



PES UNIVERSITY, Bengaluru
Department of Computer Science and Engineering
B. Tech (CSE) – 5th Semester – Aug-Dec 2023

B.TECH. (CSE)
V Semester
UE21CS341A –Software Engineering

PROJECT REPORT
on

Spotify Artist Dashboard

Submitted by :

Rishabh T A	PES2UG21CS430	Ritunjay Rao	PES1UG21CS490
Ramit V Salunke	PES1UG21CS927	Rishab Gongulur	PES1UG22CS488

August – Dec 2023
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
BENGALURU – 560100, KARNATAKA, INDIA

<i>Table of Contents</i>		
Sl. No	Topic	Page No.
1.	Project Proposal	3
2.	Project Plan	4
3.	Software Requirements Specification	6
4.	Design Diagrams	18
5.	Test Plan, Cases	19
6.	Screenshots of Test Output	21

Project Proposal

Project Description :

The project aims to create a Spotify Artist Dashboard using Streamlit and the Spotify API. The dashboard allows users to search for information about artists or tracks, displaying details such as top tracks, albums, related artists, and track information.

Target Users:

Music enthusiasts, casual listeners, and anyone using the Spotify platform looking for recommendations on their favourite artists and tracks.

Functional Features:

1. **Artist Search:** Users can search for an artist and view details like top tracks, albums, and related artists.
2. **Track Search:** Users can search for a track and view detailed information about the track along with related tracks.
3. **Responsive UI:** The dashboard provides an interactive and responsive user interface through Streamlit.

Plan of Work :

Short-Term Plan (Next Few Weeks) :

1. Define project scope and requirements
2. Set up development environment
3. Implement basic artist and track search functionality
4. Integrate and test search features
5. Begin work on recommendation algorithms

Product Ownership :

1. Rishabh T A
2. Ramit V Salunke
3. Ritunjay Rao
4. Rishab Gongulur

Project Plan :

Life-cycle followed

Agile Development Cycle

Tools Used for this Project

- **Planning:** Streamlit, Spotipy library, Python
- **Design:** Streamlit layout components
- **Git:** Version control for collaborative development
- **Development:** Python, Streamlit, Spotipy
- **Bug Tracking:** Not explicitly specified in the code (consider adding a bug tracking tool if needed)
- **Manual Testing:** Streamlit's interactive testing
- **Documentation:** Inline comments, README file
- **Deployment**
- **Monitoring and Analytics**

Deliverables classified as reuse/build components :

Reusable/Build Components:

1. Streamlit UI components (e.g., text inputs, images, columns)
2. Spotipy library integration for Spotify API interaction
3. Artist and track information retrieval functions
4. User interface for displaying artist and track details

Non-reusable/Non-Build Components:

1. Spotify API credentials (consider securing these in a more robust manner)
2. Specific artist or track search queries (may vary based on user input)

Work Breakdown Structure

Project Initiation

- Define project goals and requirements
- Set up Spotify API credentials

System Design

- Design Streamlit UI layout
- Plan data retrieval from Spotify API

Development

- Implement Artist Search functionality
- Implement Track Search functionality
- Integrate Spotipy for Spotify API interaction

Testing

- Test the application's functionality
- Address and fix bugs

Documentation and Reporting

- Add inline comments for code documentation
- Create a README file with usage instructions

Maintenance and Support

- Provide ongoing support for bug fixes and updates

Effort Estimation (in person-days)

Project Initiation: 1 person-days

User requirements: 1 person-days

System Design: 3 person-days

Development: 7 person-days

Testing: 2 person-days

Documentation and Reporting: 2 person-days

Gantt Chart



Software Requirements Specification :

Introduction

Purpose:

- This document specifies the software requirements for the development of a Spotify Artist Dashboard.
- The dashboard is intended to provide detailed information about artists, including their most streamed, liked, and downloaded songs.
- It aims to enhance the user experience by offering insights into the artist's top tracks and achievements.

Intended Audience:

- This document is intended for developers, project managers, testers, and all stakeholders involved in the development and testing of the Spotify Artist Dashboard.
- It outlines the product's scope, functionality, and requirements.

Product Scope:

- The Spotify Artist Dashboard is a web-based application that allows users to search for artists and view various details, including the most streamed, liked, and downloaded songs.
- The dashboard will offer insights into an artist's popularity and track records on Spotify.

