Software Requirements Specification – ONLINE MUSIC PODCAST

Author(s): Harish B

Version: 0.2

Date: November 11, 2024

Contents

1. Objective and Scope………………………………………………………………………………3

2. Features..3

2.1 Log-in/Sign-up module..3

2.2 Profile module...3

2.3 Premium module..4

3. Functional Requirements..4

3.1 Log-in/Sign-up...4

3.2 Homepage..4

3.3 Payment...4

4. Non-Functional Requirements...5

4.1 Portability...5

4.2 Maintainability...5

5. High Level design..5

6. Low Level design...5

7. UML Diagrams..5

7.1 Use case Diagram...5

7.2 Sequence Diagram...7

7.3 Class Diagram...9

7.4 Flow Chart...10

7.5 Entity Relationship Diagram...12

1. Objective and Scope

The objective of the project is to build a music website that must be capable of storing offline music could play at any time and by also using internet connection could search effectively the list of the updated songs can enjoy those songs and make use of it.

The scope of the project includes the following basic features:

- In olden days, music is available only in the form of CD and mostly in the streaming of Radio's pre- defined playlists which must feel the user they want to play it with only a DVD Player and it's a very tragedy because of the portable issues with the DVD Player.
- As time changes which now it's become more easier to hear the songs not with a DVD Player, Radio or any special device that only access music only but in simple we can hear and enjoy our favourite music in headphones itself and upgraded version of the headphones.
- Users can create playlists of their own choices can add various genres of songs into that playlist so that he/she can enjoy the vibe on his/her own selection and also, he/she can upload the songs or podcasts in the application.

2. Features

2.1 Log-in / Sign-up module

Sign-up: There should be sign-up for registering the user's details like their Name, Mail-Id and creating a password for their security purpose that except from the user no one will be able to use the account.

Log-in: Then after the sign-up the user must login to use their account and here also the user must give the details of his/her name and passwords which they used to register their account both credentials are matched then the user may use their accounts with ease.

2.2 Profile module

Here the user can view his/her profile and also able to upload their songs or podcasts which had been created on their own.

Also, in addition to that here the user can create, edit and delete the playlist.

2.3 Premium module

To get the premium memberships and access to maintain their pages (so far uploaded songs or podcasts views and profits).

To access the premium membership the user needs to spend money to buy the membership and it is followed by the payment process where the user can find different number of payment options like (UPI ID, Bank Transfer, Credit Card, Debit Card and so on).

3. Functional Requirements

3.1 Log-in/Sign-up

Here the user needs to register the details and also use same details while logging into their accounts.

To secure one's account it's necessary to set a strong password which contains alpha, numeric, special characters. Once these are included in one's password it makes the account very secure and no one can able to access the account except the user.

3.2 Homepage

After logging into their account only the user can be able to listen to the songs otherwise the Homepage can't be viewed by the user.

And, the user needs to leave the application means he/she can log out their accounts in the homepage itself no need to navigate into another page.

3.3 Payment

It is to provide a secure money transferring process during the user buys any memberships because they are using UPI ID, Credit and Debit Cards.

4. Non-Functional Requirements

4.1 Portability

The portability of an application is necessary to ease it to operate anywhere. So, while developing an application a developer must be able to develop the code very easily to be operated by the user at any instant.

4.2 Maintainability

Maintainability of the application by the developer is an important task because if the developer does not maintain the developed code from the admin side there is any crash in the server the client could not able to resolve it only the developer have the access to solve the crash, so it is necessary to maintain it.

5. High Level Design

Design plays an important role before developing a code, it’s like a framework within all the constraints our code should be able to develop that application. Here High-Level Design looks like an overall system architecture contains all design modules within it.

