Department of Computer Science

Software Project Proposal

Course	OBJECT ORIENTED PROGRAMMING 2	**Group No.**	

Student ID	Name	Section
22-46280-1	Maeed Ahammed	N
21-44853-2	AMIT HASAN RUHIN	N
22-49544-3	MD RIZVI KARIM OHY	N
21-45224-2	MD. HIZBUL BASHAR SHOURAV	N

Project Title:

Music Player (HARMONY)

Project Description:

A central focus for your listening experience is provided by an easy-to-use music player system with a Windows Forms user interface, designed with C# and OOP concepts. Using the built-in search and discovery tools, you may import and arrange your music collection or find new tunes. Make your own playlists to suit any mood, then take charge with simple playback controls like volume, pause, and play. Use an equalization to adjust the sound and customize the UI to your taste. With the help of this music player, you can organize and control your whole audio library in one location.

Project Requirements:	
• Entity Classes	• n-Tier Architecture (optional)
• At least 2 types of Users	• Separate Login Table
• Database CRUD operations	• Implement OOP Principles
• All the Forms MUST be connected.	• Application Layer (Form Design)
• Database Connection Class	• One Complete Repository.

- Mention all the **Entities** and **attributes for each Entity** clearly in your Project proposal.
- Specify how many roles there are in your system and what are their individual functionalities.
- You must provide your Project Proposal following the format given below.
- Use only the first page of this document as a cover page for your Project proposal.
- **Entity Classes:** The system will define classes representing real-world entities like User, Playlist, Song, Artist, Album etc. These classes will encapsulate data and provide methods for manipulation.

REQUIREMENTS:

- **User Types:** The system will support at least two types of users: Artist, Listener.
- **Separate Login Table:** A dedicated table will store user credentials (username, password) for secure login functionality.
- **Database CRUD Operations:** The application will implement functionalities to Create, Read, Update, and Delete data within the connected database.
- **OOP Principles:** The development will adhere to OOP principles like Encapsulation, Inheritance, Polymorphism, and Abstraction.
- **Form Connectivity:** All forms within the application will be seamlessly connected to allow users to navigate and interact with different functionalities.
- **Application Layer (Form Design):** Visually appealing and user-friendly Windows Forms will be designed to provide a smooth user experience.
- **Database Connection Class:** A dedicated class will handle database connection and interaction, simplifying data access.
- **One Complete Repository:** The system will implement a repository pattern to manage data access logic for one specific entity demonstrating this design pattern.

REQUIREMENT SPECIFICATION:

- User Stories

Use Case Name	Actor	User Story
Admin	Admin	As an Admin, I want to manage users in the full application, so that I can ensure the security and proper functioning of the system.
Register User	Artist, Listener, Visitor	The registered user wants to create a new playlist to organize their music collection.
Premium User	Artist, Listener	The premium user wants to access high-quality audio streaming within the music player system.
Artist	Artist	The artist wants to upload their music content to the music player system for distribution and streaming.
Search For Music	Admin, Register User, Premium User	The registered user wants to find specific songs, artists, or albums within the music player system's library.
Download Songs	Admin, Register User, Premium User, Artists	The user wants to download songs for offline playback within the music player system
Create Playlist	Register User, Premium User, Artists	The registered user wants to create a new playlist to organize their music collection.
Play Offline	Admin, Register User, Premium User, Artists	The user wants to listen to their music library even without an internet connection
Login and Logout from the system	Admin, Register User, Premium User, Artists	The user wants to securely access their account within the music player system and log out when finished.
Update profile	Admin, Register User, Premium User, Artists	The registered user wants to modify their account information within the music player system.
Manage User	Admin	The administrator wants to manage user accounts within the music player system, ensuring a healthy user base and system functionality.
Manage Subscription	Admin, Register User, Premium User, Artists	The registered user wants to manage their subscription plan within the music player system. This use case can be adapted for Admins managing system-wide plans as well.
Manage Full Application	Admin	I want to manage users in the full application, so that I can ensure the security and proper functioning of the system.
Alter Account Details	Admin, Register User, Premium User, Artists	The registered user wants to modify their account information within the music player
Upload Music	Admin, Artists	The artist wants to upload their music content to the music player system for distribution and streaming.
Manage Music	Admin, Register User, Premium User, Artists	Manages their uploaded music. Manages their music library within the music player system

