

Dharmsinh Desai University, Nadiad

Faculty of Technology, Department of Computer Engineering

B.Tech. CE Semester – VI

Subject: System Design Practice

Project Title:

Live Auctioning System

[ BestBid ]

By:

Sheth Aarsh A. - Roll No: CE126, ID: 17CEUON052

Patel Zarmy - Roll No: CE097, ID: 17CEUON

Malla Sneha - Roll No: CE127, ID: 17CEUOS

Guided By:

Prof. Brijesh S. Bhatt



Dharmsinh Desai University, Nadiad

Faculty of Technology, Department of Computer Engineering

**CERTIFICATE**

This is to certify that Object Oriented Software Engineering project entitled “Music Music Streaming Application” is the bonafide report of work carried out by

1. **Sheth Aarsh A. (17CEUON052)**
2. **Patel Zarmy (17CEUON)**
3. **Malla Sneha (17CEUOS)**

of Department of Computer Engineering, Semester VI, academic year 2019-20, under our supervision and guidance.

|  |  |  |
| --- | --- | --- |
| Guide |  | HOD |
|  |  |  |
| **Prof. Brijesh S. Bhatt** |  | **Dr. C. K. Bhensdadia** |
| Assistant Professor |  | Head of the Department |
| Department of Computer |  | Department of Computer |
| FOT, Dharmsinh Desai University, Nadiad. |  | FOT, Dharmsinh Desai University, Nadiad. |

Table of Contents

1 Abstract...………………………..…………………..…………………….….……………...…4 2 Introduction……………………………………………………………….….…….…..............5 2.1 Project Details: Brief Introduction …………………..............................................................5 2.2 Technology and Tools Used ...................................................................................................5 3 Software Requirement Specifications ………………………...…………..………………..….7 3.1 Scope ………………………………………………………...…………..……………….….7 3.2 System Functional Requirements ……………….………………………..…………….……7 3.3 Other Non‐Functional Requirements ……………….……………………...……………….13 4 Design……………….……....……………………….………………………..……..….…….14 4.1 Use Case Diagram .………….…………..…….………………………….………….……...14 4.2 Class Diagram..…………….………………………...….………………….……………….16 4.3 Sequence Diagram……..……………………………………………….…….…......……….17 4.4 Activity Diagram………….…………………………………………………….……..…….20 4.5 State Diagram……………………………………………………………………..……….…21 4.6 E-R Diagram ………….………………………………………………..…...…………….....22 4.7 Data Dictionary …..………………………………………………………………….............23 5 Implementation Details ……………………………….……………………………….…....…27 5.1 Modules …………………………………..…………………………………………….…...27 5.2 Functional prototypes ……………………………………………………………………….28 6 Testing ………………………………………………………………………………………...30 6.1 Testing Method….………………………………….……………………………....……….30 7 Screen‐shots of the System …………………………………………………......……..……...30 8 Conclusion ……………………………………………………………………….…...……….32 9 Limitations and Future Extensions of System ……………………………………………...…33 10 Bibliography ……………………………………………………….……………………..….33

1. Abstract

With the new coming technologies, the world around us has transformed to a form making our daily life more comfortable. The new techniques available now satisfy our daily needs in a simpler way than earlier. Similarly, the way we used to listen music from old gramophones to CDs has now taken a form of online music applications. Sunofy is one of the examples. A user must get a good user experience along with good music of his/her own choice. The features like application design, usability, streaming quality and music sharing capability matter for a music platform.

2. Introduction

**2.1 Brief Introduction**

****

Sunofy is a music streaming application which allows user to listen songs, create his/her playlist, add songs, delete playlist and manage profile. The application provides various media player controls for user convenience. Ultimately, the application provides a user interface for music streaming.

**2.2 Tools/Technologies Used**

**Technologies:**

Python

Django

Bootstrap

JavaScript

jQuery

Redis

**Tools:**

Visual Studio Code

Git

3. Software Requirement Specifications

**3.1 Product Scope**

This system is designed to enable the user to browse the songs, listen to the songs, manage profile and get song recommendations.

**Types of User**

1. User

2. Admin

**3.2 System Functional Requirements**

**R.1 User**

R.1.1 Manage Account

Description: User can manage his/her account.

R.1.1.1 User Login

Description: It enables user authentication. A valid user account must be used for an existing user.

Input: A user can login using his/her user name and password.

Output: The system will verify that the login name matches the login password. If the user name or password is invalid, the appropriate error message will be indicated and the user will be requested to re-enter user name and password. If the user inputs are valid, the main page will be displayed.

R.1.1.2 User Registration

Description: This is implemented to enable a new user authentication. A valid user account must be used for an existing customer or a new customer can register.

Input: If the customer is a new user, he can request to register with the system.

Output: The system displays a registration page and asks the customer to choose a user name, password and enter a valid email id, security question and answer.

R.1.2 Manage User Profile

Description: User maintains his/her profile

R.1.2.1 Follow Friends

Description: User can follow other users in their contacts or artists

Input: User selection

Output: confirmation of success

R.1.2.2 View Followers

Description: User can see list of his/her followers

Input: User selection

Output: List of followers

R.1.2.3 View Followings

Description: User can view the list users or artists his/her follows.

Input: User selection

Output: List of followings

R.1.2.4 Unfollow friend or artist

Description: User can unfollow friend or artist it followed

Input: User selection

Output: confirmation of success

R.1.2.5 Share Profile

Description: User can share his profile with other users

Input: User selection

Output: confirmation message

R.1.3 Manage Songs

Description: User can manage songs

R.1.3.1 Listen Song

Description: user can listen to the songs

Input : User selection.

Output: plays song in music player

R.1.3.2 Download Songs

Description: user can download his/her favorite songs

Input : User selection.

Output: confirmation

R.1.3.3 Browse songs

Description: A customer can browse through the different genres, categories and can also view the details of the products such as the description, price and songs listing etc. Depending upon the genre/category selected, the contents of the table are accessed using a select query.

Input: User will select one of the genres and its category.

Output: The system will display song list and information of the selected genre and/or category. The song list will be displayed and 10 songs will be displayed on each page and the rest (if any) will be on the next page. This will be executed using the “pagination” property i.e. there will a link named “Previous” and “Next” on the bottom of page to enable the users to go to the next and previous pages to view songs. The current page of the user will also be displayed on every page.

R.1.3.4 Like Song

Description: user can like the songs

Input : User selection.

Output: confirmation message

R.1.4 Manage Playlist

Description: User can manage playlist

R.1.4.1 Create Playlist

Description: User can create a playlist of songs according to his taste.