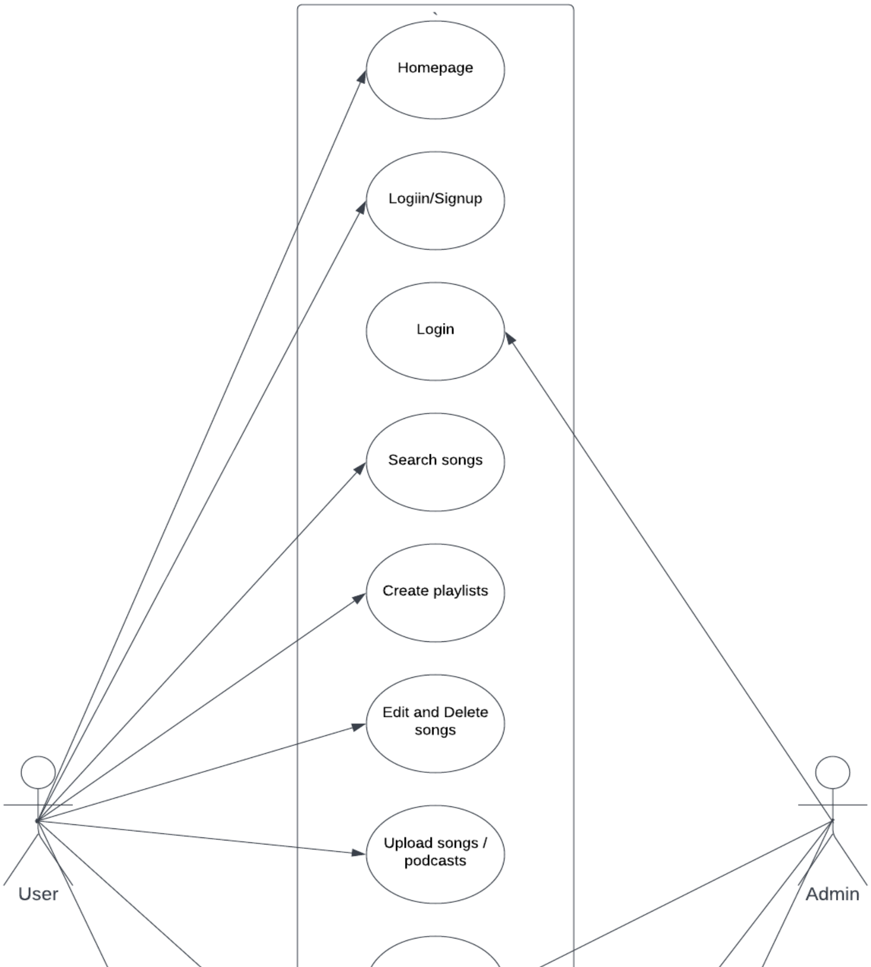
6. Low Level Design

Low level Design is a component level design process, and it is a detailed architecture of design that follows a step –by –step process refinement process. It deals with the planning, coding and execution of the various components, modules and steps in the HLD, at an individual level.