References:

- **Spotify API Documentation**
- **Spotify Developer Terms of Service**

Overall Description**Product Perspective:**

- **The Spotify Artist Dashboard is a standalone web application that interacts with the Spotify API to retrieve artist data. It is not part of a larger system but serves as a supplementary tool for Spotify users.**

Product Functions:

- **Artist search**
- **Displaying artist information and listing the most popular songs**
- **Artist recommendation based on the search**
- **Track Search**
- **Displaying track information**
- **Track recommendation based on the search**

User Classes and Characteristics:

- **Regular Users: Individuals who use the dashboard to search for their favorite artists and access artist information.**
- **Admins: System administrators responsible for user management and system maintenance.**

Operating Environment:

- **The Spotify Artist Dashboard will operate in a web environment and should be compatible with modern web browsers. It will interact with the Spotify API and rely on a web server for hosting.**

Design and Implementation Constraints:

- **The dashboard must comply with the Spotify Developer Terms of Service.**
- **The software will be developed using Streamlit, Spotify library, Python**

Assumptions and Dependencies:

- The availability and reliability of the Spotify API.
- Compliance with Spotify's data usage policies.
- Availability of web hosting and a domain for deployment.

External Interface Requirements

User Interfaces:

The user interface will be web-based and include:

- Search bar for artist and track queries
- Displaying artist information and listing the most popular songs
- Artist recommendation based on the search
- Displaying track information
- Track recommendation based on the search

Software Interfaces:

- The software will interact with the Spotify API for artist data retrieval and may use external libraries or frameworks for web development.

Communications Interfaces:

- The dashboard requires communication with the Spotify servers via the Spotify Web API

System Features

System Feature 1 (Artist Search):

Description and Priority

- This pivotal functionality empowers users to effortlessly search for their preferred artist by inputting the artist's name.

Stimulus/Response Sequences

- **Stimulus:** User initiates the process by entering the artist's name into the dedicated input field.
- **Response:** The system swiftly issues a query to the Spotify API, orchestrating a targeted search operation for the specified artist.

Functional Requirements

- The system must adeptly capture and process user input for the artist's name.
- It should establish seamless communication with the Spotify API, deploying the entered artist's name as a search parameter.
- In the case of an unsuccessful search, the system is mandated to gracefully communicate the absence of the artist to the user.
- An adept error-handling mechanism must be in place to manage exceptions gracefully and provide clear feedback to the user.

System Feature 2 (Artist Information Display):

Description and Priority

- This fundamental functionality enhances the user experience by presenting comprehensive details regarding a selected artist.

Stimulus/Response Sequences

- Stimulus: User showcases interest by selecting an artist from the search results.
- Response: The system meticulously retrieves and elegantly displays an array of information relevant to the selected artist.

Functional Requirements

- A comprehensive presentation of the artist's information, encompassing details such as images, follower count, and genres.
- The inclusion of captivating artist images in the sidebar for visual appeal and connection with the user.
- Display of the artist's followers count and a concise list of genres, creating a succinct yet informative snapshot of the artist's profile.
- An aesthetically pleasing layout ensuring clarity and coherence in the presentation of artist information.

System Feature 3 (Artist Recommendation)

Description and Priority

- This functionality enriches user interaction by furnishing curated recommendations of artists closely associated with the user's selected artist.

Stimulus/Response Sequences

- **Stimulus:** User explores the artist's information and expresses curiosity about related artists.
- **Response:** The system promptly generates and showcases a thoughtfully curated list of recommended artists, establishing a harmonious connection with the user's preferences.

Functional Requirements

- Seamless integration with the Spotify API to leverage its recommendation engine, thereby delivering a tailored set of recommended artists based on the user's selection.
- Display of essential details for each recommended artist, including names and visual representations.
- Inclusion of clickable links for each recommended artist, facilitating effortless exploration and navigation for users.
- Consideration of user interface design, limiting the number of recommendations to a user-friendly quantity (e.g., top 5), maintaining an engaging and uncluttered experience.

System Feature 4 (Track Search):

Description and Priority

- This pivotal functionality empowers users to seamlessly search for a desired track by inputting the track name.

Stimulus/Response Sequences

- **Stimulus:** User engages by entering the track name into the dedicated input field.
- **Response:** The system promptly dispatches a query to the Spotify API, orchestrating a search operation for the specified track.

