. NAMITA SINGH (ROLL NO. 202110101130020, BCA 51)
2. ANANYA SHUKLA (ROLL NO. 202110101130019, BCA 51)

1. **MODULE DISCRIPTION**

* Data Loading Module: Handles loading Spotify dataset from CSV using Pandas.
* User Interface Module: Utilizes Streamlit for an intuitive, multi-column UI.
* Numerical Column Analysis Module: Generates histograms for selected numerical columns.
* Genre Analysis Module: Explores genre distribution and analyzes average popularity, danceability, and energy.
* Artist Name Analysis Module: Allows user to search for an artist, displaying associated tracks.
* Track Analysis Module: Provides detailed information on individual tracks, including visualizations.
* Popularity Analysis Module: Offers descriptive statistics and visualizations for track popularity.
* Genre Graphs and 3D Scatter Plots Module: Generates genre-based graphs and 3D scatter plots.
* Analysis Options Module: Presents users with options like Genre Analysis, Artist Name Analysis, etc.
* Modular Structure Module: Organizes code into functions for maintainability and scalability.

# DATA FLOW DIAGRAM

**Streamlit App**

**- User Interaction**

**- Data Analysis**

**- Analysis Results**

**- Graphical Output**

**Data Stores**

**- Loaded Data (CSV)**

**- Analysis Results**

**Data Sources**

**- Spotify Dataset**

**Result Visualizations**

**and Analysis Details**

**Genre Graphs, 3D**

**Scatter Plots**

**- Genre Analysis**

**- Artist Name Analysis**

**- Track Analysis**

**- Popularity Analysis |**

**- 3D Scatter Plots**

**Data Processing and Cleaning**

**- Data Transformation**

**- Statistical Analysis**

**- Graph Generation**

**Data Stores**

**- Genre Analysis Data**

**- Artist Analysis Data**

**- Track Analysis Data**

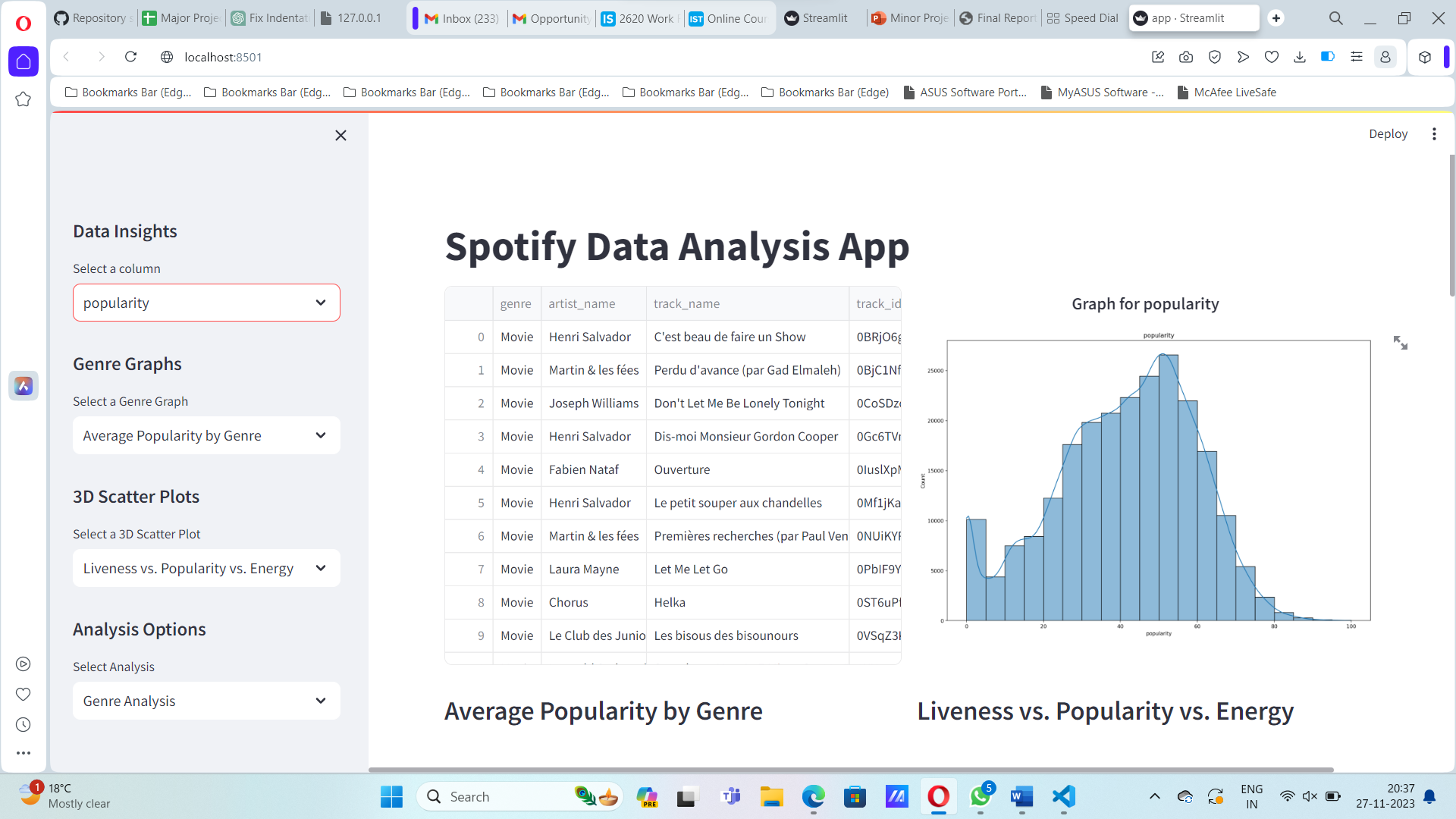
**-**

**Data Stores**

**- Processed Data**

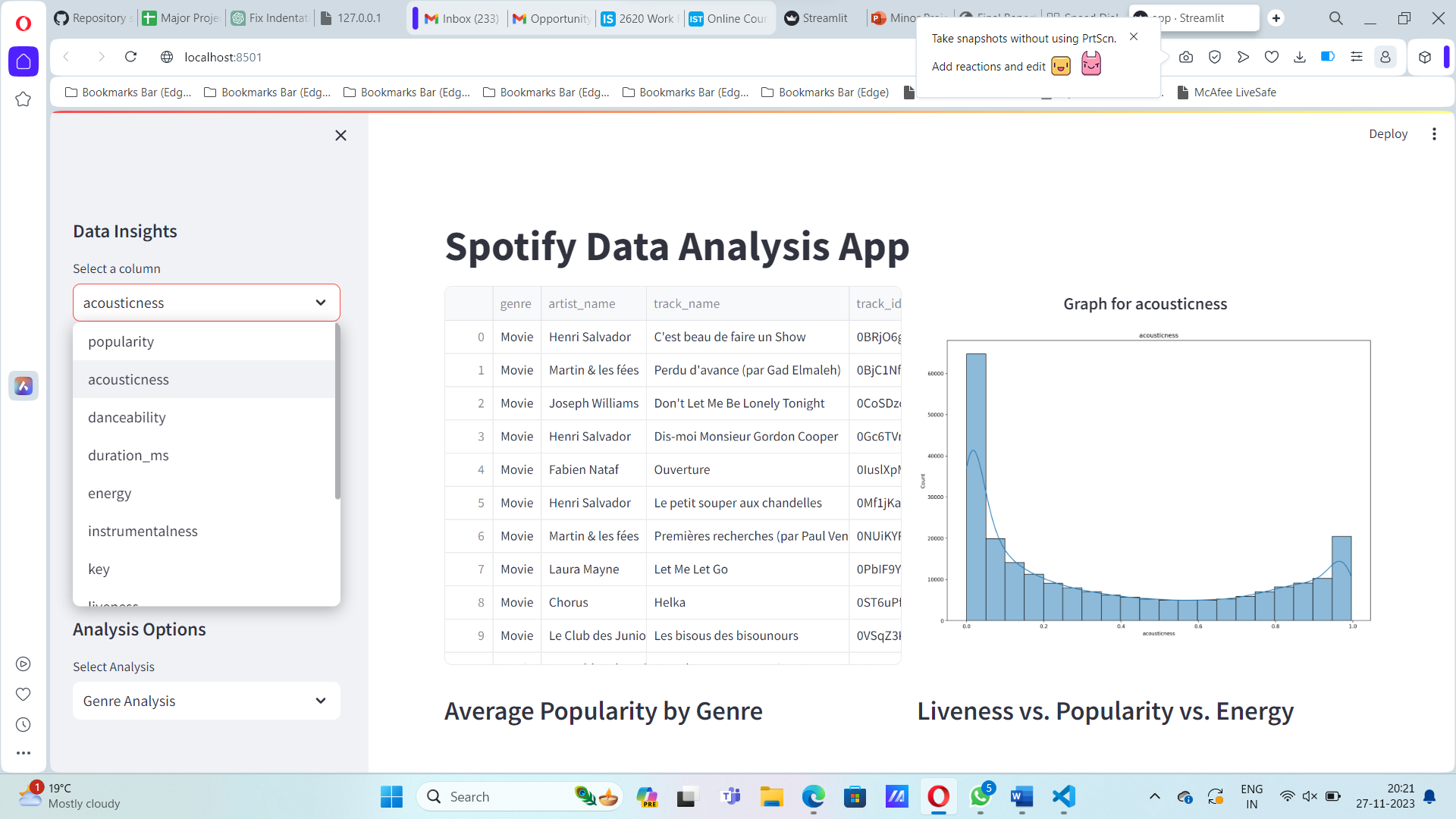
**- Analysis Results**

# Pictures Section

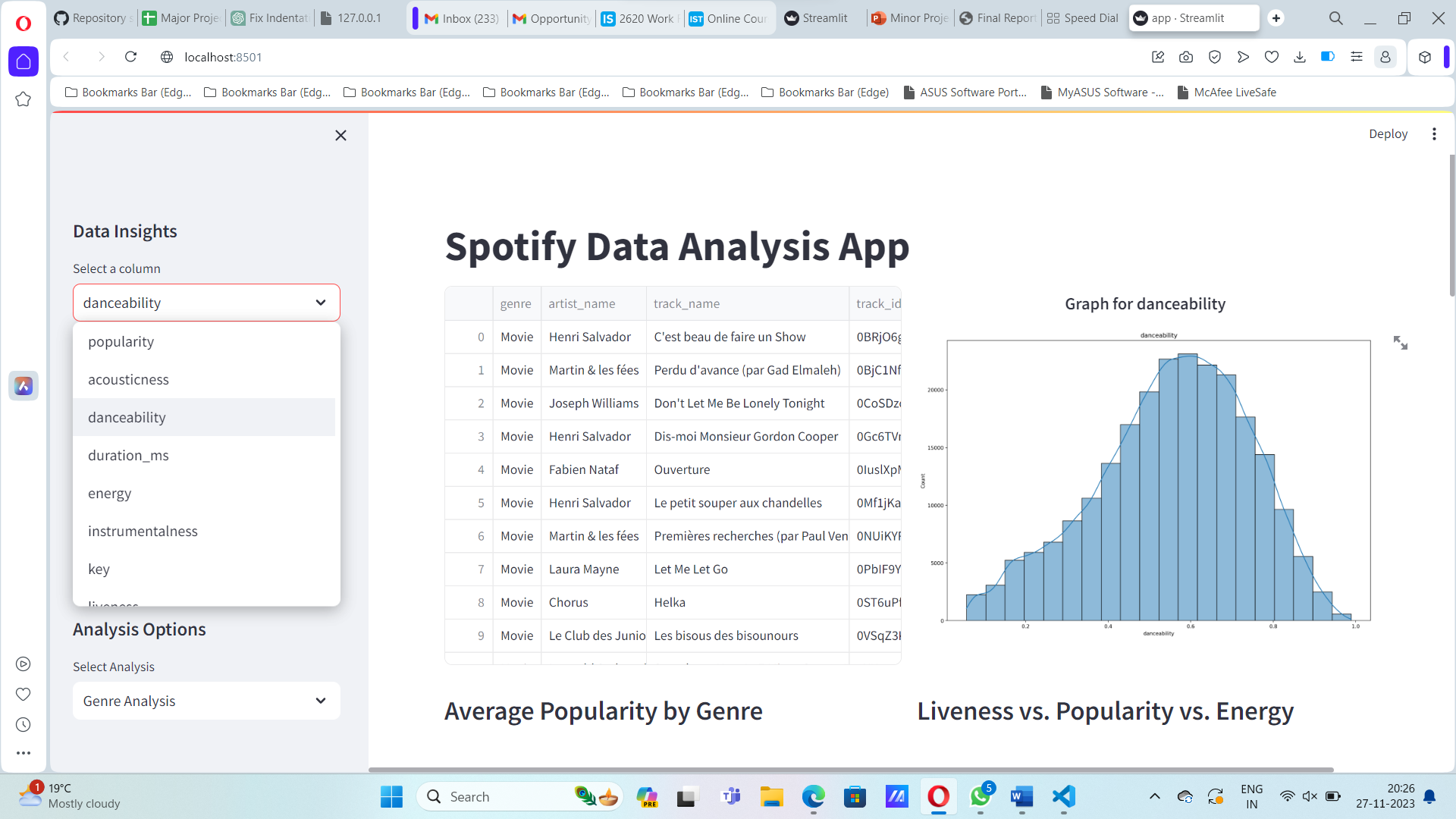


**Front Page**

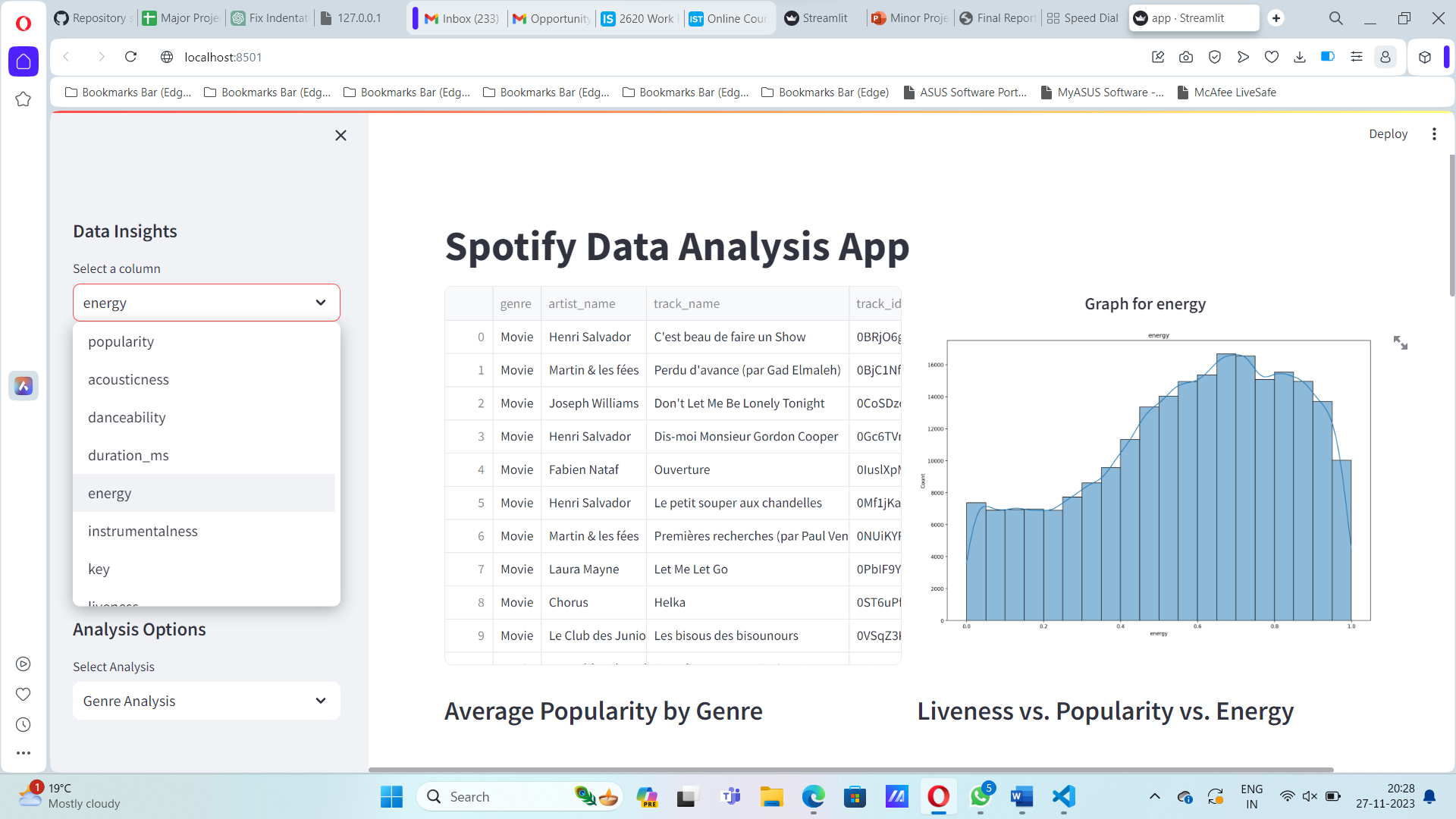




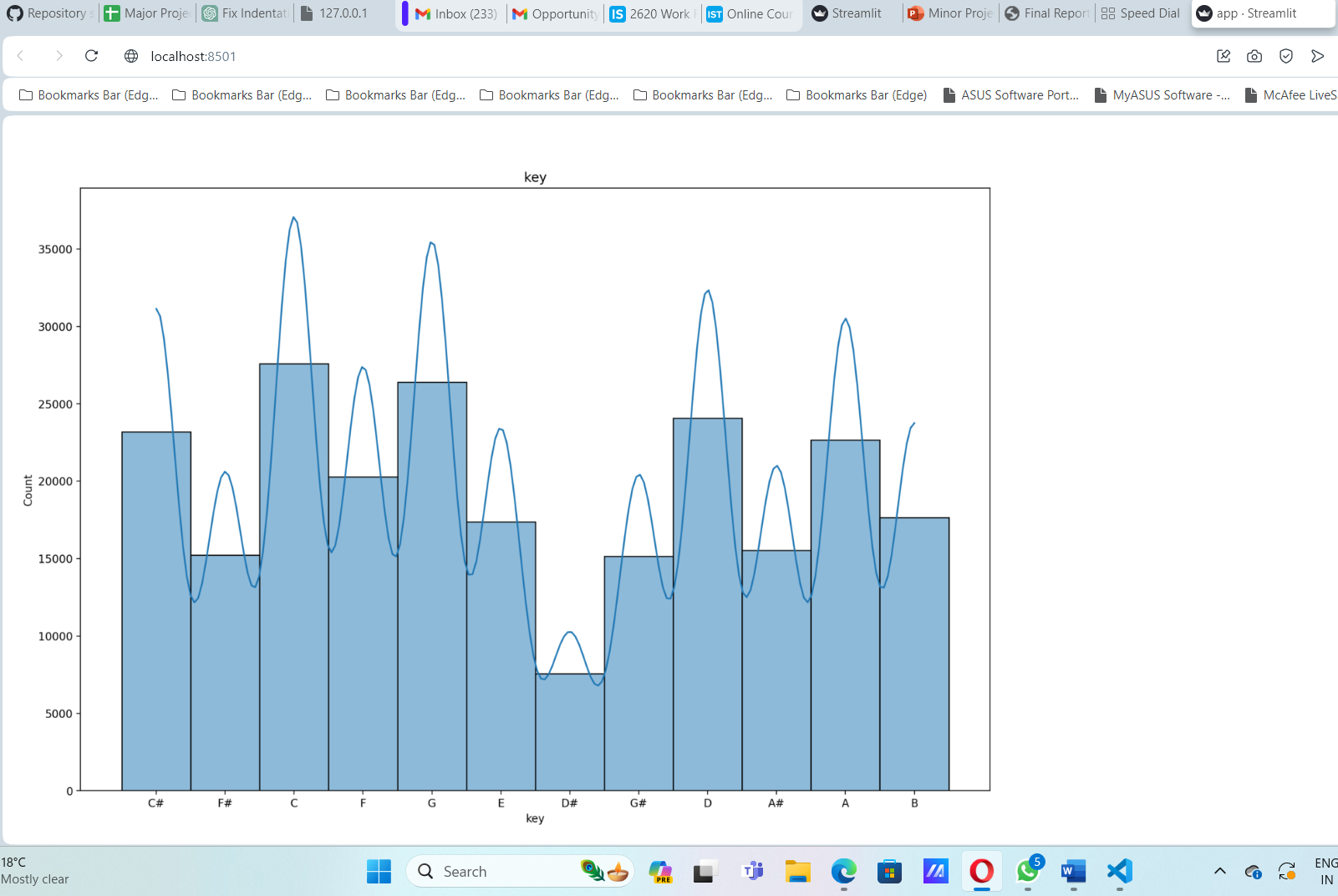
**Acousticness graph**

****

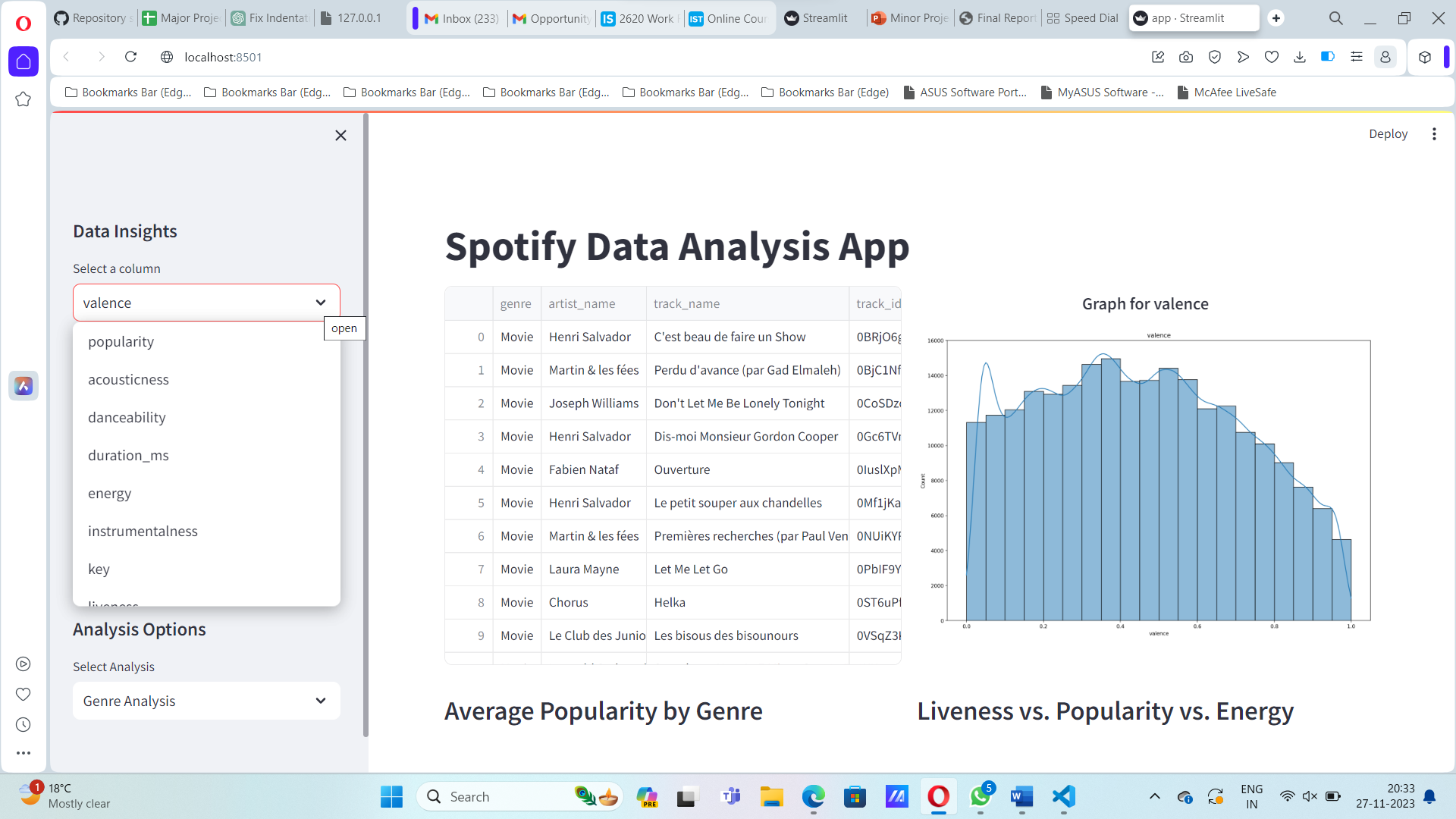