7. UML Diagrams

7.1 Use Case Diagram

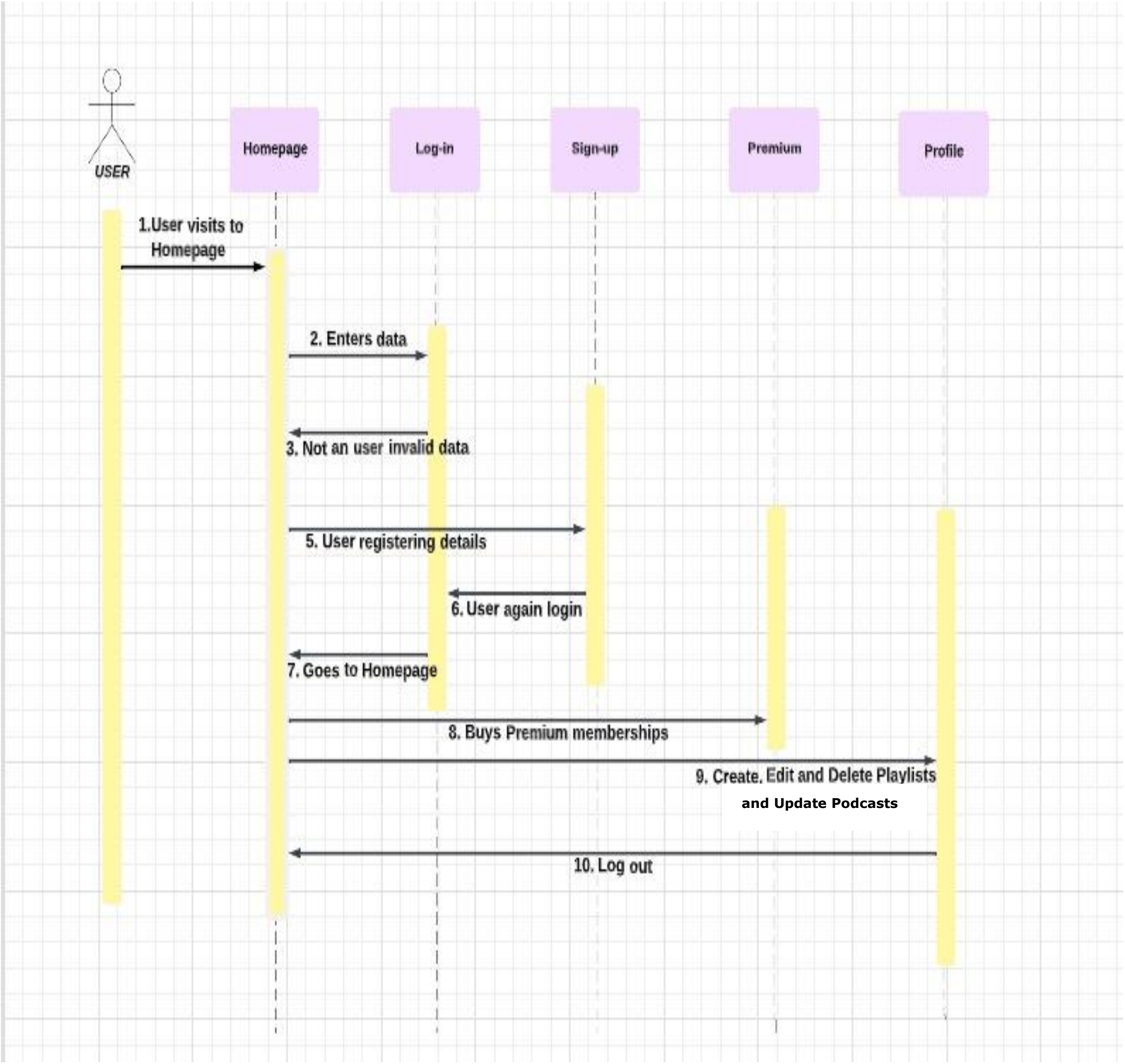
Use case diagrams give a graphic overview of the actors involved in a system, different functions needed by those actors and how these different functions interact. Use-case diagrams describe the high-level functions and scope of a system.


Homepage
Logiin/Signup
Login
Search songs
Create playlists
Edit and Delete songs
Upload songs / podcasts
User
Admin