Functional Requirements

- The system must robustly process and capture user input for the track name.
- It should exhibit efficient communication with the Spotify API, employing the entered track name as a search parameter.
- In the event of an unsuccessful search, the system is mandated to gracefully communicate the absence of the track to the user.
- An exceptional scenario handling mechanism must be in place to elegantly manage errors and provide clear feedback to the user.

System Feature 5 (Track Information Display):

Description and Priority

- This cardinal functionality enriches the user experience by presenting intricate details regarding a selected track.

Stimulus/Response Sequences

- **Stimulus:** User manifests interest by selecting a track from the search results.
- **Response:** The system diligently retrieves and elegantly displays an array of information pertinent to the selected track.

Functional Requirements

- A comprehensive display of the track's metadata, encompassing elements such as name, album, artists, release date, popularity, and an aesthetically pleasing album cover image.
- The provision of an intuitive hyperlink to the Spotify platform, affording users the opportunity to delve deeper into the selected track.
- Prudent formatting and layout considerations to ensure a visually cohesive presentation of the information.

System Feature 6 (Track Recommendation)

Description and Priority

- This functionality amplifies user engagement by offering curated recommendations of tracks closely associated with the user's selected track.

Stimulus/Response Sequences

- **Stimulus:** User explores the track information and expresses curiosity about related tracks.
- **Response:** The system promptly generates and showcases a thoughtfully curated list of recommended tracks, establishing a harmonious connection with the user's preferences.

Functional Requirements

- Seamless integration with the Spotify API to leverage its recommendation engine, thereby delivering a tailored set of recommended tracks based on the user's selection.
- Display of essential details for each recommended track, including names, artists, album names, and visually striking album cover images.
- Inclusion of clickable links for each recommended track, facilitating effortless exploration and navigation for users.

- Thoughtful consideration of user interface design, limiting the number of recommendations to a user-friendly quantity (e.g., top 5).

Other Non-Functional Requirements

Performance Requirements:

- The system should respond to user queries within a reasonable time frame. Searches and data retrieval should be efficient.

Safety Requirements

- The system must ensure the privacy and security of user data, following Spotify's guidelines and regulations.

Security Requirements

- The system must implement user identity authentication and protect user data in accordance with best practices and Spotify's security policies.

Software Quality Attributes

- The system should prioritize usability, reliability, and performance. It should be easy to use and responsive to user interactions.

Business Rules

- Only registered users on spotify can access certain features of the dashboard.
- User data will be kept confidential and used in compliance with privacy regulations.

Other Requirements

- The dashboard must comply with international data protection regulations.
- All code must be documented for ease of maintenance.
- The dashboard must be mobile-responsive for a seamless user experience on various devices.

Software Requirements

Development Tools:

- **Integrated Development Environment (IDE):** Software development environment such as Visual Studio Code
- **Version Control:** Git for version control and collaboration among developers.
- **Text Editor:** For editing python files

Web Development Technologies:

Front-End:

- **Streamlit:** Streamlit is a Python library used for creating interactive and web-like applications for data science and machine learning.

Back-End:

- **Python:** Python is used as the primary programming language for implementing the back-end logic of the application. It is responsible for handling interactions with the Spotify API, processing data, and providing information to the front-end.
- **Spotipy:** Spotipy is a Python library that serves as a client for the Spotify Web API. It simplifies the process of interacting with the Spotify API by providing functions to search for artists, tracks, retrieve top tracks, albums, related artists, and more. Spotipy is used as the back-end tool for communicating with the Spotify API.
- **Spotify API:** While not a tool per se, the Spotify API is a crucial component of the back-end. It allows the application to access and retrieve data from the Spotify platform, including artist information, track details, and related data.
- **Git:** Git is used for version control, enabling collaborative development and tracking changes to the codebase. While not explicitly specified in the code, the use of Git is common in software development projects.

