Music management

system

Project

Logo, company name

Description automatically generated

Submitted to

**Mam Atika Islam**

Submitted by

Ibraheem Qasim

42896

Sarmad Murtaza

42904

**School of Computing and Innovation**

Riphah International University, Lahore Campus.

1. **Introduction**

Our project is all about creating a super easy-to-use Music Management System. It's like having your own organized library where you can find, create, and share your favorite tunes hassle-free.

1. **Objective**

We want to make everyone's music experience better by building a system that helps you manage your music easily. Whether you're an artist or just love jamming out, our goal is to create a platform that makes finding and enjoying music a piece of cake.

1. **Problem Description**

**What:** Ever had trouble finding the right song or organizing your playlists? Our project solves exactly that!

**Why:** Because everyone deserves a simple way to manage their music. No more chaos in your playlists or struggling to discover new songs.

**How:** We're creating a system where you can easily create playlists, collaborate with others on music, and get awesome recommendations. Musicians, music fans, and industry folks – everyone gets a share of the music love.

1. **Features**

- Click-and-Play Interface

- One-Click Playlist Creation

- Easy-to-Use Songwriting Tools

- Organize Music with Simple Tags

- Get Song Suggestions Based on Your Tastes

- Personalized User Profiles

- Collaborate with Artists Effortlessly

- Quick Genre Sorting

- Easy Sign-up and Log-in

- Access Anywhere, Anytime

1. **Users**

1. Music Lovers

2. Musicians

3. Artists

4. Anyone who enjoys a good beat!

1. **Requirements**
   1. **Easy Sign-up**
      1. Users should be able to sign up with just an email.
      2. Logging in should be as easy as clicking a button.
   2. **Playlist Magic**
   3. **Tag It Right**
   4. **Collaborative Jam**
      1. Users can invite others to collaborate on playlists.
      2. Real-time collaboration features for songwriting.
2. **References**

No reference added in this document

**Project – Phase 1**

In project phase 1 there are Functional Requirements , Non-Functional Requirements, Constraint Requirements and Domain Requirements

1. 10 Functional Requirements
2. 5 Non-Functional Requirements
3. 3 Constraints Requirements
4. 2 Domain Requirement

**Functional Requirements**

1. Easy Sign-up
2. Playlist Creation
3. Tag It Right
4. Collaborative Jam
5. Smart Recommendations
6. Personalized Profiles
7. Artist Collaboration Hub
8. Quick Genre Sorting

**Functional Requirements Description**

1. Easy Sign-up :

User will able to sign up with just an email and login in will be easy as clicking button.

1. Playlist Magic :

User can create a playlist by clicking at one button and also user can share playlist by sending a link

1. Tag it right :

Each song should be taggable with genre and mood. User can sort songs based on tags

1. Collaborative Jam:

User can invite other to collaborate on playlist and also Real-time collaboration features for songwriting

1. Smart Recommendations :

The system should suggest songs based on user preferences and Recommendations should get smarter with usage

1. Personalized Profiles :

Users can customize their profiles with pictures and bios.And profile show user most played genre

1. Quick Genre Sorting :

Users can easily filter songs by genre and provide icons for quick identification

1. Artist Collaboration Hub :

This fun have Dedicated spaces for artists to connect and collaborate and they can direct message and share file

**Non Functional Requirements :**

1. Lightning Fast
2. Keep It Safe
3. Intuitive User Interface
4. Scalability

**Non Functional Requirements Description :**

1. Lightning Fast :

The system should respond super fast, almost like magic.

1. keep it safe :

Your info should be super secure – like having a lock on your diary

1. Intuitive User Interface :

The system should be easy to navigate, even for first-time users.