**Danceability Graph**

****

**Energy Graph**

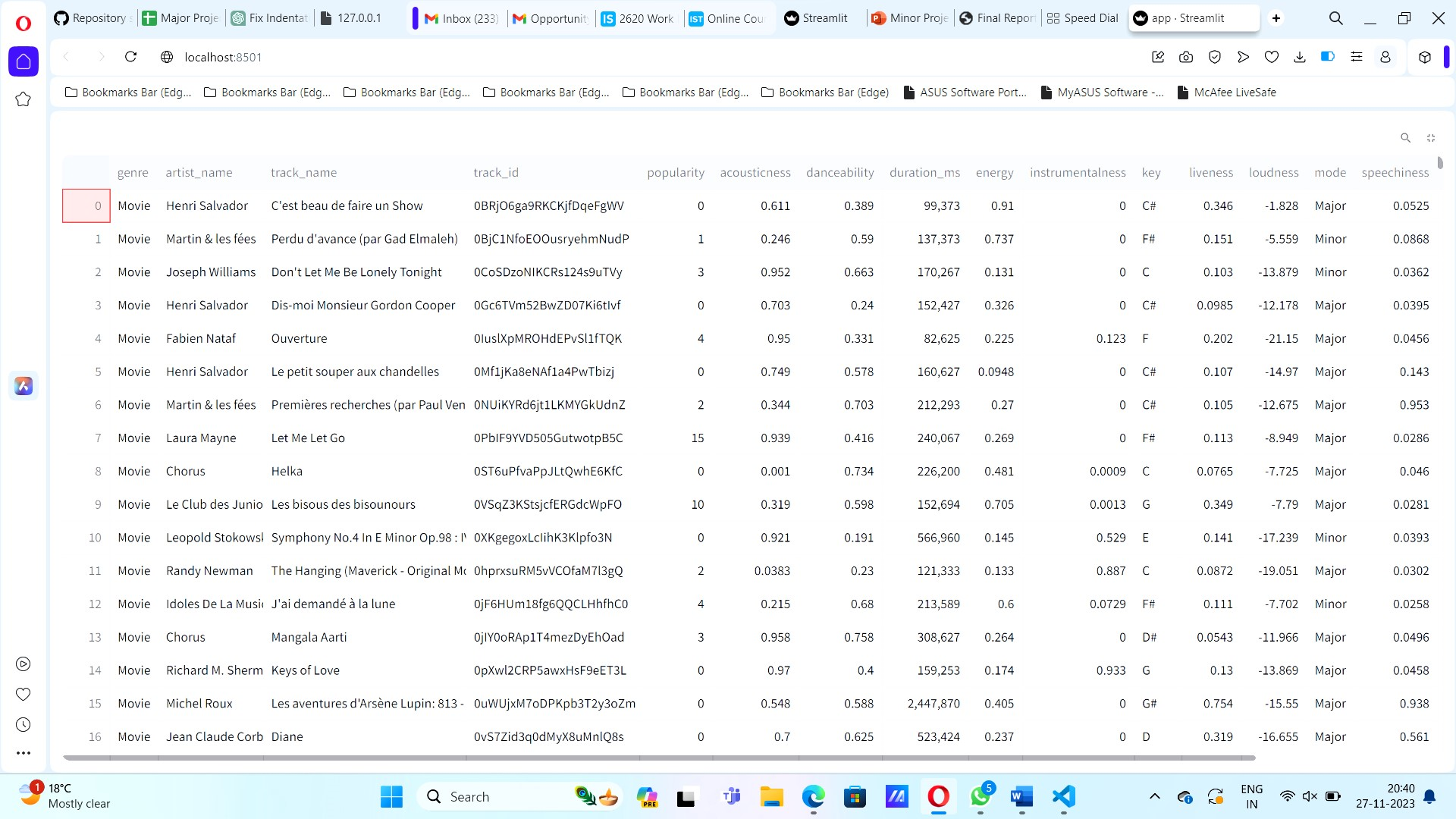
****

**Key Graph**

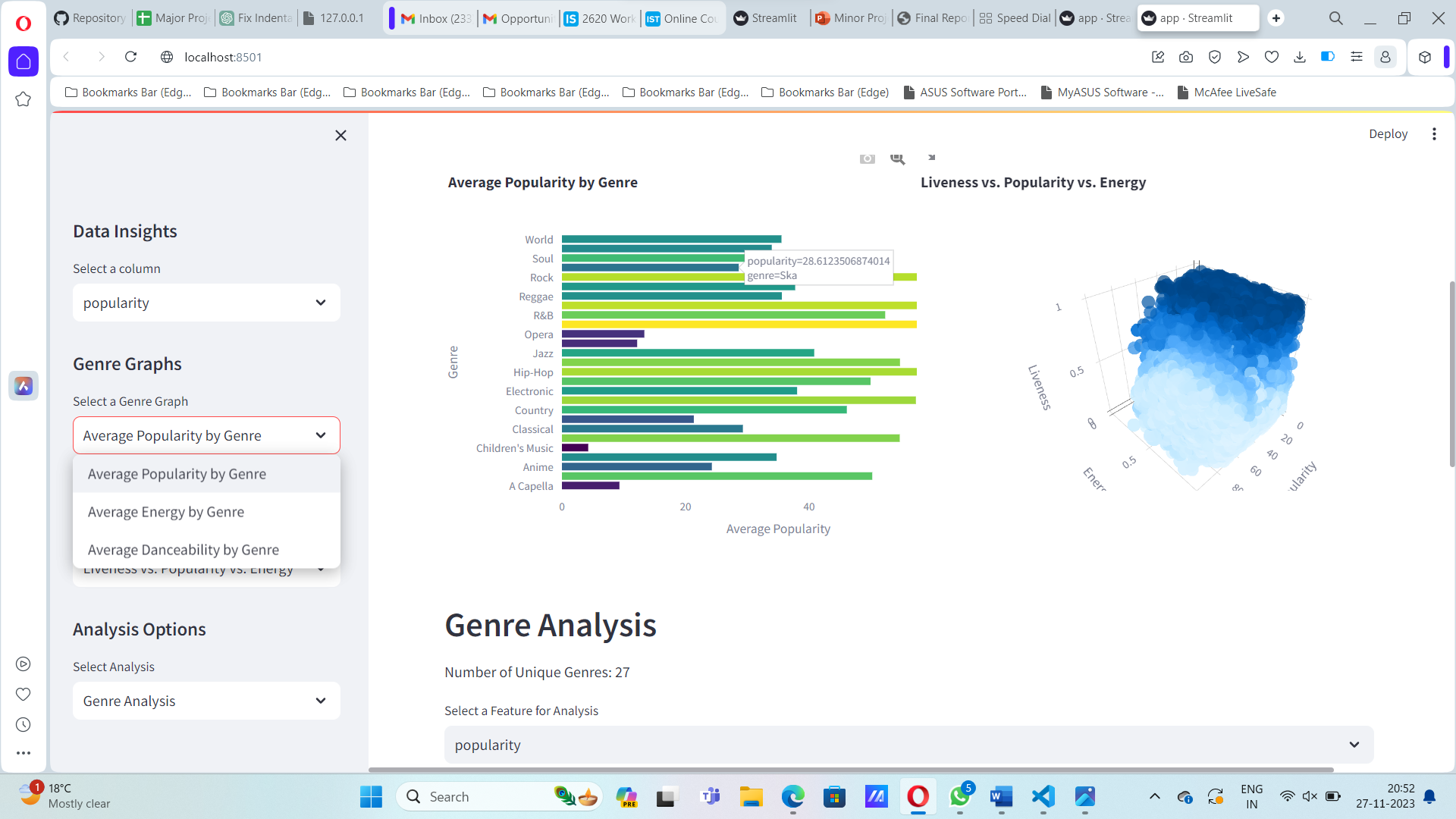
****

**Valence Graph**

**And many more graphs available insights data**

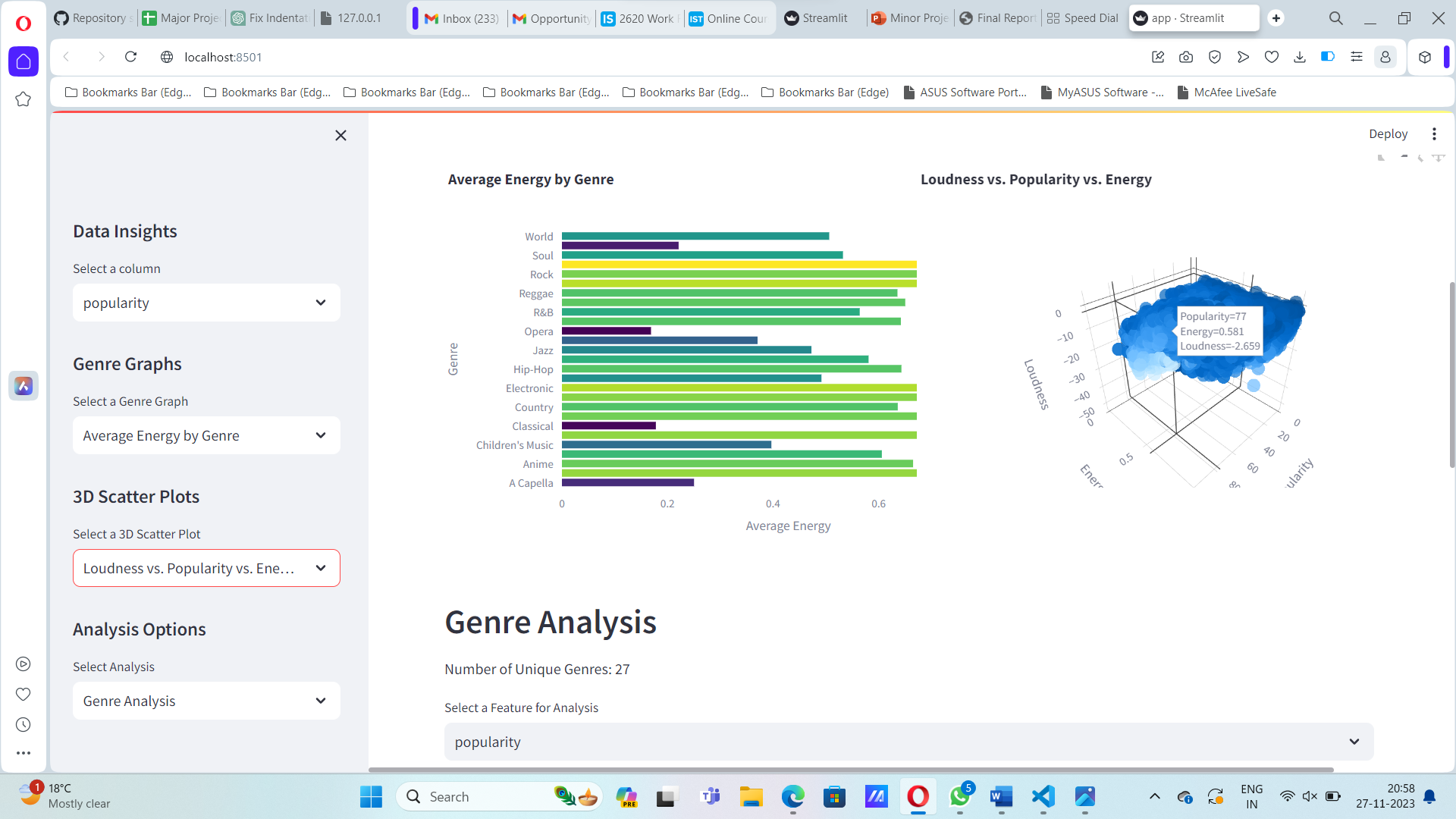


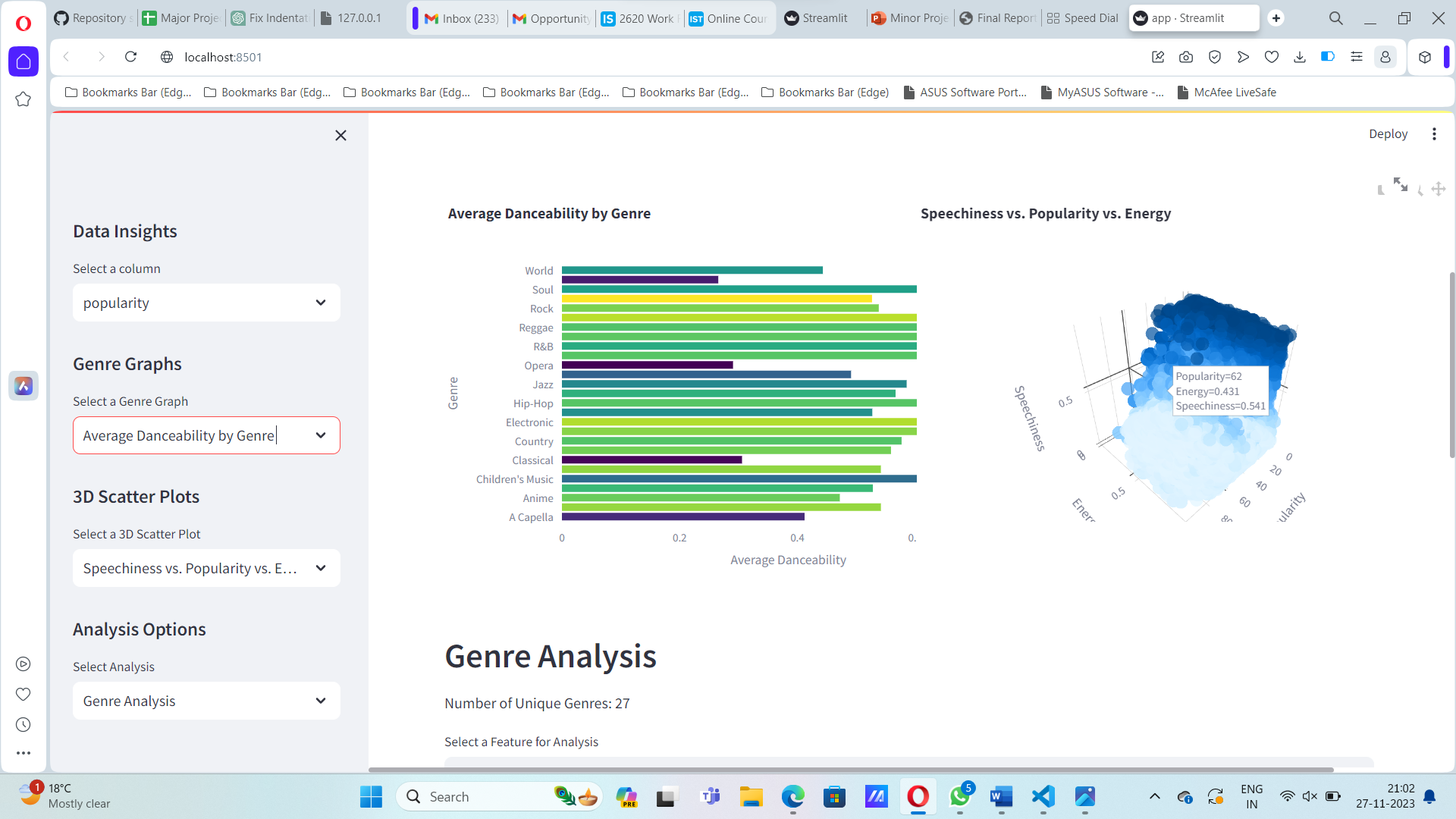
**Available Data**

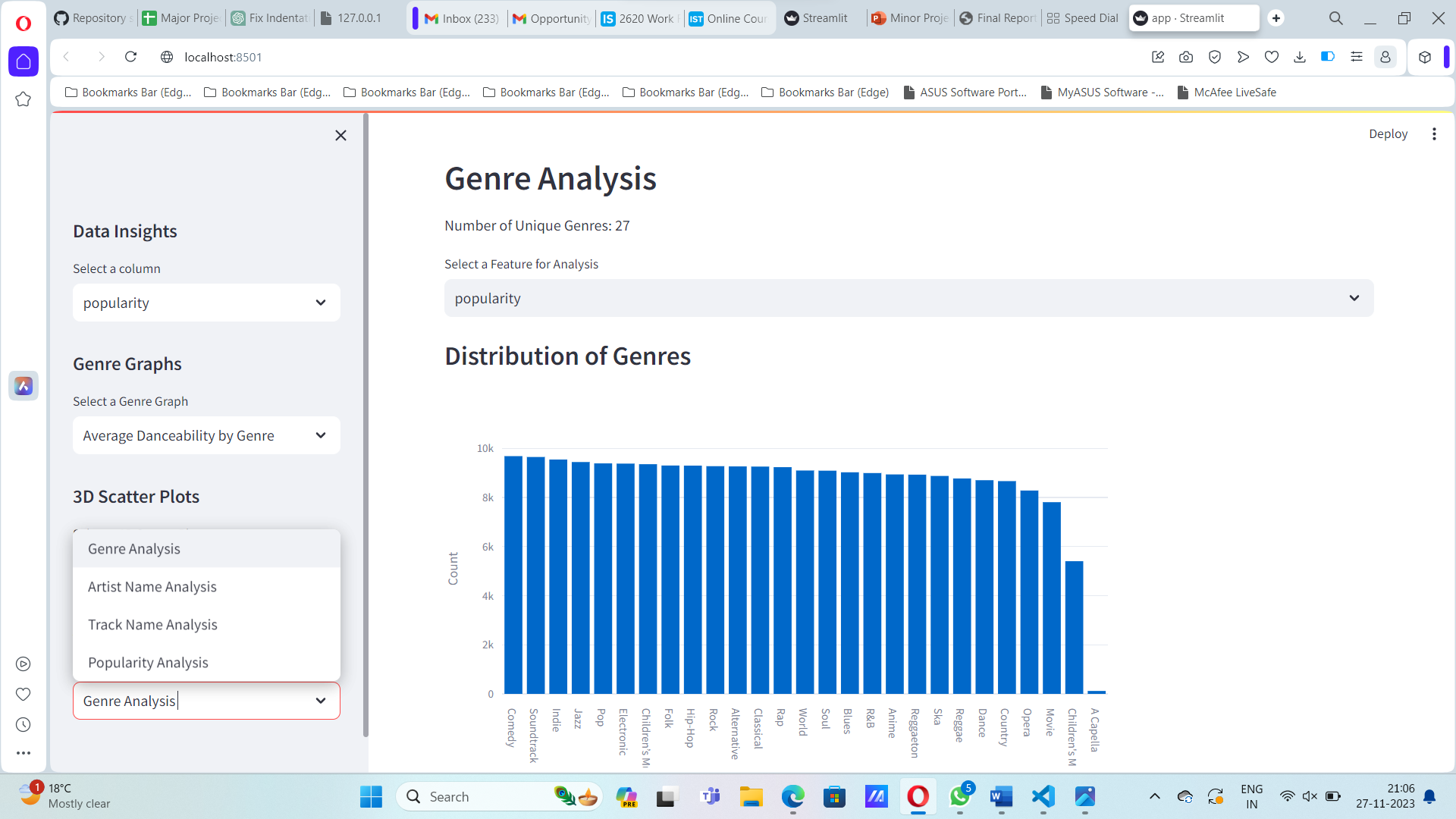
****

**Genre Analysis And 3D Scatter Plot**

## 

****

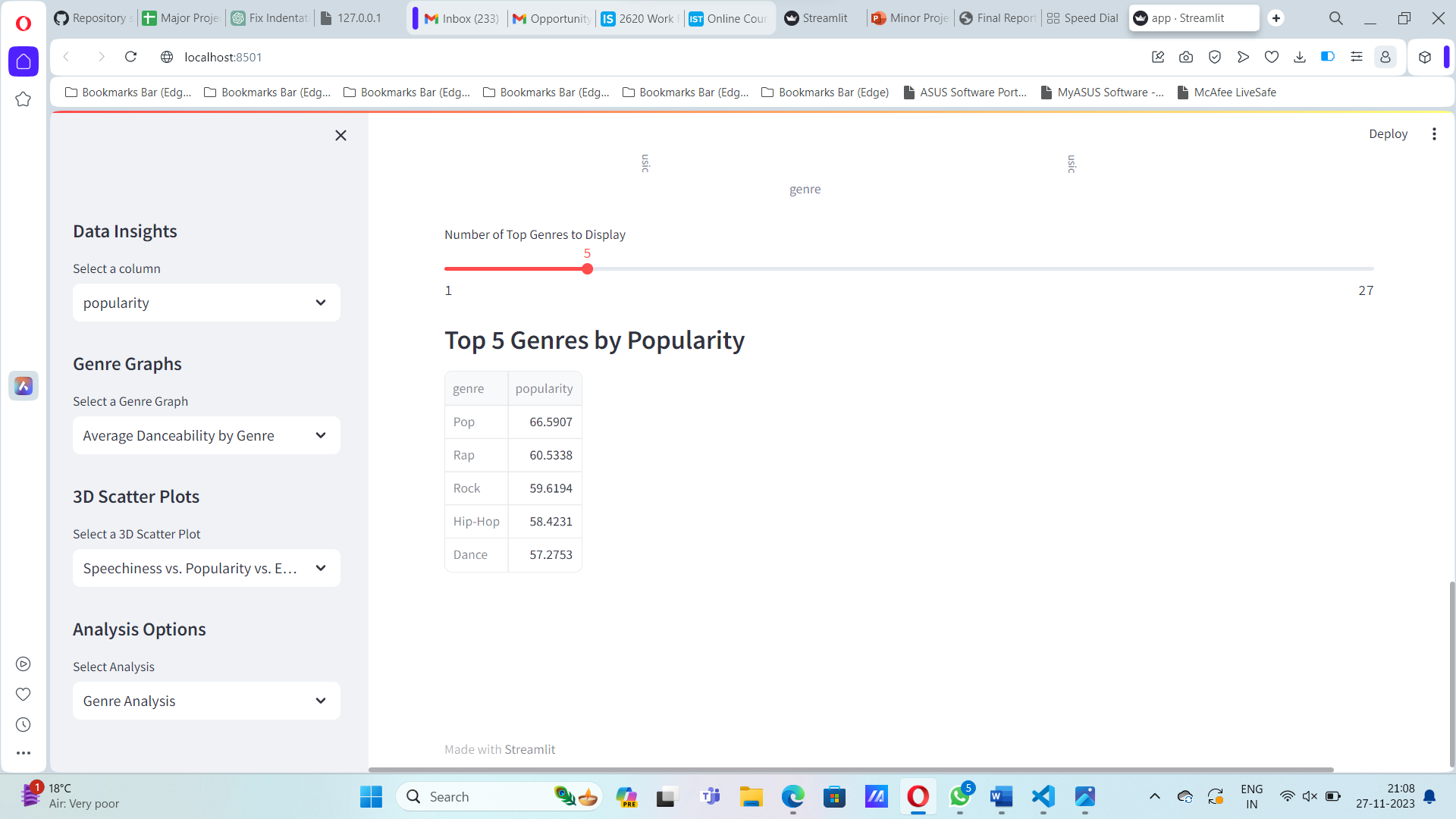
****

****

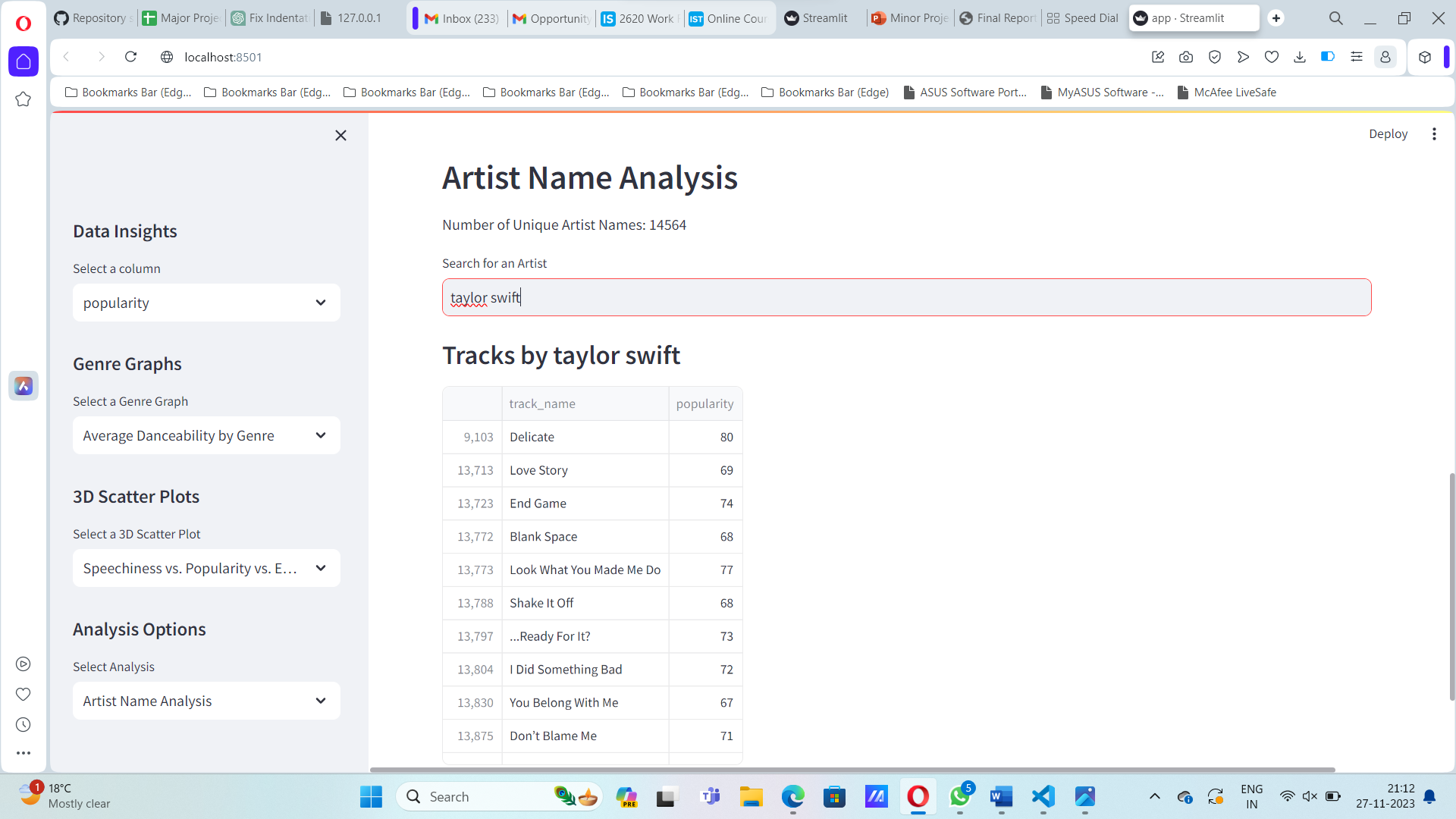
**Analysis Options**

**Genre Analysis**

## 

****

**Top 5 Genres by Popularity**

****

**Search Any Artist And get All The Available Tracks**

# 

# Select Track For Track Analysis

# 

# 

# Popularity Analysis

# 

# 

# 

# Coding Section

import streamlit as st

import pandas as pd

import matplotlib.pyplot as plt

import plotly.express as px

import plotly.graph\_objs as go

import seaborn as sns

st.set\_page\_config(layout='wide')

st.title("Spotify Data Analysis App")

def load\_data():

    df = pd.read\_csv(r'C:\Users\Namita\Downloads\SpotifyFeatures.csv')

