**FINAL PROJECT**

**Course: INFO 1113 A13**

**Instructor: Dr. Abhijit Sen**

**Date of Submission: August 2, 2021**

**Members:**

**Jatin Sharma ()**

**Parmpal Singh Bajwa (100400167)**

**Rasnoor Singh ()**

**Rene De Lara ()**

**MusiVibe**

**Executive Summary**

Imagine a world where you find a song in a second that resonates with your mood. It is extremely frustrating to find a perfect track by sifting through countless playlists with as many songs in them. An average person wastes 1 minute to find the right track of 3:30 minutes. Our project challenges the foundation of all the current music apps which organise their content by genres.

Our new application begins with 2-month free access to premium experience for the user to evaluate its worth. Following will be the main features that our innovative application will provide:

* Songs are searched through the algorithm of mood, not genre.
* A matching track is put into the search bar and a list generates which matches the vibe of that song.
* Introduces making separate playlists for these songs as per the user.
* A premium experience, at reasonable price, with options of downloading and no interruptions by adds.

Feedback by company employees’ friends and relatives provides excellent response to the application as they explain how easy and quicky it is to find a song that sets them in the right mood. They highlighted the time saving as the most lucrative advantage, while the interface had its problems. However, as the development of the project progresses, significant changes will continue to be made in order to improve the user experience.

**Introduction:**

The project, MusiVibe, is application-development project. This project will enlist all the functional as well as non-functional requirements of the application. There will be four use case diagrams which will describe the corresponding functional requirement since use case diagramming is used to model the functionality of the system. A class diagram will also show the basic structure of object classes and their internal relationships within the system. Then, a thorough description of four use case diagrams and a sequence diagram will be included to provide extensive details of the specific case. Interface prototypes will also be developed and presented to give an idea how different windows will appear to the user. Moreover, relational database tables normalised to 3NF form will be constructed too, to delineate how data is stored and databases are managed.

**Requirements:**

***Functional Requirements:***

1. When the user enters the name of the song, the system reverts a list of songs matching the vibe of the song.
2. When the user enters the vibe of the song like melancholy or chill, the system reverts corresponding playlists.
3. The application’s server shall be daily updated with new as well as old tracks.
4. Evaluate the duration of the song played and sends information to the server.
5. The application makes recommendations to the user using stored information the server.
6. Two extensions of application: free and premium. The latter has to be bough through monthly or yearly subscription.

***Non-functional Requirements:***

1. The buttons in the app will be bigger enough to be visible and clickable.
2. The colors will be dull like beige to showcase an aesthetic appearance.
3. The application must function 24/7.
4. The response time should not exceed two seconds.
5. Safe installation for the user with built-in cyber protection.

**Use Case Diagrams**

#1

***Login***

Diagram

Description automatically generated

#2

***Choose Access Type***

Diagram

Description automatically generated

#3

***Get Premium Access***

Diagram

Description automatically generated

***Use Case Description of #3***

|  |
| --- |
| **Use case tittle:** Get Premium Access |
| **Primary actor:** Music Listener |
| **Level:** Kite (Summary) |
| **Stakeholders:** Music Listener, Credit Bureau |
| **Precondition**: Music Listener has the application installed and ready to use. |
| **Minimal guarantee:** User can access the music with ads (i.e., no effect on previous experience of application) |
| **Success guarantee:**  User is allowed to access the Music without any ads after the user has purchased the premium access with a credit card |
| **Trigger:** Music Listener opens “Buy Premium” page |
| **Main description**:   * The Listener buys premium access by entering Credit Card Details, which are verified by the Credit Bureau. * No interruptions of ads in the music experience. * Introduces making separate playlists for these songs as per the   User’s discretion.   * Premium experience of playing songs while in the offline mode. |
| **Extensions:**  The music listener can peruse the music available on a website without anu interruption after paying for the premium access.  1a. The Listener is interrupted by an add while buying access.  1a1. The process is restarted.  1a2. The Listener closes the application.  2a. Credit Bureau cannot verify that the listener has necessary credit.  2a1. Listener is notified of the issue and they close the application.  2a2. The Listener is notified of the issue and they try again with new card.  3a. Confirmation of the purchase is interrupted.  3a1. The Listener tries another way of confirming the purchase.  3a2. The Listener closes the application. |

#4

***Search Options***

Diagram

Description automatically generated

#5

***Diagram

Description automatically generatedSearch by Song Name***

***Use Case Description of #5***

|  |
| --- |
| **Use case tittle:**  Search by Song Name |
| **Primary actor:** Music Listener |
| **Level:** Kite (Summary) |
| **Stakeholders:** Music listener |
| **Precondition**: The listener has the application installed with or without Premium Access |
| **Minimal Guarantee**: The application returns generic list of songs. |
| **Success guarantee:**  Music Listener can find the songs matching the vibe of the song entered in the search bar. |
| **Trigger:** Music Listener types in the name of a song in search bar |
| **Main description**:   * The name of the song is entered in the search bar. * The server runs an algorithm to find the vibe of the song. * The application then presents a playlist of the songs matching that vibe. * The music listener gets to play any song they like. |
| **Extension:**  The music listener, in this case, types for the song whose vibe they like. Once the listener searches the song, confirmation of the music vibe is done from the listener.  1a. A vibe matching that song does not exist.  1a1. The listener closes the application.  2a. The vibe’s name is not confirmed by the Listener.  2a1. The listener enters the name of the specific vibe in search bar.  3a. Search is interrupted by an ad.  3a1. Listener waits for the ad to finish and listens to music afterwards.  3a2. Listener closes the application. |