1. Scalability :

The system should handle an increasing number of users and songs without slowing down

**Constraint Requirements :**

1. Cost.
2. Time duration during development

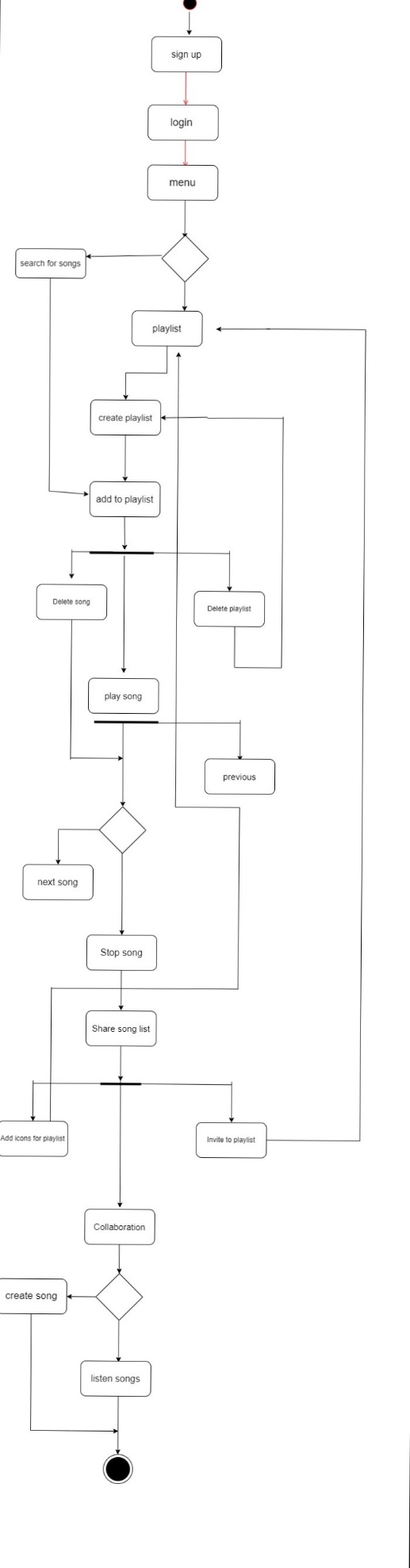
**Domain Requirements :**

Accessibility Standards

**Project – Phase 2**

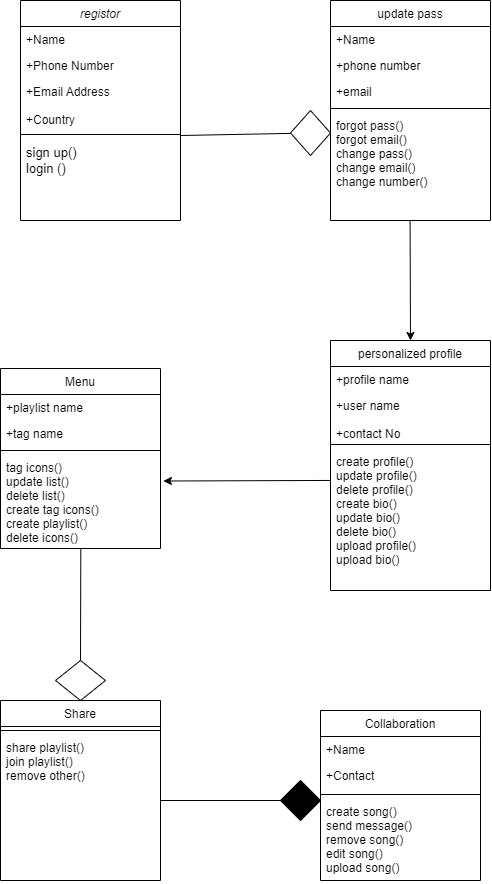
**Activity Diagram:**

The activity diagram of project is given below :

****

**Class Diagram :**

The class diagram of this project is given below.

****

There are total 6 classes in this project where every each one of them are connected to each other somehow. The register class is aggregated with class update pass which means both are combine and working on a some task but both are independent classes.

**Register class :**

* **Attributes**

1. Name : string
2. Phone number : int
3. Email : string
4. Country : string.

* **Methods**

1. Sign up
2. Login

**Update class :**

* **Attributes:**

1. Name : string
2. Phone number : int
3. Email : string

* **Methods**

1. Forgot pass
2. Forgot email.
3. Change pass
4. Change email.
5. Change number

**Personalized profile class :**

* **Attributes:**

1. User Name : string
2. Profile name: string
3. Contact no : int

* **Methods**

1. Create profile
2. Delete profile
3. Update bio
4. Create bio
5. Delete bio
6. Change profile pic
7. Delete profile pic