    return df

df = load\_data()

col1, col2 = st.columns(2)

col3, col4 = st.columns(2)

with col1:

    st.dataframe(df)

with col2:

    numerical\_columns = ['popularity', 'acousticness', 'danceability', 'duration\_ms', 'energy',

                     'instrumentalness', 'key', 'liveness', 'loudness', 'speechiness',

                     'tempo', 'valence']

    st.sidebar.header("Data Insights")

    col = st.sidebar.selectbox("Select a column", numerical\_columns)

    st.markdown(f"<h3 style='text-align: center; font-size: 18px;'>Graph for {col}</h3>", unsafe\_allow\_html=True)

    fig, ax = plt.subplots(figsize=(15, 10))

    sns.histplot(df[col], bins=20, kde=True, ax=ax)

    ax.set\_title(f"{col}")

    st.pyplot(fig)

def genre\_analysis(df):

    st.header("Genre Analysis")

    num\_unique\_genres = df['genre'].nunique()

    st.write(f"Number of Unique Genres: {num\_unique\_genres}")

    feature\_choice = st.selectbox("Select a Feature for Analysis", ["popularity", "danceability", "energy"])

    genre\_counts = df['genre'].value\_counts()

    st.subheader("Distribution of Genres")

    fig\_genre = px.bar(genre\_counts, x=genre\_counts.index, y=genre\_counts.values, labels={"x": "Genre", "y": "Count"})

    st.plotly\_chart(fig\_genre)

    top\_genre\_count = st.slider("Number of Top Genres to Display", min\_value=1, max\_value=num\_unique\_genres, value=5)

    popular\_genres = df.groupby('genre')[feature\_choice].mean().sort\_values(ascending=False).head(top\_genre\_count)

    st.subheader(f"Top {top\_genre\_count} Genres by {feature\_choice.capitalize()}")

    st.write(popular\_genres)

def artist\_name\_analysis(df):

    st.header("Artist Name Analysis")

    num\_unique\_artists = df['artist\_name'].nunique()

    st.write(f"Number of Unique Artist Names: {num\_unique\_artists}")

    artist\_name = st.text\_input("Search for an Artist")

    if artist\_name:

        filtered\_df = df[df['artist\_name'].str.contains(artist\_name, case=False, na=False)]

        st.subheader(f"Tracks by {artist\_name}")

        st.write(filtered\_df[['track\_name', 'popularity']])

def track\_analysis(df):