Hardware Requirements

Development Environment:

- **A computer or workstation with sufficient processing power, memory, and storage for web development.**
- **Multi-core CPU and at least 8GB of RAM are recommended for efficient development.**

Appendix A: Glossary

1. **Spotify Artist Dashboard:** A web-based application providing detailed information about artists, including their most streamed, liked, and downloaded songs, enhancing the user experience on Spotify.
2. **Stakeholders:** Individuals or groups involved in the development and testing of the Spotify Artist Dashboard, including developers, project managers, testers, and system administrators.
3. **Product Scope:** The defined boundaries and functionalities of the Spotify Artist Dashboard, specifying its purpose and intended user interactions.
4. **References:** External documents or resources crucial for the development, including the Spotify API Documentation and Spotify Developer Terms of Service.
5. **User Classes:** Categorizations of users based on their roles, including Regular Users and Admins, each having specific interactions with the dashboard.
6. **Operating Environment:** The web-based environment in which the Spotify Artist Dashboard operates, compatible with modern web browsers and relying on the Spotify API and web hosting.

- 7. Design and Implementation Constraints:** Restrictions and limitations during the design and implementation phase, including compliance with Spotify Developer Terms of Service and the use of Streamlit, Spotipy library, and Python.
- 8. Assumptions and Dependencies:** Preconditions considered true without validation and external factors on which the project depends, such as the availability and reliability of the Spotify API.
- 9. External Interface Requirements:** Specifications for user interfaces, software interfaces, and communication interfaces, detailing how users interact with the system and how it interacts with external components like the Spotify API.
- 10. System Features:** Key functionalities of the Spotify Artist Dashboard, including Artist Search, Artist Information Display, Artist Recommendation, Track Search, Track Information Display, and Track Recommendation.
- 11. Performance Requirements:** Specifications regarding the system's responsiveness and efficiency, ensuring timely responses to user queries and efficient data retrieval.
- 12. Safety Requirements:** Measures to ensure the privacy and security of user data, adhering to Spotify's guidelines and regulations.
- 13. Security Requirements:** Specifications for implementing user identity authentication and protecting user data in accordance with best practices and Spotify's security policies.

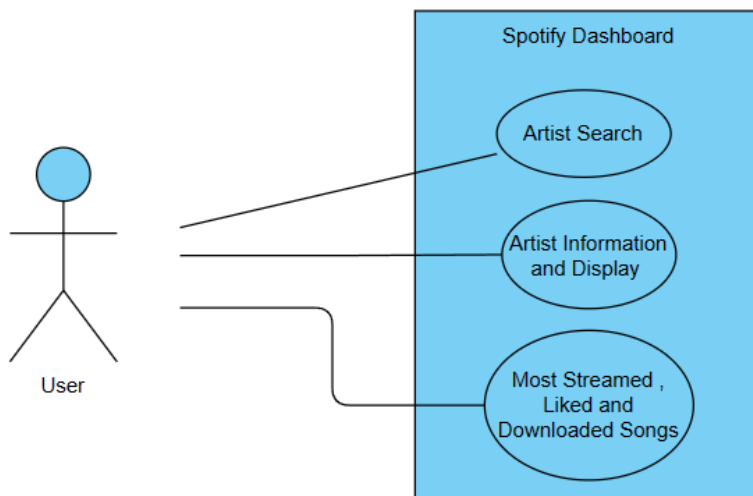
- 14. Software Quality Attributes:** Characteristics prioritized in the development, including usability, reliability, and performance, ensuring the application is easy to use and responsive.
- 15. Business Rules:** Regulations governing the use of the Spotify Artist Dashboard, including access limitations for non-registered users and the confidential use of user data.
- 16. Other Requirements:** Additional specifications, including compliance with international data protection regulations, code documentation, and mobile responsiveness for various devices.
- 17. Development Tools:** Tools and technologies used in the development process, including the Integrated Development Environment (IDE), Version Control (Git), Text Editor, Streamlit, Python, Spotipy, and the Spotify API.
- 18. Hardware Requirements:** Specifications for the hardware needed in the development environment, including a computer or workstation with sufficient processing power, memory, and storage for web development.