**Menu class :**

* **Attributes:**

1. playlistName : string
2. tag name : string

* **Methods**

1. Tag icons
2. Update list
3. Create list
4. Delete list
5. Delete icons

**Share class :**

* **Methods:**

1. Share play list
2. Remove play list
3. Join playlist

**Share class :**

* **Attributes:**

1. Name : string
2. Phone number : int

* **Methods:**

1. Create song
2. Remove song
3. Upload song
4. Send message

**User case description**

|  |
| --- |
| **User Case Name:** Music management system |
| **Summary:** Makes the user to easy to understand the music application. The user can listen , create , share and upload song |
| **Participating Actors:** User |
| **Purpose: Music application** |
| **Pre-condition: User must Register** |
| **Post-condition: playlist will be created** |

**Actor and system response:**

|  |  |  |
| --- | --- | --- |
| S# | **Actor action** | **System Response** |
| 1 | Open Application | Show menu |
| 2 | Select search option | Search bar will open |
| 3 | Select any song | Playlist will be created and song will be added |
| 4 | Select any playlist name | Playlist will be added to home menu |
| 5 | Selected share option | Share option will display sharing list |

**Event Flow:**

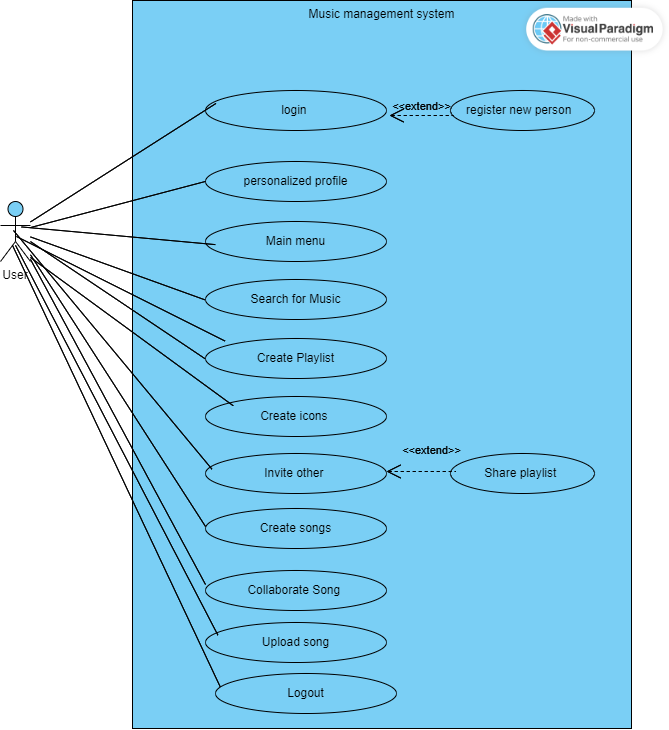
1. the user starts the application or system
2. the application display the first page which is about registration or login
3. the system display the menu
4. the user starts with search for songs.
5. After searching user can add it to playlist or create a new one
6. User can also share his playlist or invite others

**Alternative Flow:**

In case of wrong input the system will go back to step 2 or step 3.