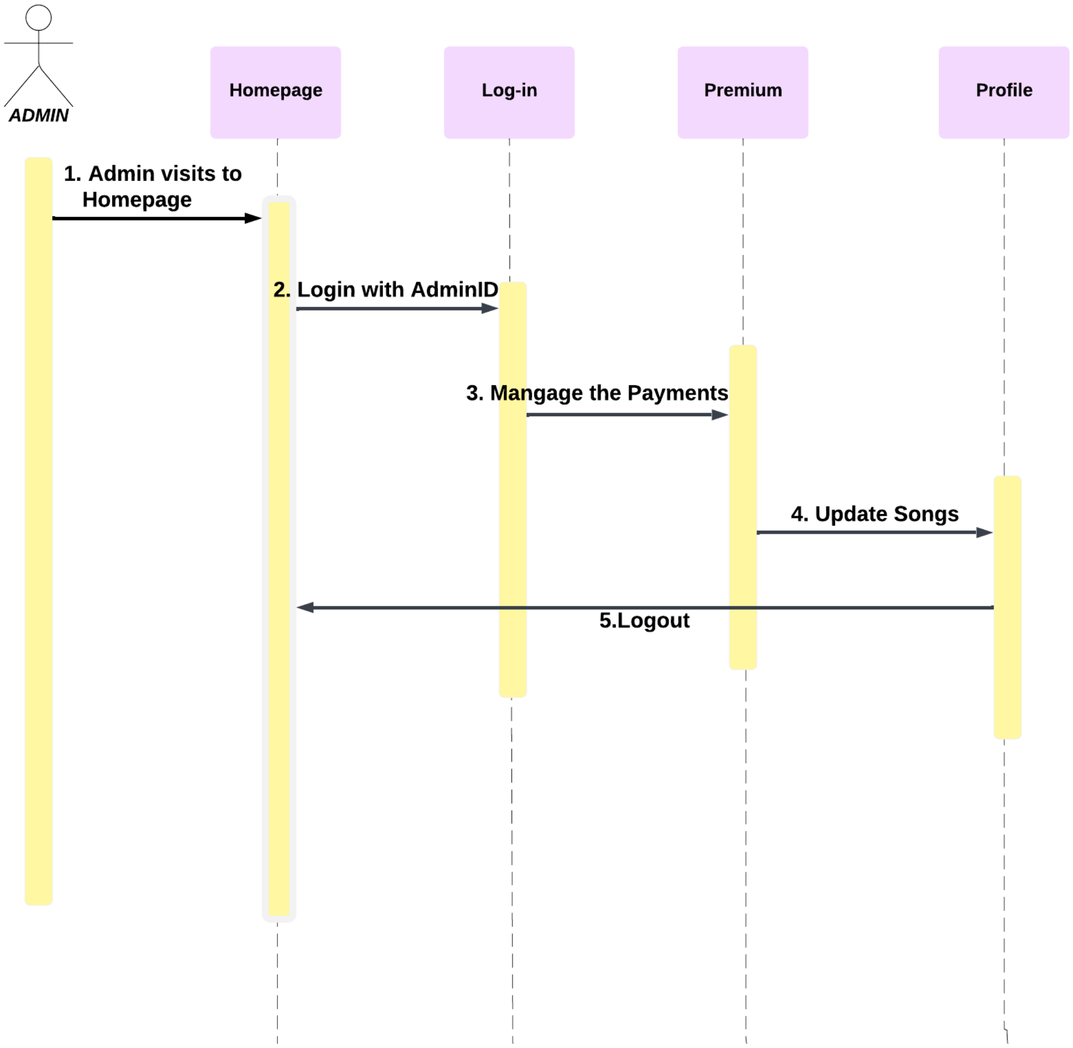
7.2 Sequence Diagram

A sequence diagram is a Unified Modeling Language (UML) diagram that illustrates the sequence of messages between objects in an interaction. A sequence diagram consists of a group of objects that are represented by lifelines, and the messages that they exchange over time during the interaction.

User Side Sequence Diagram:

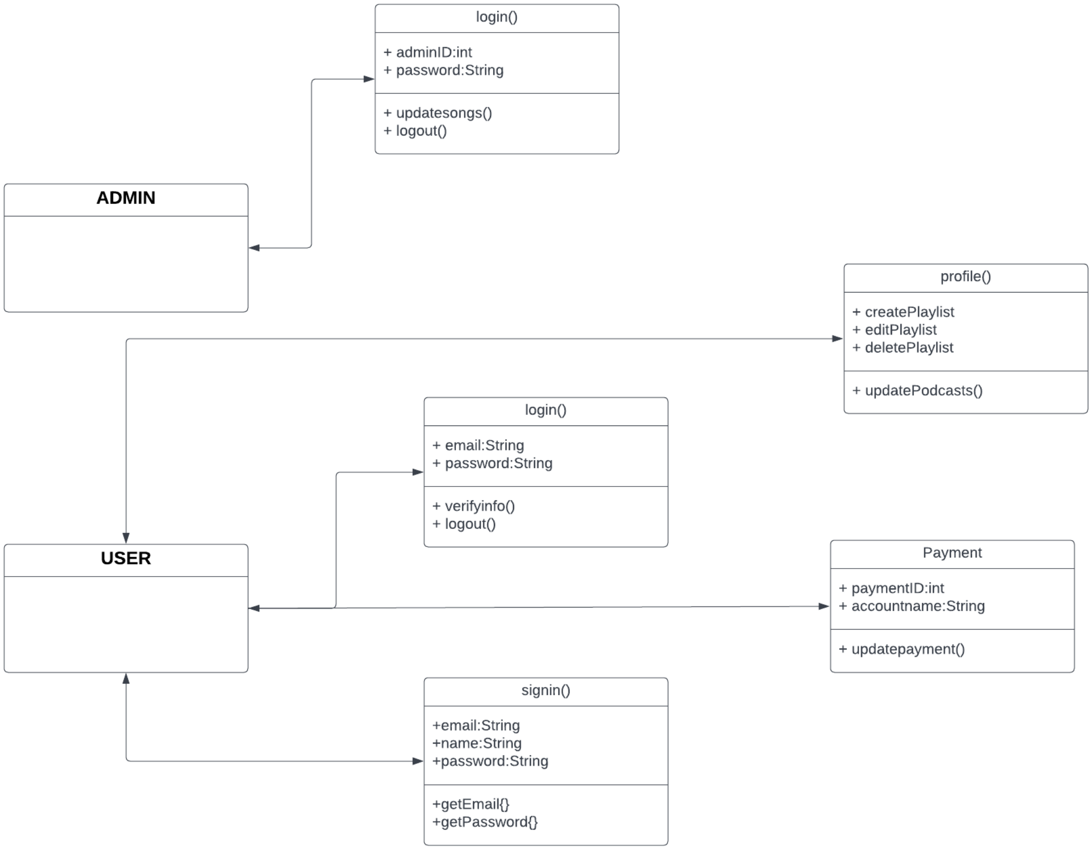

USER
Homepage
Log-in
Sign-up
Premium
Profile
1.User visits to Homepage
2. Enters data
3. Not an user invalid data
5. User registering details
6. User again login
7. Goes to Homepage
8. Buys Premium memberships
9. Create, Edit and Delete Playlists
and Update Podcasts
10. Log out

Admin Side Sequence Diagram:



7.3 Class Diagram

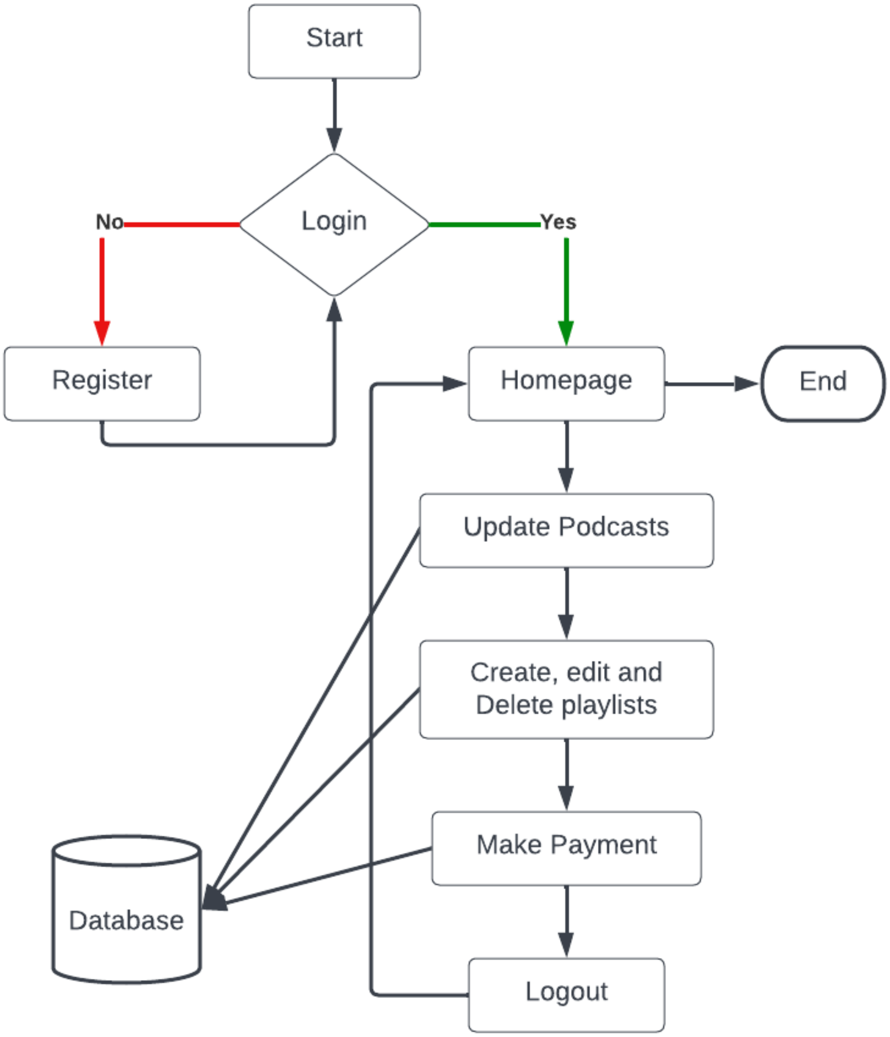
Class diagrams are the blueprints of your system or subsystem. You can use class diagrams to model the objects that make up the system, to display the relationships between the objects, and to describe what those objects do and the services that they provide. Class diagrams are useful in many stages of system design.