    st.header("Track Analysis")

    track\_choice = st.selectbox("Select a Track", df['track\_name'].unique())

    if track\_choice:

        selected\_track = df[df['track\_name'] == track\_choice]

        st.subheader(f"Details for Track: {track\_choice}")

        st.write(selected\_track)

        st.subheader(f"Track Analysis for {track\_choice}")

        attributes = ['danceability', 'energy', 'valence']

        fig1, ax1 = plt.subplots(figsize=(10, 6))

        for attribute in attributes:

            ax1.bar(attribute, selected\_track.iloc[0][attribute], label=attribute)

        ax1.set\_title(f"Track Attributes for {track\_choice}")

        ax1.set\_ylabel("Value")

        ax1.legend()

        st.pyplot(fig1)

        if 'date' in df.columns:

            popularity\_over\_time = selected\_track[['date', 'popularity']]

            popularity\_over\_time['date'] = pd.to\_datetime(popularity\_over\_time['date'])

            popularity\_over\_time.set\_index('date', inplace=True)

            fig2, ax2 = plt.subplots(figsize=(12, 6))

            ax2.plot(popularity\_over\_time.index, popularity\_over\_time['popularity'], marker='o', linestyle='-')

            ax2.set\_title(f"Popularity Over Time for {track\_choice}")

            ax2.set\_xlabel("Date")

            ax2.set\_ylabel("Popularity")

            st.pyplot(fig2)

def popularity\_analysis(df):

    st.header("Popularity Analysis")

    st.subheader("Descriptive Statistics for Popularity")

    st.write(df['popularity'].describe())

    st.subheader("Popularity Distribution")

    fig\_hist = px.histogram(df, x='popularity', nbins=10, title="Popularity Distribution")

    st.plotly\_chart(fig\_hist)

    top\_count = st.number\_input("Number of Top Tracks to Display", min\_value=1, max\_value=len(df), value=10)

    bottom\_count = st.number\_input("Number of Bottom Tracks to Display", min\_value=1, max\_value=len(df), value=10)

    st.subheader(f"Top {top\_count} Highly Popular Tracks")

    top\_popular\_tracks = df.nlargest(top\_count, 'popularity')

    fig\_top = go.Figure(data=[go.Table(header=dict(values=["Track Name", "Popularity"]),