Input: List of songs and playlist name

Output: Success message

R.1.4.2 Delete Playlist

Input: User selection

Output: Success message

R.1.4.3 Edit Playlist

Description: User can edit playlist

Input: User selection

Output: confirmation of success

R.1.4.4 View Playlist

Description: User can view playlist

Input: User selection

Output: List of playlists

**R.2 Admin**

R.2.1 Manage Albums/Categories

Description: User can manage albums/categories

R.2.1.1 Create Albums

Description: User can add specific songs and form albums.

Input: Song files and details

Output: Success message

R.2.1.2 Delete Album

Description: User can delete album.

Input: Song files and details

Output: Success message

R.2.1.3 Edit Album

Description: User can edit album by adding or deleting songs or its details.

Input: User Selection

Output: Success message

R.2.1.4 View Albums

Description: User can view albums.

Input: User Selection

Output: List of albums

R.2.2 Manage Songs

R.2.2.1 Add songs

Description: User can add the songs with all the details.

Input: Song file and details

Output: Success message

R.2.2.2 Browse songs

Description: A customer can browse through the different genres, categories and can also view the details of the products such as the description, price and songs listing etc. Depending upon the genre/category selected, the contents of the table are accessed using a select query.

Input: User will select one of the genres and its category.

Output: The system will display song list and information of the selected genre and/or category. The song list will be displayed and 10 songs will be displayed on each page and the rest (if any) will be on the next page. This will be executed using the “pagination” property i.e. there will a link named “Previous” and “Next” on the bottom of page to enable the users to go to the next and previous pages to view songs. The current page of the user will also be displayed on every page.

R.2.2.3 Delete Song

Description: User can delete the song

Input : User selection.

Output: confirmation of deletion

**3.3 Other Nonfunctional Requirements**

**1. Performance**

The system must be interactive and must not involve long delays. Though in case of opening the app components or loading the page the system shows the delays less than 2 seconds.

The performance of uploading and downloading the medical records is highly based on internet bandwidth and physical distance from the server.

**2. Safety**

The users’ data is highly personal. The system has authorization to avoid any un-authorized access to user’s private data.

**3. Reliability**

As the system has personal data, its reliability is the major factor for consideration.

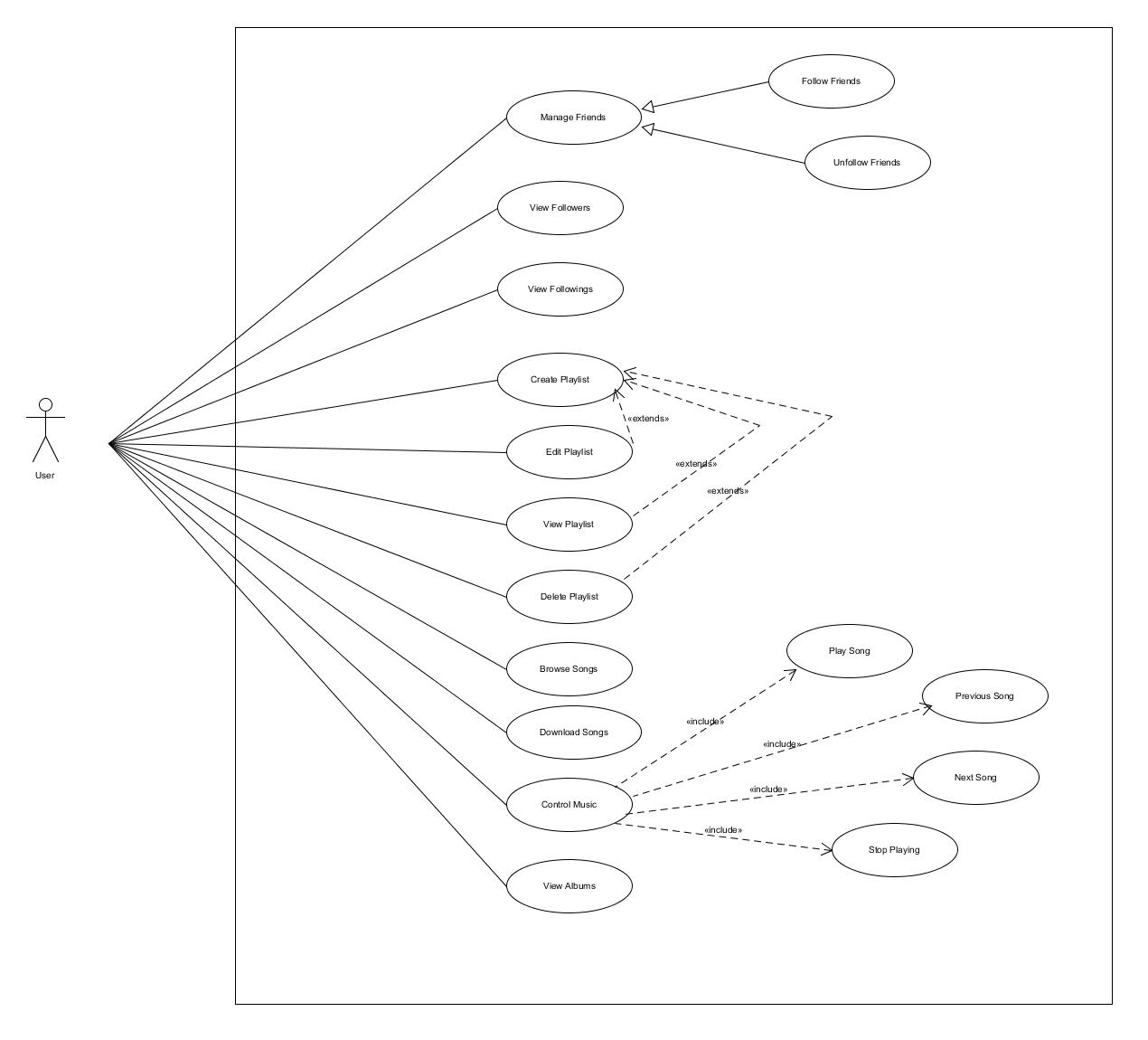
**4. Database**

System requires to access user’s data fast to maintain the performance.