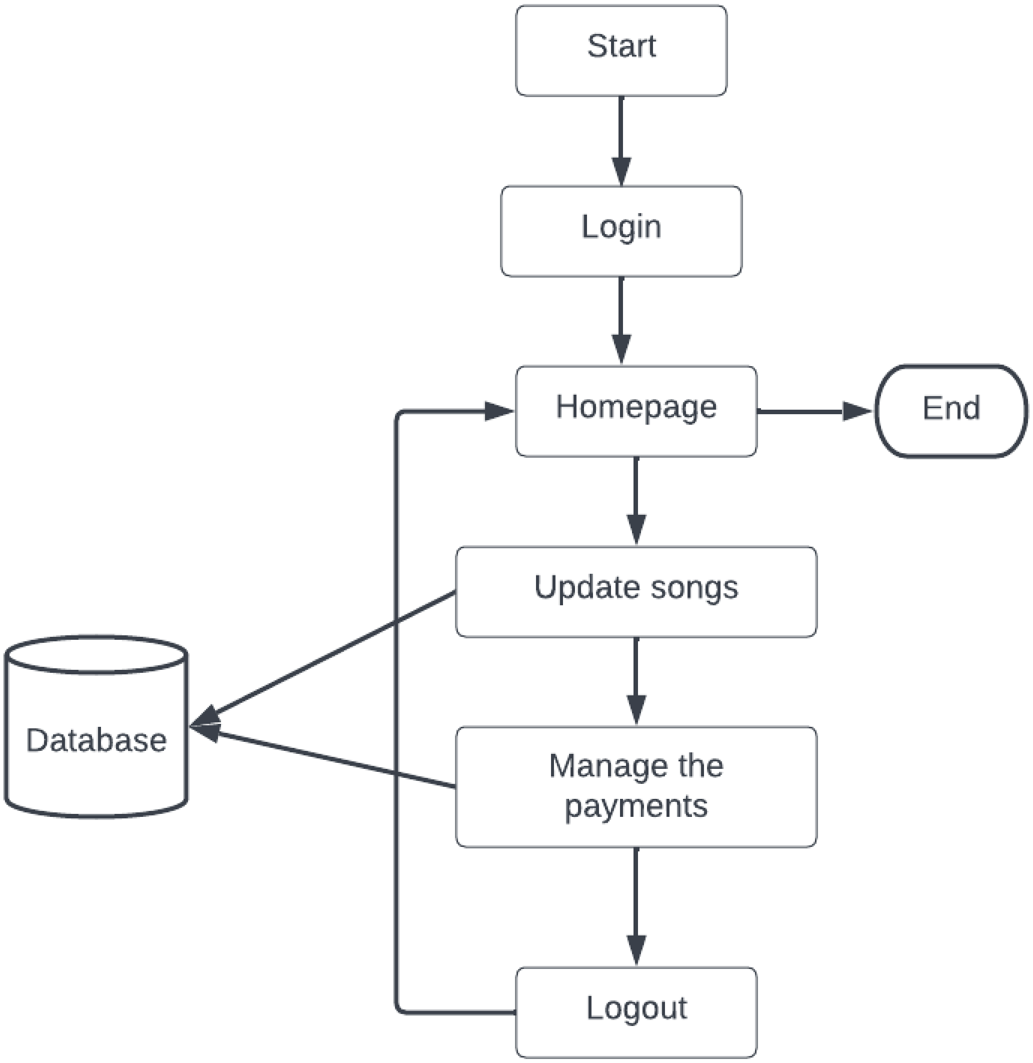
7.4 Flow Chart

A flowchart is a picture of the separate steps of a process in sequential order. It is a generic tool that can be adapted for a wide variety of purposes, and can be used to describe various processes, such as a manufacturing process, an administrative or service process, or a project plan.

User Side Flow Chart:

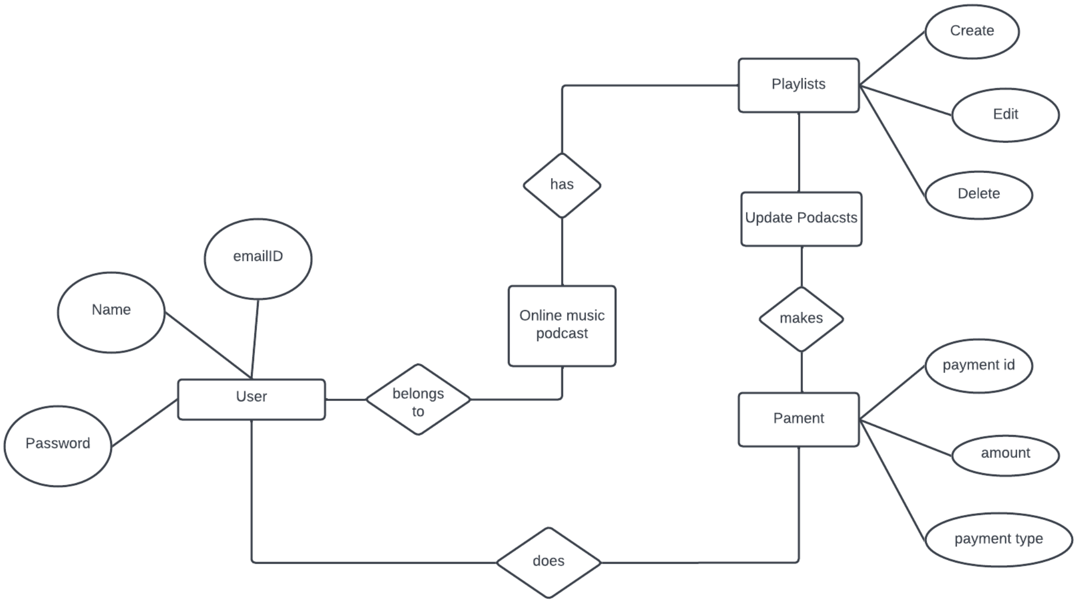


Admin Side Flow Chart:



7.5 Entity Relationship Diagram

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation that depicts relationships among people, objects, places, concepts or events within an information technology (IT) system.


Create
Playlists
Edit
has
Delete
Update Podacsts
emailID
Name
Online music podcast
makes
payment id
User
belongs to
Pament
Password
amount
payment type
does