Appendix B: Requirement Traceability Matrix

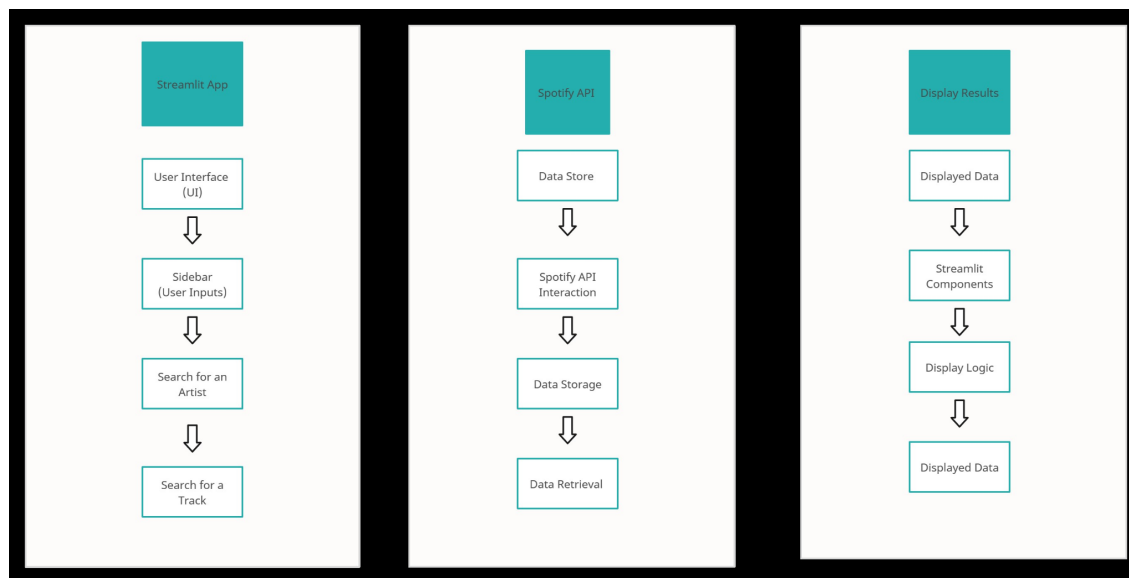
Sl. No	Requirement ID	Brief Description of Requirement	Architecture Reference	Design Reference	Code File Reference	Test Case ID	System Test Case ID
1	REQ001	Display artist information	Sidebar, Expander	Streamlit layout	app.py, lines 15-41	TC001	STC001
2	REQ002	Display top tracks of an artist	Spotify API, Sidebar	Streamlit layout	app.py, lines 43-51	TC002	STC002
3	REQ003	Display album information	Spotify API,	Streamlit layout	app.py, lines 53-71	TC003	STC003
4	REQ004	Display top tracks from an album	Expander	Streamlit layout	app.py, lines 73-84	TC004	STC004
5	REQ005	Display related artists	Expander	Streamlit layout	app.py, lines 86-99	TC005	STC005
6	REQ006	Search for a track and display info	Two-column layout	Streamlit layout	app.py, lines 101-131	TC006	STC006

Design Diagrams:

1. Use Case Diagram



2. Data Flow Diagram



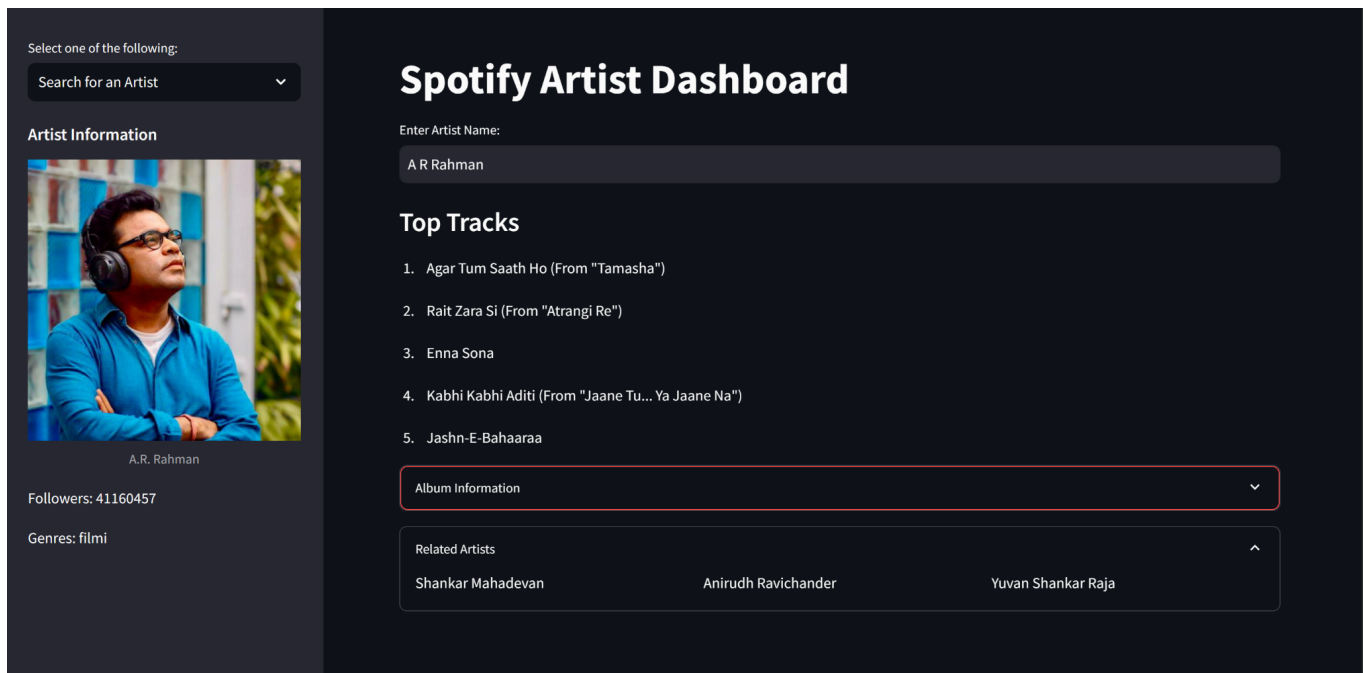
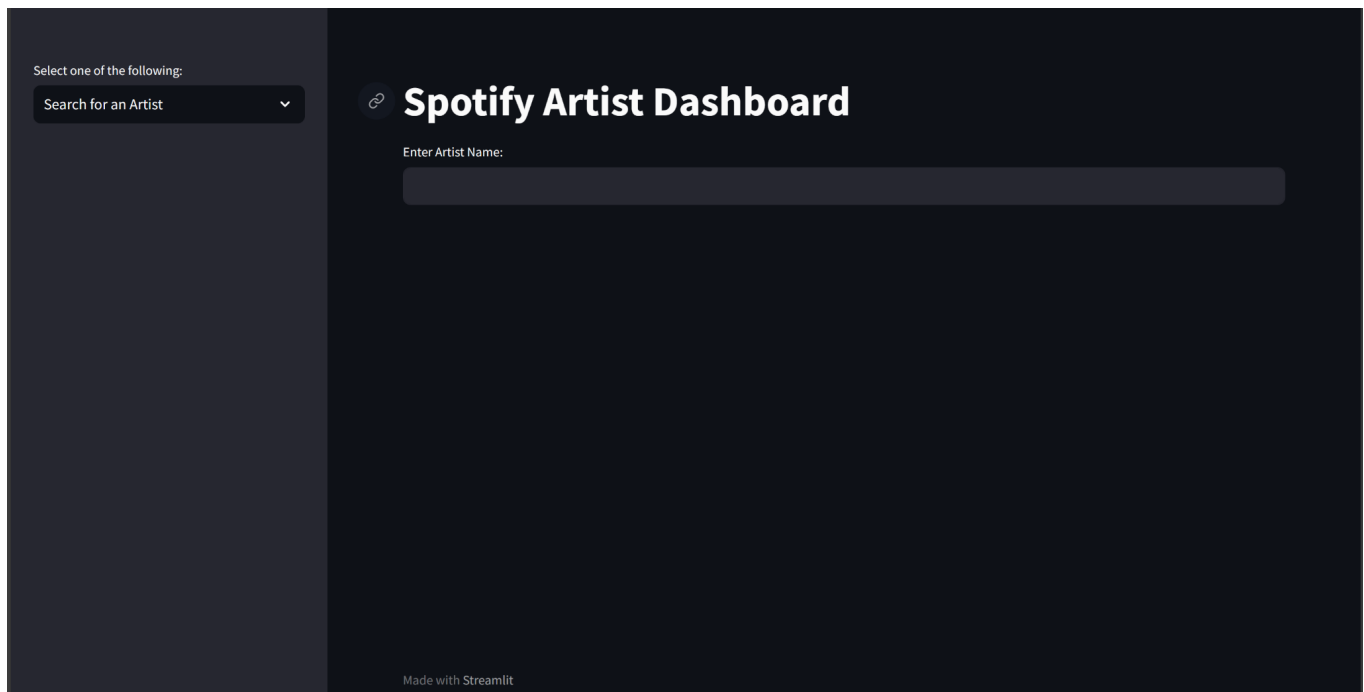
Test Plan and Cases:

Test Case:

Test Case ID	Name of Module	Test Case Description	Pre-conditions	Test Steps	Test Data	Expected Results	Actual Results	Test Result
TC001	Artist Search	Verify artist information display	User searches for an artist, and the application should display the artist's image, followers, genres, and top tracks.	Spotify API credentials provided in the code	1. Enter the artist name in the input field. 2. Click the search button.	Valid artist name	Artist information is displayed correctly, including image, followers, genres, and top tracks.	Pass
TC002	Top Tracks Display	Verify top tracks are displayed	After artist search, the application should display the top tracks of the artist.	Artist information displayed (from TC001)	No additional steps required.	Valid artist with top tracks	Top tracks of the artist are displayed correctly.	Pass
TC003	Album Information	Verify album information display	After artist search, the application should display the top albums of the artist.	Artist information displayed (from TC001)	Expand the "Album Information" section.	Valid artist with albums	Top 3 albums of the artist are displayed correctly, including album name, image, and release date.	Pass
TC004	Top Tracks from Album	Verify top tracks from an album are displayed	After expanding the album information, the application should	Album information displayed (from	Expand the album details.	Valid album ID	Top 5 tracks from the album are displayed correctly.	Pass

			display the top tracks from the selected album.	TC003)				
TC005	Related Artists	Verify related artists are displayed	After artist search, the application should display related artists in a separate section.	Artist information displayed (from TC001)	Expand the "Related Artists" section.	Valid artist with related artists	Related artists are displayed correctly.	Pass
TC006	Track Search and Display	Verify track information display	User searches for a track, and the application should display track details including name, artists, album, etc.	Spotify API credentials provided in the code	1. Enter the track name in the input field. 2. Click the search button.	Valid track name	Track information is displayed correctly, including name, artists, album, release date, and popularity.	Pass
TC007	Related Tracks Display	Verify related tracks are displayed	After track search, the application should display related tracks on the main page.	Track information displayed (from TC006)	No additional steps required.	Valid track with related tracks	Related tracks are displayed correctly, including name, artists, album, and images.	Pass
TC008	Error Handling	Verify error messages are displayed for invalid inputs	User enters invalid artist/track names, and the application should display appropriate error messages.	Spotify API credentials provided in the code	1. Enter an invalid artist name. 2. Click the search button.	Invalid artist name	Error message is displayed indicating that the artist is not found.	Pass


Screenshots :



Select one of the following:

Search for an Artist

Artist Information




A.R. Rahman

Followers: 41160457

Genres: filmi

Album Information

[Sangamam \(Karaoke\)](#)



Sangamam (Karaoke)

Release Date: 2023-11-15

Top Tracks:

1. Mazhai Thulli


2. Varaha Nadhikarai

3. Sowkiyama Kannae

4. Mudhal Murai

5. Margazhi Thingal Allava

[Pippa \(Original Motion Picture Soundtrack\)](#)



Pippa (Original Motion Picture Soundtrack)

Release Date: 2023-11-14

Top Tracks:

1. Rampage - From "Pippa"


2. Main Parwaana - From "Pippa"

3. Jazbaat - From "Pippa"

4. Karar Oi Louho Kapat - From "Pippa"

5. Mohabbatein Shukriya - From "Pippa"

[En Swasa Katre \(Karaoke\)](#)



En Swasa Katre (Karaoke)

Release Date: 2023-11-07

Top Tracks:

1. Jumballaka

2. Chinna China

3. En Swasa Katre

4. Thathi Aaduthe

5. Theendai

Related Artists

Select one of the following:

Search for a Track

Spotify Artist Dashboard

Enter Track Name:

Made with Streamlit

Aug-Dec 2023

UE21CS341A : SE

Page 22 / 23

Select one of the following:


Search for a Track

Spotify Artist Dashboard

Enter Track Name:

Agar Tum Saath Ho

Track Information



Tamasha

Name: [Agar Tum Saath Ho](#)

Artists: Alka Yagnik, Arijit Singh

Album: Tamasha


Release Date: 2015-10-16

Popularity: 77

0:00 / 0:29


Related Tracks on the Main Page

- [Chithi Na Koi Sandesh](#) - Jagjit Singh



Kahin Door Jab
Din Dhal Jaye

- [Teri Yaad Yaad Yaad](#) - Ghulam Ali



Bewafaa
(Original Motion
Picture
Soundtrack)