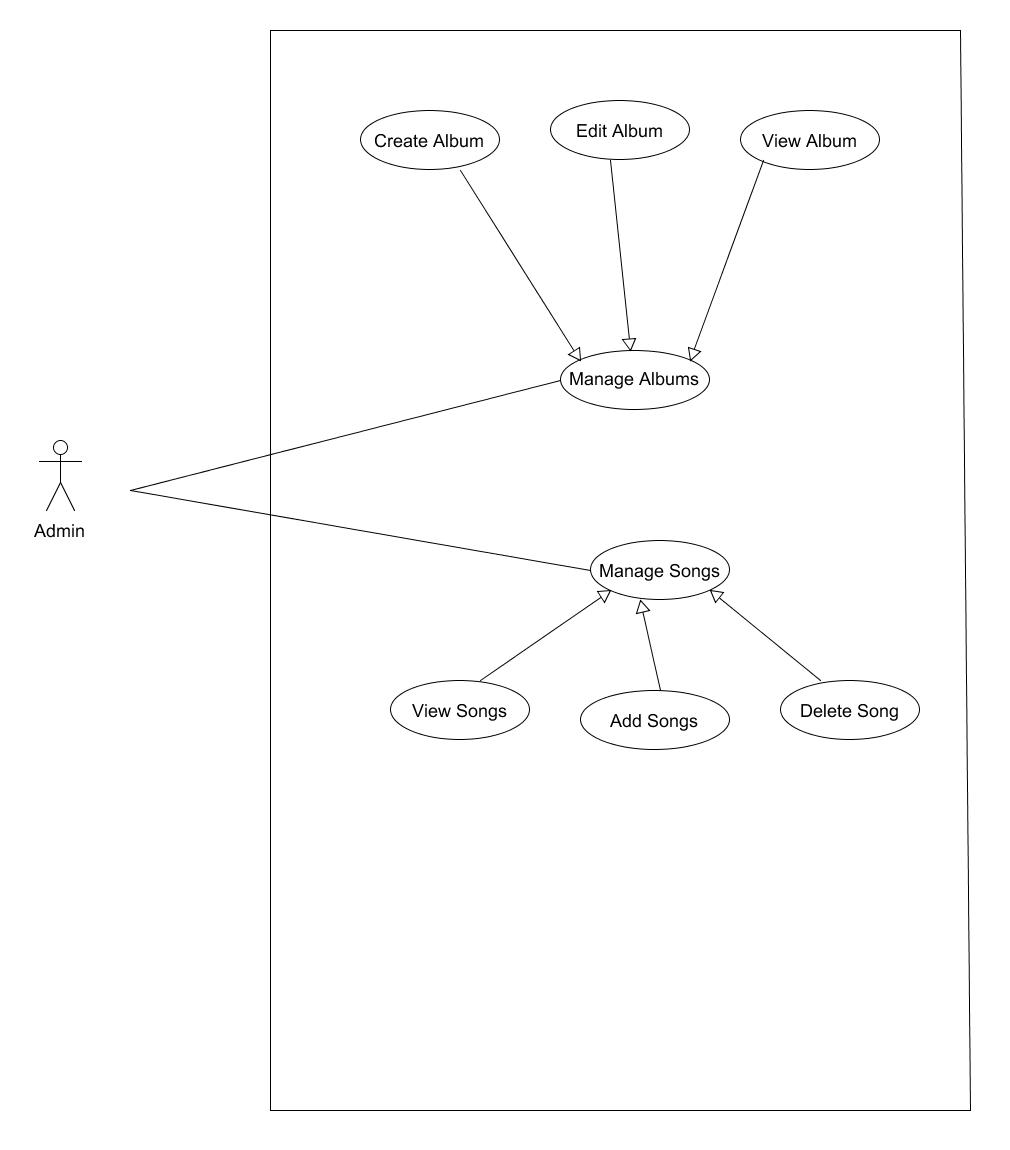
4. Design

**4.1 Use Case Diagram**

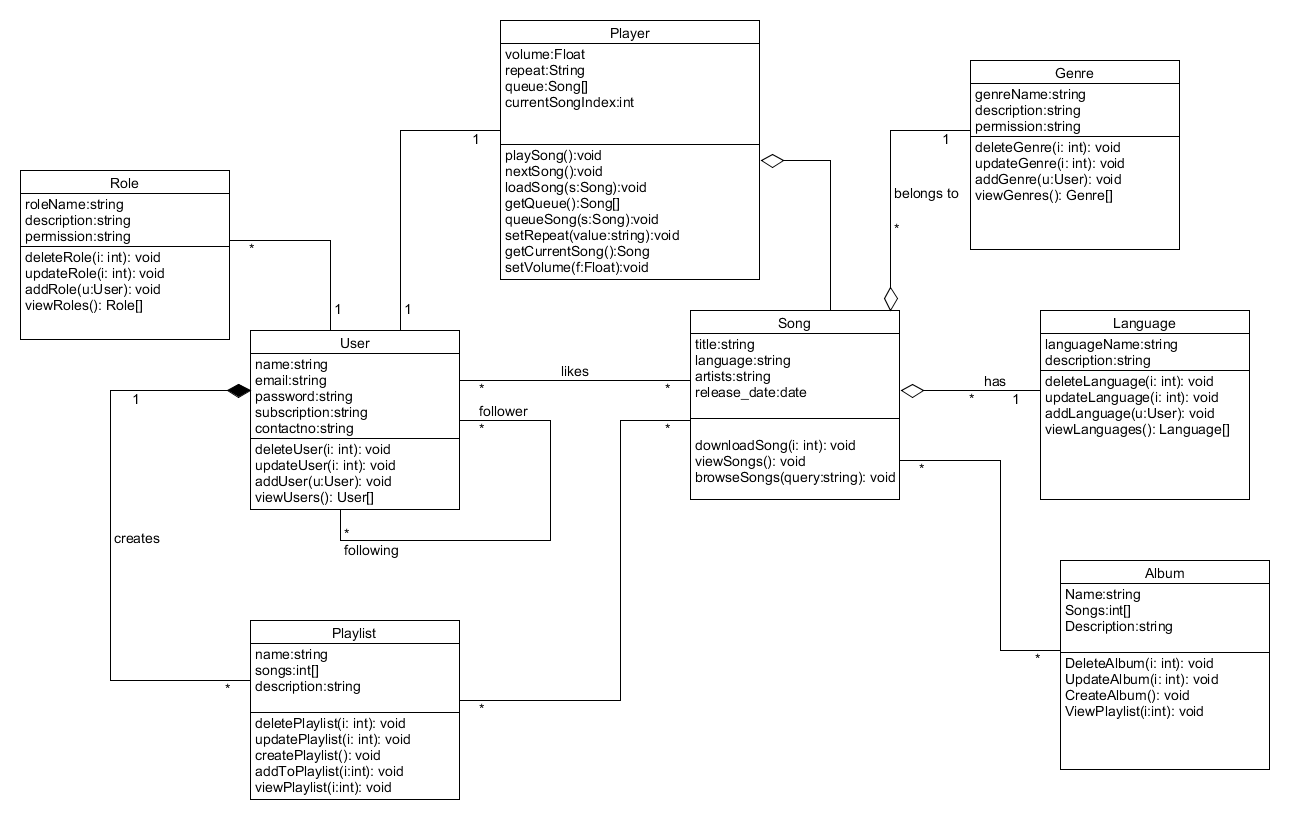
**User**

****

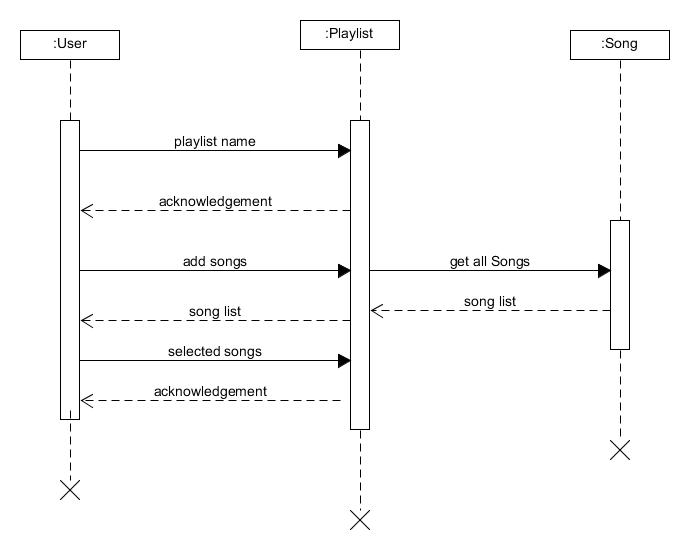
**Admin**

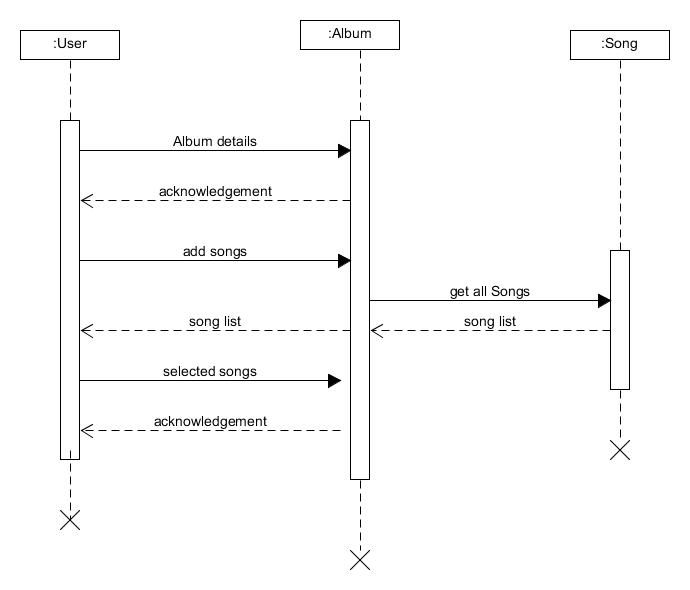
****

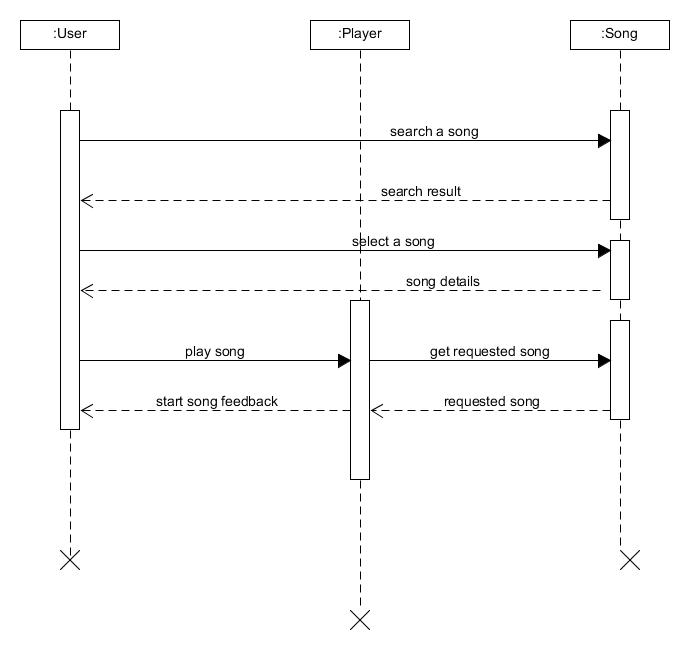
**4.2 Class Diagram**

****

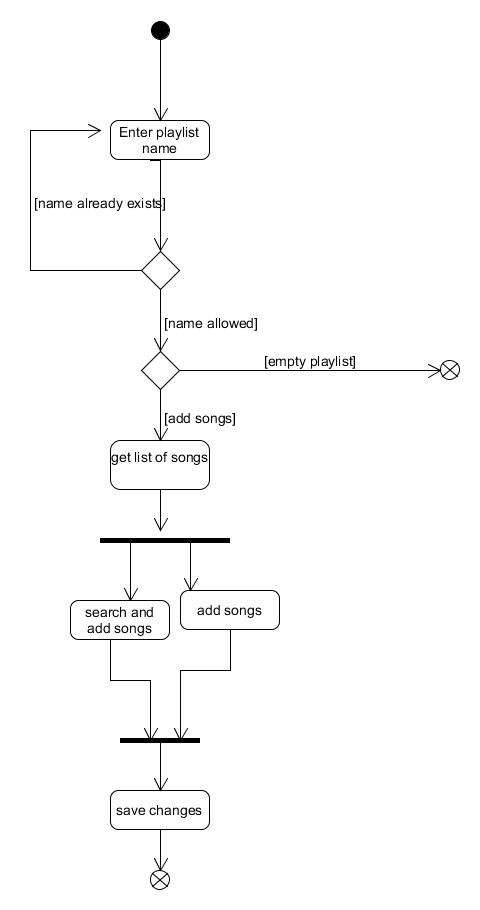
**4.3 Sequence Diagrams**

****

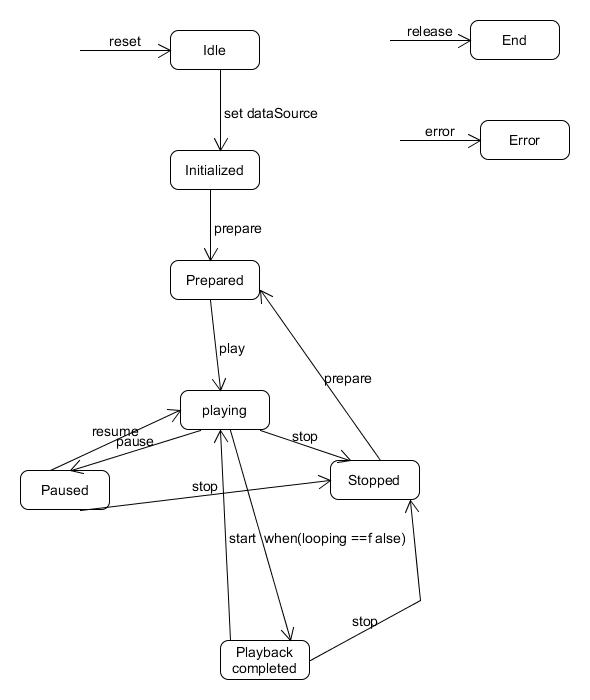
****

****

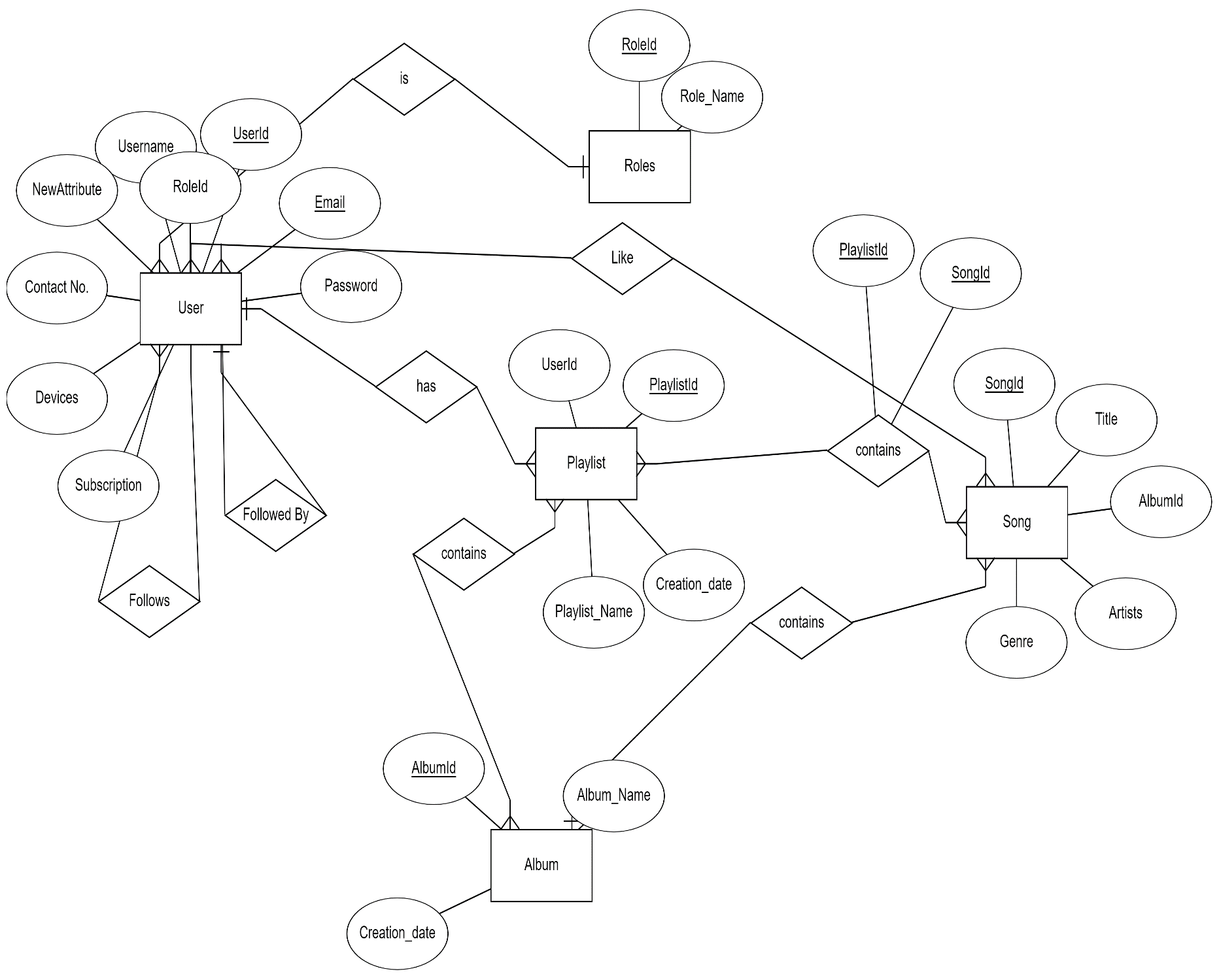
**4.4 Activity Diagram**

****

**4.5 State Diagram**

****

**4.6 E-R Diagram**

****

**4.7 Data Dictionary**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Users | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | UserId | varchar | 24 | Yes | yes | yes | - |  |
| 2 | Username | varchar | 14 | No | no | no | - |  |
| 3 | Email | varchar | 50 | Yes | yes | no | - |  |
| 4 | Password | varchar | 20 | Yes | no | no | - |  |
| 5 | Devices | varchar | int | Yes |  |  |  |  |
| 6 | Subscription | varchar | 30 | Yes | - | - | - | Default=”Free” |
| 7 | Contact No. | varchar | 10 | No | - | - | - | - |
| 8 | RoleId | int | - | Yes | - | FK | Roles | - |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
| Languages | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | LanguageId | varchar | 24 | Yes | yes | yes | - |  |
| 2 | LanguageName | varchar | 14 | Yes | yes | no | - |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