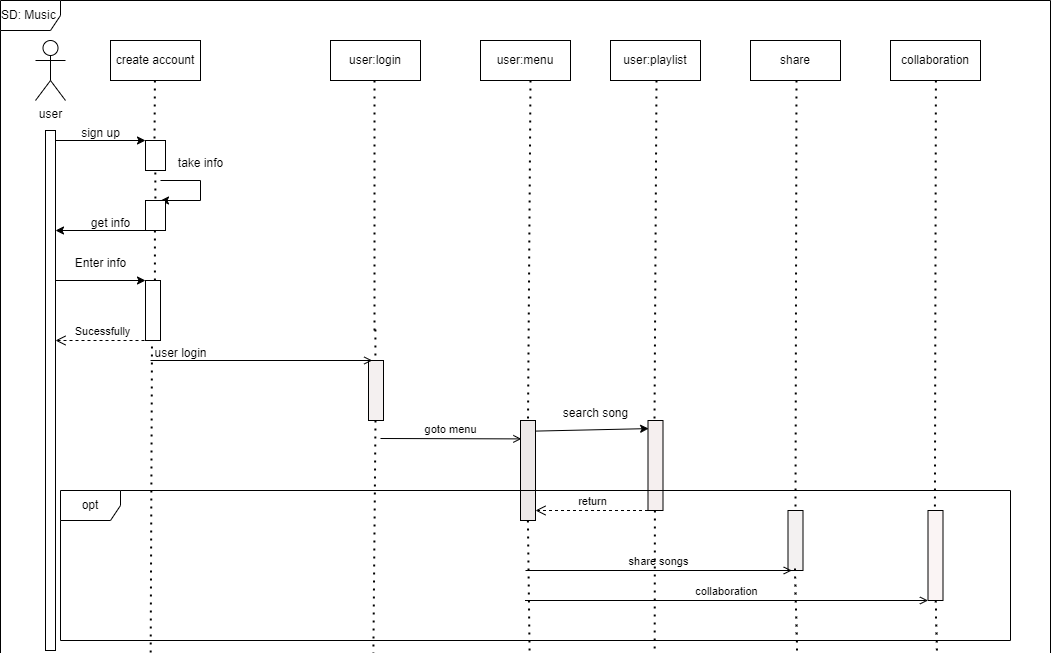
**Use case diagram :**

The use case diagram of this project is given below.

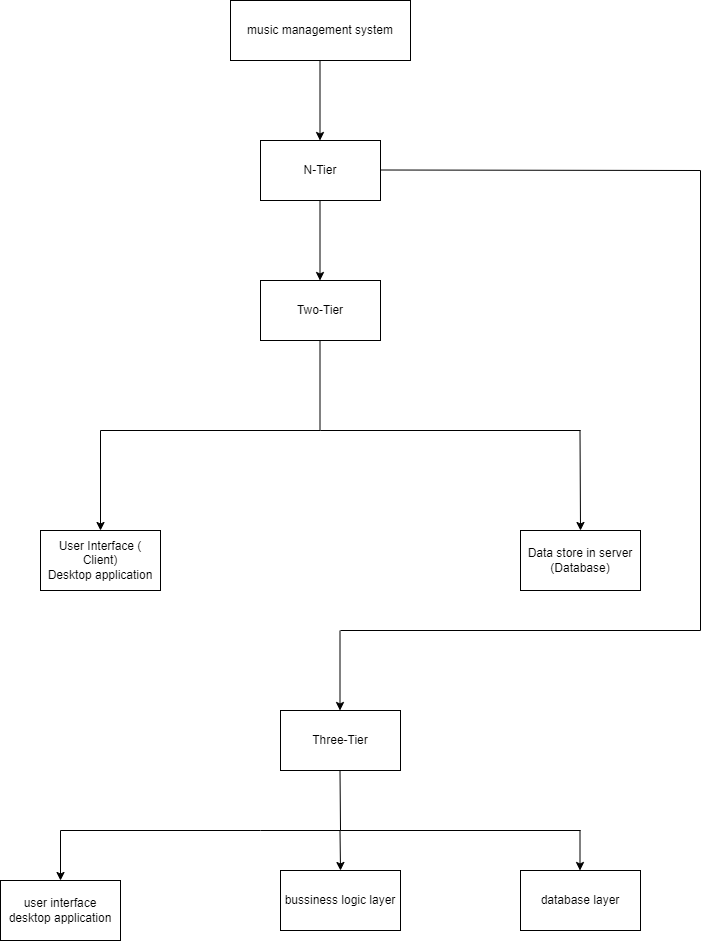


**Sequence diagram :**

the sequence diagram of music management system project is below.



**Architecture diagram :**

****

**Architecture Flow :**

The project is built with the use of two tier and three tier which is the part of n- tier. The two tier consist of two parts the client and database . this project use two tier in a such way that the user will open the application which is the interface and start to search for songs which he like . all the data where the songs are stored or playlist which he will created is stored on a server which is database. Similarly it also use three tier architecture in such way that it consist of a business logic which acts as interface between client and data access layer. All the necessary logic are written in this section.

**Diagram used :**

1. Activity diagram
2. Class diagram
3. Use case description
4. Use case model
5. Sequence diagram
6. Architecture diagram.