#6

***Search by Vibe Name***

Diagram

Description automatically generated

#7

***Play Music***

Diagram

Description automatically generated

**Use Case Description of #7**

|  |
| --- |
| **Use case tittle:**  Play Music |
| **Primary actor:** Music Listener |
| **Level:** Kite (Summary) |
| **Stakeholders:** Music listener |
| **Precondition**: The listener has the application installed |
| **Minimal Guarantee**: The application shows Music options |
| **Success guarantee:**  Music Listener listens to music as per their own wish |
| **Trigger:** Music Listener taps on a song to play it |
| **Main description**:   * The Listener taps on any song to listen to, whether through search of vibe/song. * The application plays the song with options to pause, play, stop the song plus volume bar. * The Listener can choose what to do with these options as per their own discretion. * The Listener can search for another song while listening to the current song. |
| **Extensions:**  The music listener, in this case, plays music to listen to, thus accessing the core function of the application.  1a. The Listener does not tap on any song.  1a1. The listener closes the application.  2a. The Listener is unable to pauses/stop the song.  2a1. Song is played again.  2a2. The listener closes the application.  3a. The listener is not able to manipulate the volume bar.  3a1. The application is relaunched.  3a3. The listener closes the application.  4a. Music is interrupted by an ad.  4a1. Listener waits for the ad to finish and plays another song.  4a2. Listener closes the application. |

#8

***Get Recommendation from App***

Diagram

Description automatically generated

#9

**Add Social Media Accounts**

Diagram

Description automatically generated

***Use Case Description of #9***

|  |
| --- |
| **Use case tittle:**  Add Social Media Accounts |
| **Primary actor:** Music Listener |
| **Level:** Kite (Summary) |
| **Stakeholders:** Music listener, Social Media Apps |
| **Precondition**: The listener has the application installed and some personal social media accounts |
| **Minimal Guarantee**: The application remains unlinked with social media accounts with the same access the user had |
| **Success guarantee:**  Music Listener gets their social media accounts linked with the application |
| **Trigger:** Music Listener accesses the option of “Add Social Media Accounts” |
| **Main description**:   * The Listener goes to “Add Social Media Accounts” option available in the homepage. * The application offers the list of apps that can be linked: Facebook, Instagram and Twitter. * The Listener chooses one app at one time and enter the credentials to link the account. * The Listener can now follow their friends and listen to each other’s playlists while remaining on the MusiVibe app. |
| **Extensions:**  The music listener, in this case, tries to link their social media accounts with the MusiVibe to build a community and share their taste of music with one another.  1a. The Listener does not have any social media accounts.  1a1. The listener closes this tab.  1a2. The listener closes the application.  2a. The Listener does not have one/two of the available social media accounts.  2a1. The available account is linked.  2a2. The other accounts are left unlinked.  3a. Process is interrupted by an ad.  3a1. Listener waits for the ad to finish the linking of social media accounts.  3a2. Listener closes the application. |

**Class Diagram**

Diagram

Description automatically generated

**Sequence Diagram**

The following sequence diagram is based on the use case diagram of “Search by Song Name” (#5).

**Diagram

Description automatically generated**

**Interface Prototypes:**

***Diagram

Description automatically generatedHierarchy Flowchart***

***The Listener has the option to Sign in with Username/Password or Sign up with Google/Facebook***

***Graphical user interface, application

Description automatically generated***

Graphical user interface, application

Description automatically generated***Sign in Options of Facebook/Google***

Graphical user interface, text, application

Description automatically generated

***Graphical user interface, application

Description automatically generatedSearch By Vibe***

***Graphical user interface, text, application

Description automatically generatedSearch by Song***

***Playlist Edit Layout***

Graphical user interface

Description automatically generated

***Graphical user interface, application

Description automatically generatedMusic Player***

**Relational Database Tables**

**USER**

|  |  |  |
| --- | --- | --- |
| USER\_ID | Name | Premium\_Status |
| 1234 | Joe | Yes |
| 5555 | Susan | No |

**PLAYLIST**

|  |  |  |  |
| --- | --- | --- | --- |
| Playlist\_ID | USER\_ID | Playlist\_Name | Playlist\_Order |
| 7\*b86 | 1234 | Sad Jams | \*\*\*\* |
| 9@0%2 | 5555 | Driving Tunes | \*\*\*\* |

**SONG**

|  |  |  |
| --- | --- | --- |
| Song\_ID | Vibe\_ID | Name |
| 33333 | \_uplift | Bohemian Rhapsody |
| 57200 | \_sd | That funny feeling |

**VIBE**

|  |  |
| --- | --- |
| Vibe\_ID | Name |
| \_uplift | Uplifting |
| \_sd | Sad |

**Diagram

Description automatically generated**