| Language-Users | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | Id | int | - | yes | yes |  |  |  |
| 2 | UserId | int | 24 | yes | No | FK | Users |  |
| 3 | LanguageId | int | 14 | yes | no | FK | Languages |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
| Roles | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | RoleId | varchar | 24 | yes | yes | yes | - |  |
| 2 | RoleName | varchar | 14 | yes | yes | no | - |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
| Songs | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | SongId | int | - | yes | yes | yes | - |  |
| 2 | Title | varchar | 14 | yes | yes | no | - |  |
| 3 | AlbumId | varchar | 50 | no | - | FK | Album |  |
| 4 | GenreId | int | - | no | no | FK | Genre | - |
| 5 | Artists | varchar | 100 | yes | no | - |  |  |
| 6 | LanguageId | int | - | yes | no | FK | Language |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
| Genres | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | GenreId | int | - | yes | yes | yes | - |  |
| 2 | GenreName | varchar | 14 | yes | yes | no | - |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
| Playlists | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | PlaylistId | int |  | yes | yes | yes | - |  |
| 2 | UserId | int |  | yes | no | FK | Users |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Playlist-Songs | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | Id | varchar | 24 | yes | yes | yes | - |  |
| 2 | PlaylistId | int | 14 | yes | no | FK | Playlists |  |
| 3 | SongId | int | 50 | yes | no | FK | Songs |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
| Albums | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | AlbumId | int | - | yes | yes | yes | - |  |
| 2 | AlbumName | varchar | 14 | yes | yes | no | - |  |
| 3 | Creation\_Date | DateTime | - | yes | - | no | - |  |
| 4 | GenreId | int | - | yes | - | FK | Genres |  |
| 5 | LanguageId | int | - | yes | - | FK | Languages |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Followers | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | Id | int |  | yes | yes | yes | - |  |
| 2 | UserId | int |  | yes | no | FK | Users |  |
| 3 | FollowerId | int |  | Yes | no | FK | Users |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Followings | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | Id | int |  | Yes | yes | yes | - |  |
| 2 | UserId | int |  | Yes | no | FK | Users |  |
| 3 | FollowingId | int |  | Yes | no | FK | Users |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Liked Songs | | | | | | | | |
| Sr no. | Field Name | Data Type | Width | Required | Unique | PF/FK | Referred Table | Description |
| 1 | Id | int |  | Yes | yes | yes | - |  |
| 2 | UserId | int |  | Yes | no | FK | Users |  |
| 3 | SongId | int |  | Yes | no | FK | Songs |  |