                                       cells=dict(values=[top\_popular\_tracks['track\_name'], top\_popular\_tracks['popularity']]))])

    st.plotly\_chart(fig\_top)

    st.subheader(f"Bottom {bottom\_count} Less Popular Tracks")

    bottom\_popular\_tracks = df.nsmallest(bottom\_count, 'popularity')

    fig\_bottom = go.Figure(data=[go.Table(header=dict(values=["Track Name", "Popularity"]),

                                         cells=dict(values=[bottom\_popular\_tracks['track\_name'], bottom\_popular\_tracks['popularity']]))])

    st.plotly\_chart(fig\_bottom)

    artist\_track\_counts = df['artist\_name'].value\_counts().reset\_index()

    artist\_track\_counts.columns = ['Artist', 'Track Count']

    artist\_track\_counts = artist\_track\_counts.sort\_values(by='Track Count', ascending=False)

with col3:  # Add Genre Graphs to col3

    st.sidebar.header("Genre Graphs")

    genre\_avg\_attributes = df.groupby('genre')[['popularity', 'energy', 'danceability']].mean().reset\_index()

    graph\_choice = st.sidebar.selectbox(

            "Select a Genre Graph",

            ["Average Popularity by Genre", "Average Energy by Genre", "Average Danceability by Genre"]

        )

    if graph\_choice == "Average Popularity by Genre":

        st.subheader("Average Popularity by Genre")

        fig = px.bar(

            genre\_avg\_attributes,

            x='popularity',

            y='genre',

            orientation='h',

            title='Average Popularity by Genre',

            color='popularity',

            color\_continuous\_scale='viridis')

        fig.update\_layout(xaxis\_title='Average Popularity', yaxis\_title='Genre')

        st.plotly\_chart(fig)

    elif graph\_choice == "Average Energy by Genre":

        st.subheader("Average Energy by Genre")

        fig2 = px.bar(

            genre\_avg\_attributes,

            x='energy',

            y='genre',

            orientation='h',

            title='Average Energy by Genre',

            color='energy',

            color\_continuous\_scale='viridis'

        )

        fig2.update\_layout(xaxis\_title='Average Energy', yaxis\_title='Genre')

        st.plotly\_chart(fig2)

    elif graph\_choice == "Average Danceability by Genre":

        st.subheader("Average Danceability by Genre")

        fig3 = px.bar(

            genre\_avg\_attributes,

            x='danceability',

            y='genre',

            orientation='h',

            title='Average Danceability by Genre',

            color='energy',

            color\_continuous\_scale='viridis'

        )

        fig3.update\_layout(xaxis\_title='Average Danceability', yaxis\_title='Genre')

        st.plotly\_chart(fig3)

with col4:

    st.sidebar.header("3D Scatter Plots")

    scatter\_col = st.sidebar.selectbox(

        "Select a 3D Scatter Plot",

        ["Liveness vs. Popularity vs. Energy", "Loudness vs. Popularity vs. Energy", "Speechiness vs. Popularity vs. Energy"]

    )

    if scatter\_col == "Liveness vs. Popularity vs. Energy":

        st.subheader("Liveness vs. Popularity vs. Energy")

        fig\_liveness = px.scatter\_3d(

            df,

            x='popularity',

            y='energy',

            z='liveness',

            color='liveness',

            title="Liveness vs. Popularity vs. Energy",

            labels={"popularity": "Popularity", "energy": "Energy", "liveness": "Liveness"},

            opacity=0.7

        )

        fig\_liveness.update\_layout(

            scene=dict(

                xaxis\_title="Popularity",

                yaxis\_title="Energy",

                zaxis\_title="Liveness",

            )

        )

        st.plotly\_chart(fig\_liveness)

    elif scatter\_col == "Loudness vs. Popularity vs. Energy":

        st.subheader("Loudness vs. Popularity vs. Energy")

        fig\_loudness = px.scatter\_3d(

            df,

            x='popularity',

            y='energy',

            z='loudness',

            color='loudness',

            title="Loudness vs. Popularity vs. Energy",

            labels={"popularity": "Popularity", "energy": "Energy", "loudness": "Loudness"},

            opacity=0.7

        )

        fig\_loudness.update\_layout(

            scene=dict(

                xaxis\_title="Popularity",

                yaxis\_title="Energy",

                zaxis\_title="Loudness",

            )

        )

        st.plotly\_chart(fig\_loudness)

    elif scatter\_col == "Speechiness vs. Popularity vs. Energy":

        st.subheader("Speechiness vs. Popularity vs. Energy")

        fig\_speechiness = px.scatter\_3d(

            df,

            x='popularity',

            y='energy',

            z='speechiness',

            color='speechiness',

            title="Speechiness vs. Popularity vs. Energy",

            labels={"popularity": "Popularity", "energy": "Energy", "speechiness": "Speechiness"},

            opacity=0.7

        )

        fig\_speechiness.update\_layout(

            scene=dict(

                xaxis\_title="Popularity",

                yaxis\_title="Energy",

                zaxis\_title="Speechiness",

            )

        )

        st.plotly\_chart(fig\_speechiness)

def main():

    st.sidebar.header("Analysis Options")

    df = load\_data()

    analysis\_choice = st.sidebar.selectbox(

        "Select Analysis",

        ["Genre Analysis", "Artist Name Analysis", "Track Name Analysis", "Popularity Analysis"]

    )

    if analysis\_choice == "Genre Analysis":

        genre\_analysis(df)

    elif analysis\_choice == "Artist Name Analysis":

        artist\_name\_analysis(df)

    elif analysis\_choice == "Track Name Analysis":

        track\_analysis(df)

    elif analysis\_choice == "Popularity Analysis":

        popularity\_analysis(df)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

# CONCLUSION

Our data analysis application, tailored for Spotify datasets, serves as a comprehensive tool for users seeking insights into music trends. The user-friendly interface, coupled with diverse analysis modules, ensures a seamless experience. By adopting efficient coding practices and modular structures, our project aims to meet the analytical needs of music enthusiasts and researchers alike. This application proves valuable in understanding genre dynamics, artist popularity, and track attributes, providing a holistic view of the Spotify dataset. In conclusion, our project aligns with the objective of software planning, offering a robust framework for data exploration that evolves with user requirements and advances in the field of music analysis.

**At the end it is concluded that we have made effort on following points…**

* Background and Context: Provided a comprehensive overview of the project's background, establishing its relevance to existing work in the music analysis domain.
* Aims and Objectives: Clearly stated the overarching goals and specific objectives guiding our efforts throughout the development process.
* Purpose, Scope, and Applicability: Articulated the purpose of the application, defined its scope, and outlined its practical applications in the realm of music data exploration.
* Problem Definition: Clearly defined and delineated the specific problem domain addressed by the project, emphasizing the need for effective music data analysis.
* Requirement Specifications: Presented a detailed breakdown of the system's requirements, specifying the actions that users can perform and the expected outcomes.
* Features and Operations: Explored the features and operations of the application in meticulous detail, elucidating the user interface and functionality through comprehensive screen layouts.

1. **REFERENCE**

<https://www.w3schools.com/php/>https://www.geeksforgeeks.org/php[tutoril/](https://www.geeksforgeeks.org/php-tutorial/)

<https://www.tutorialspoint.com/php/index.htm>[https://apache.org](https://apache.org/)</><https://www.tutorialspoint.com/execute_sql_online.php><https://riptutorial.com/ebook/php><https://freecomputerbooks.com/webPhpBooks.html><https://www.mysql.com/why-mysql/books/><https://riptutorial.com/ebook/mysql>

<https://www.oreilly.com/library/view/learning-mysql/0596008643/>