5. Implementation Details

The system consists of 3 basic modules namely

1. User Module
2. Playlist Module
3. Song Module

Each module consists of several methods to implement the required functionality. Implementation is done using C# language and Windows Forms. Database used in these modules is SQL.

**User Module**

This module is the base for authentication and authorization to ensure the security aspect of the user.

**Playlist Module**

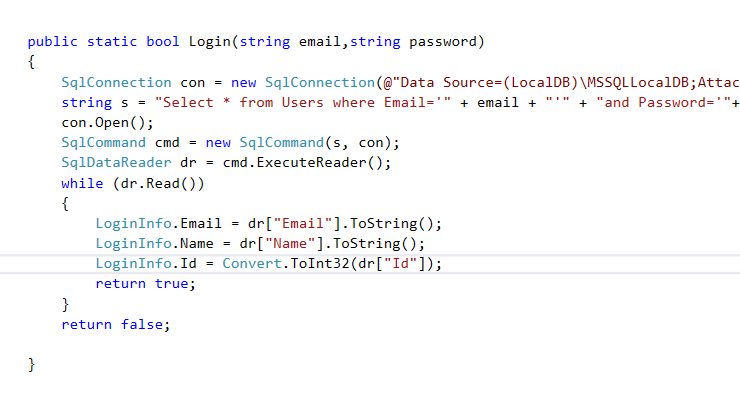
This module handles the functions related to playlists. It allows user to create playlist and add songs to it. It allows to update and delete existing playlists.

**Song Module**

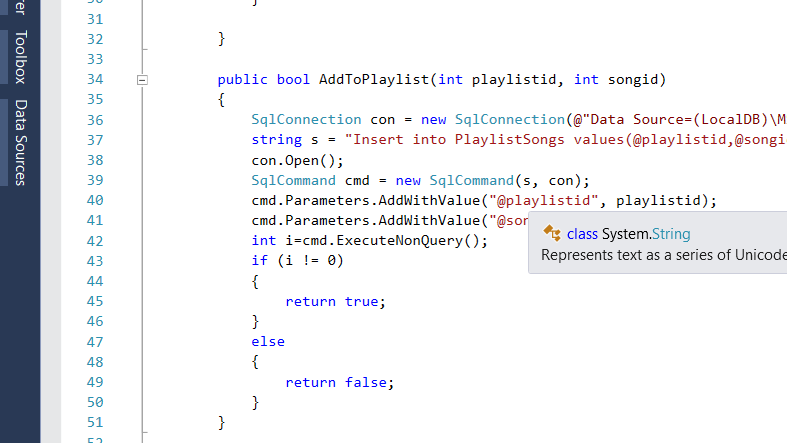
This module handles streaming of music and manages database of songs available in the application.

**5.2 Function prototypes**

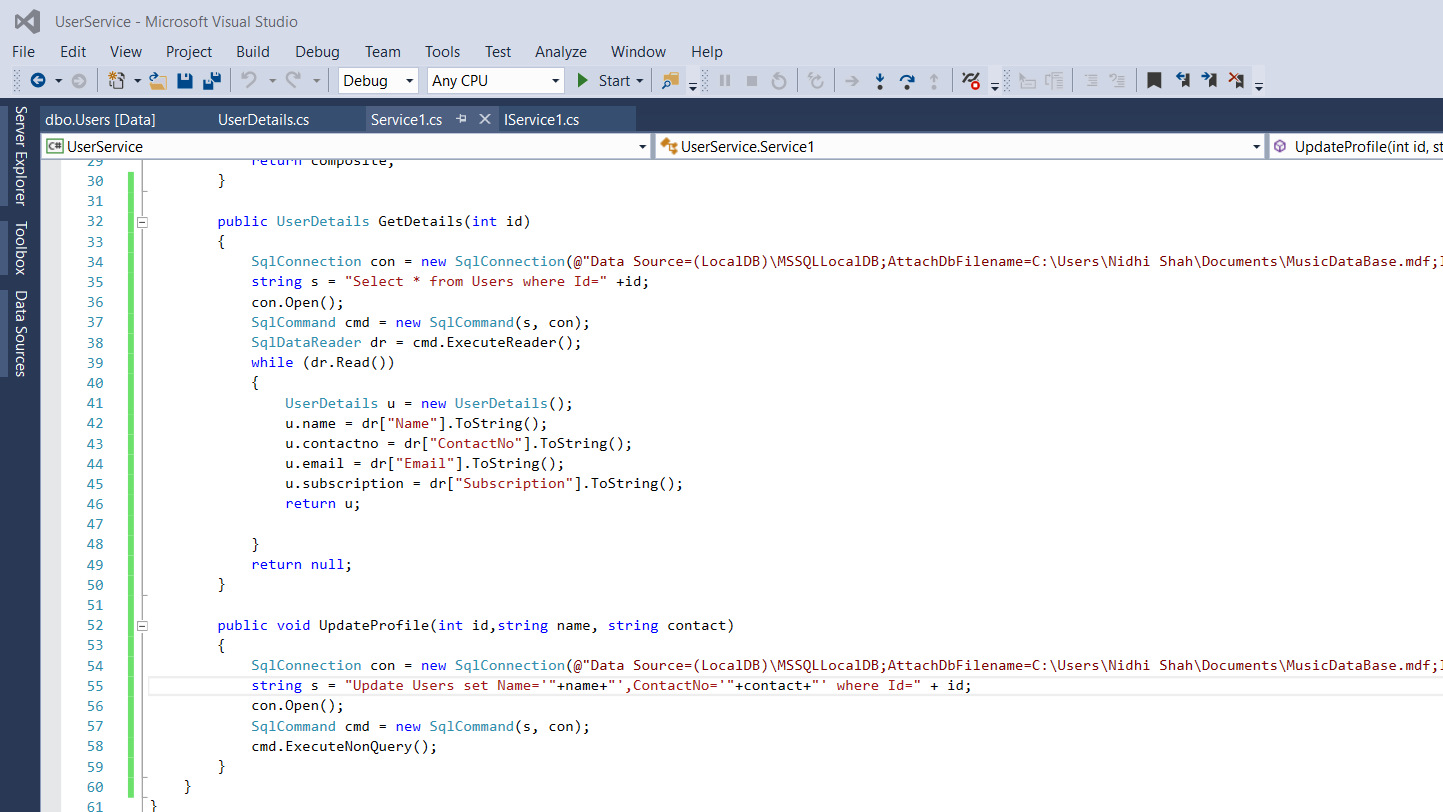
Login



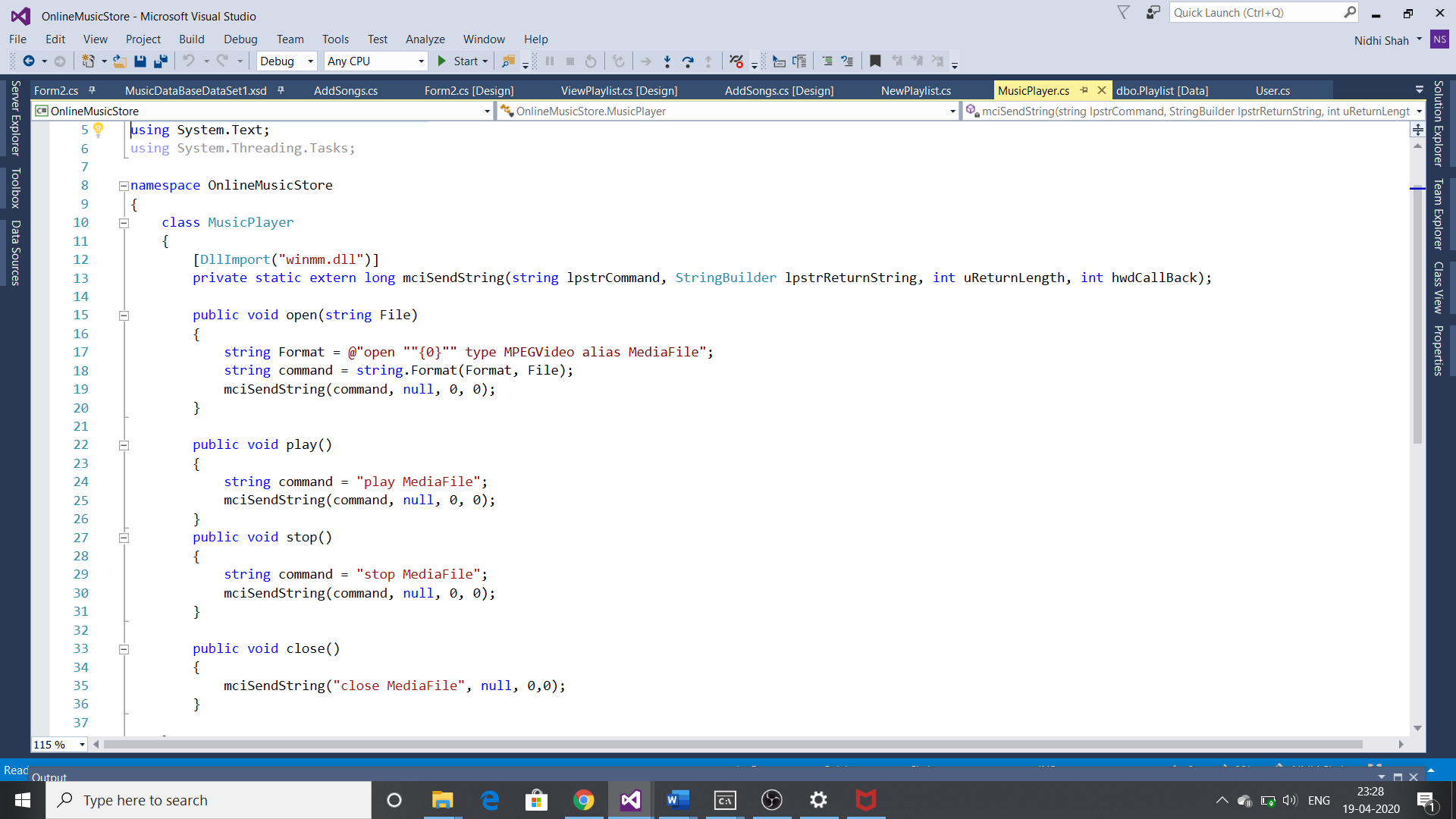
Add To Playlist



Get Details, Update Profile



Music Player

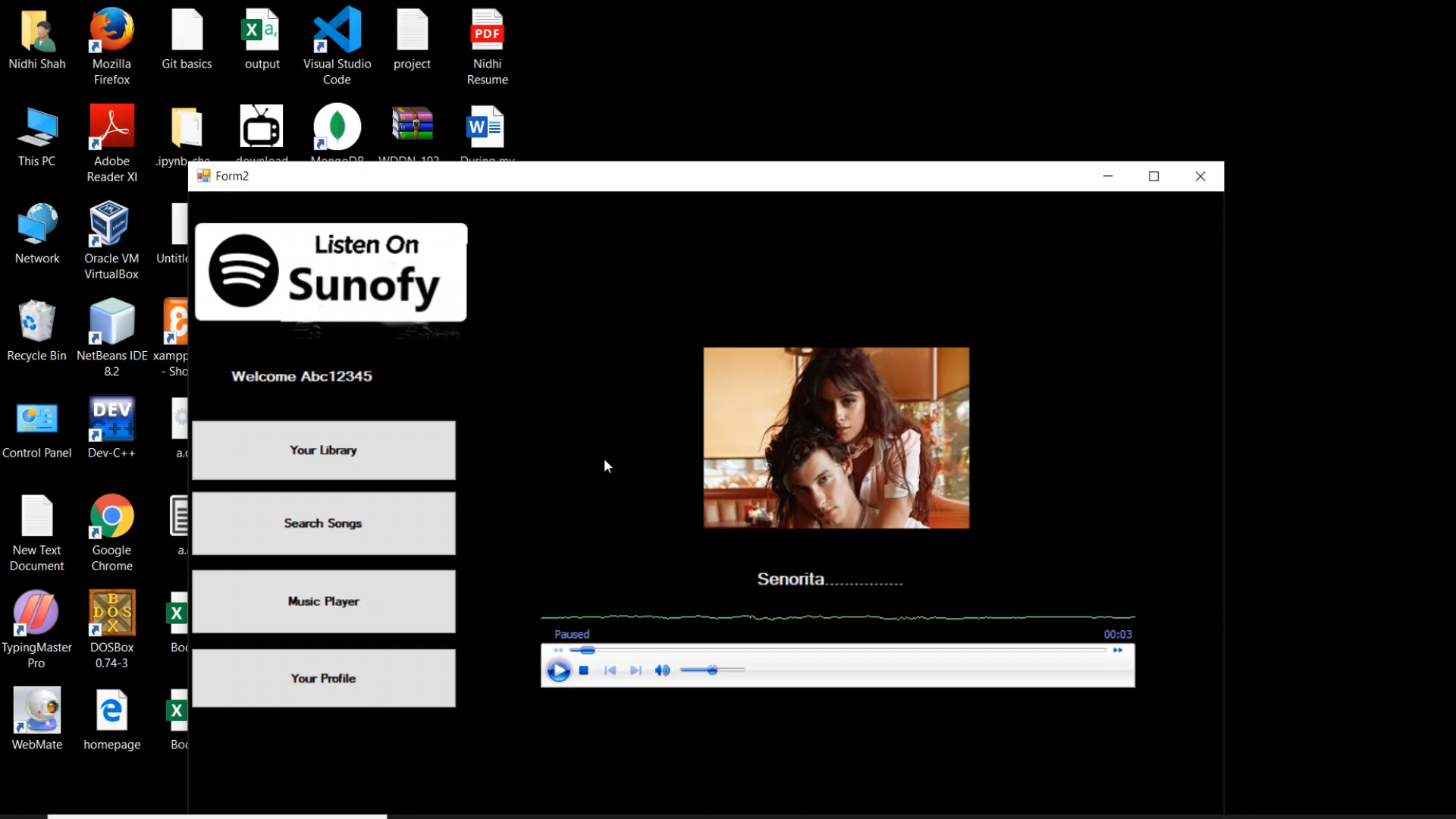


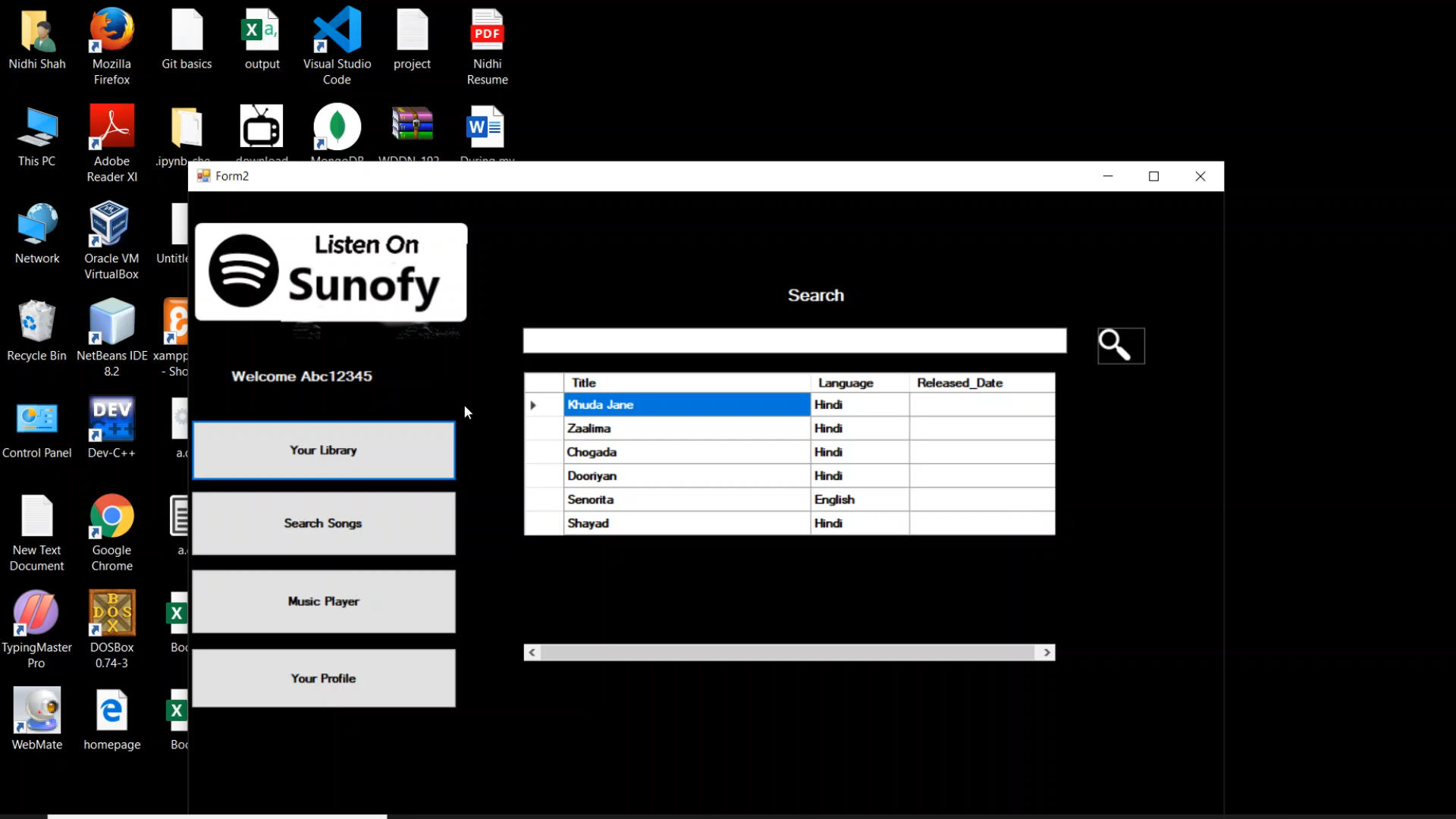
6. Testing

Manual testing was performed in order to find and fix the bugs in development process.

7. Screenshots







8. Conclusion

The functionalities are implemented in system after understanding all the system modules according to the requirements. Functionalities that are successfully implemented in the system are:

* User registration containing all the necessary validation on field
* Login
* User authentication
* Logout
* Playlist creation, deletion and update
* Music player
* Add to playlist
* Update user profile

After the implementation and coding of system, comprehensive testing was performed on the system to determine the errors and possible flaws in the system.

9. Limitations and Future Enhancements

The system has adequate scope for modification in future.

The system can be extended for user to share their playlists with other users, users can view profiles and follow each other.

The system can also be integrated with Artificial Intelligence to generate playlist based on user history and create albums based on user’s mood.

10. Reference / Bibliography

Following links and websites were referred during the development of this project.

stackoverflow.com

medium.com

w3schools.com

github.com

docs.microsoft.com

msdn.microsoft.com