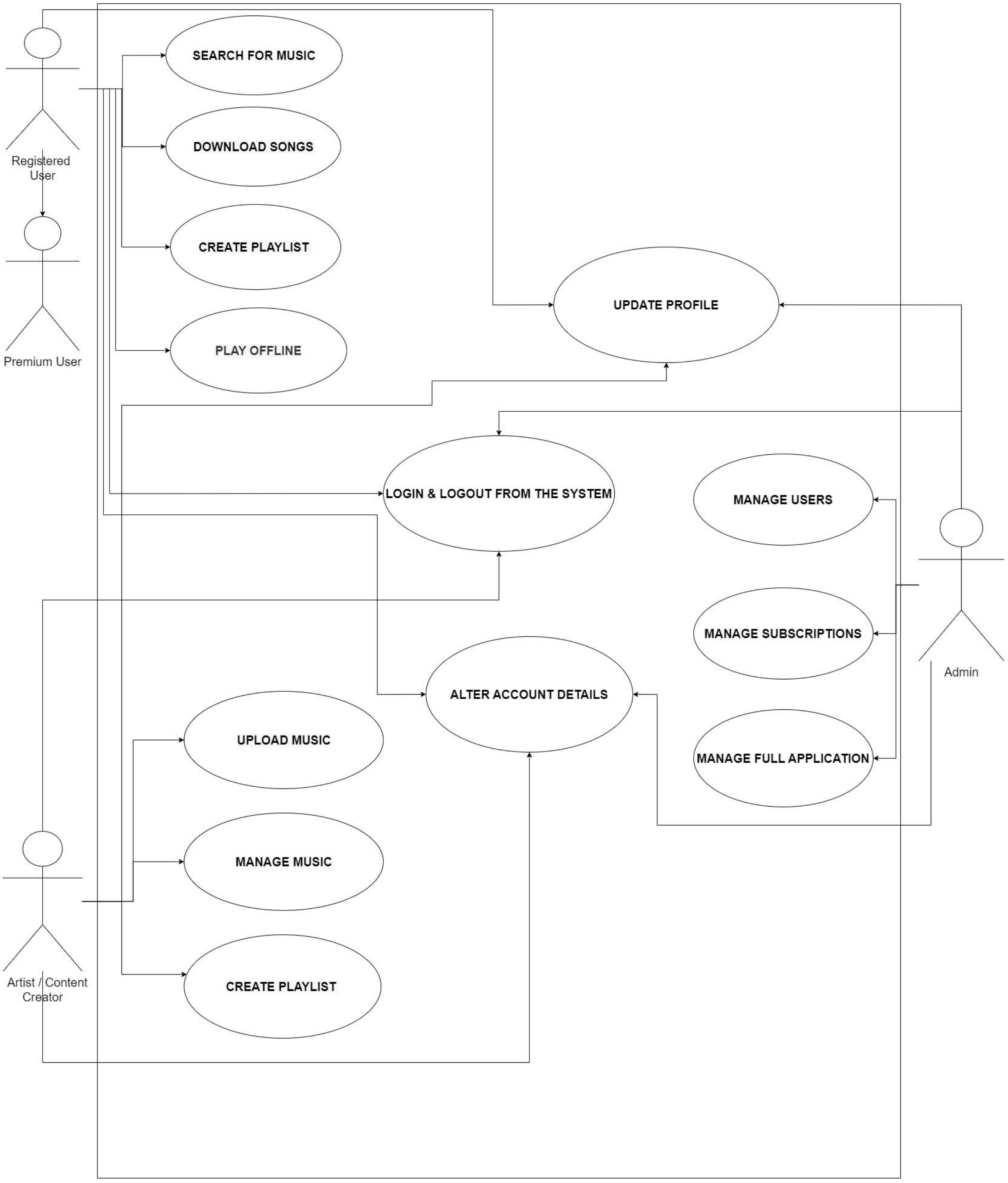
DATA DICTIONARY:

Key	Name	Data Type	Length	Nullable
USER				
PRIMARY	U_ID	INTEGER	10	NO
	U_NAME	VARCHAR	30	NO
	U_DATEOFBIRTH	VARCHAR	30	NO
	U_EMAIL	VARCHAR	20	NO
	U_PASSWORD	VARCHAR	20	NO
PLAYLIST				
PRIMARY	PL_ID	INTEGER	10	NO
	PL_TITLE	VARCHAR	30	NO
	PL_CREATIONDATE	VARCHAR	30	NO
	PL_CATEGORY	VARCHAR	30	NO
FOREIGN	U_ID	INTEGER	10	NO
SONG				
PRIMARY	S_ID	INTEGER	10	NO
	S_TITLE	VARCHAR	20	NO
	S_GENRE	VARCHAR	30	NO
	S_RELEASEDATE	VARCHAR	20	NO
	S_DURATION	VARCHAR	50	NO
FOREIGN	AL_ID	INTEGER	10	NO
ARTIST				
PRIMARY	A_ID	INTEGER	10	NO
	A_NAME	VARCHAR	20	NO
	A_GENRE	VARCHAR	20	NO
	A_BIOGRAPHY	VARCHAR	50	NO
ALBUM				
PRIMARY	AL_ID	INTEGER	10	NO
	AL_NAME	VARCHAR	20	NO
	AL_GENRE	VARCHAR	50	YES
	AL_CATEGORY	VARCHAR	20	NO
	AL_RELEASEDATE	VARCHAR	20	NO
FOREIGN	A_ID	INTEGER	10	NO
LOGIN				
PRIMARY	L_ID	INTEGER	10	NO
	L_USERNAME	VARCHAR	30	NO
	L_PASSWORD	VARCHAR	30	NO
	L_ROLE	VARCHAR	30	NO
FOREIGN	U_ID	INTEGER	10	NO
PREMIUM				
FOREIGN	PAYMENT_ID	INTEGER	10	NO
	P_AMOUNT	INTEGER	10	NO
	DATE	VARCHAR	30	NO
	PROMOCODE	VARCHAR	30	YES
PAYMENT				
PRIMARY	PAYMENT_ID	INTEGER	10	NO
	P_PAYMENT_TYPE	VARCHAR	30	NO
SONGS_IN_PLAYLIST				
FOREIGN	S_ID	INTEGER	10	NO
FOREIGN	PL_ID	INTEGER	10	NO

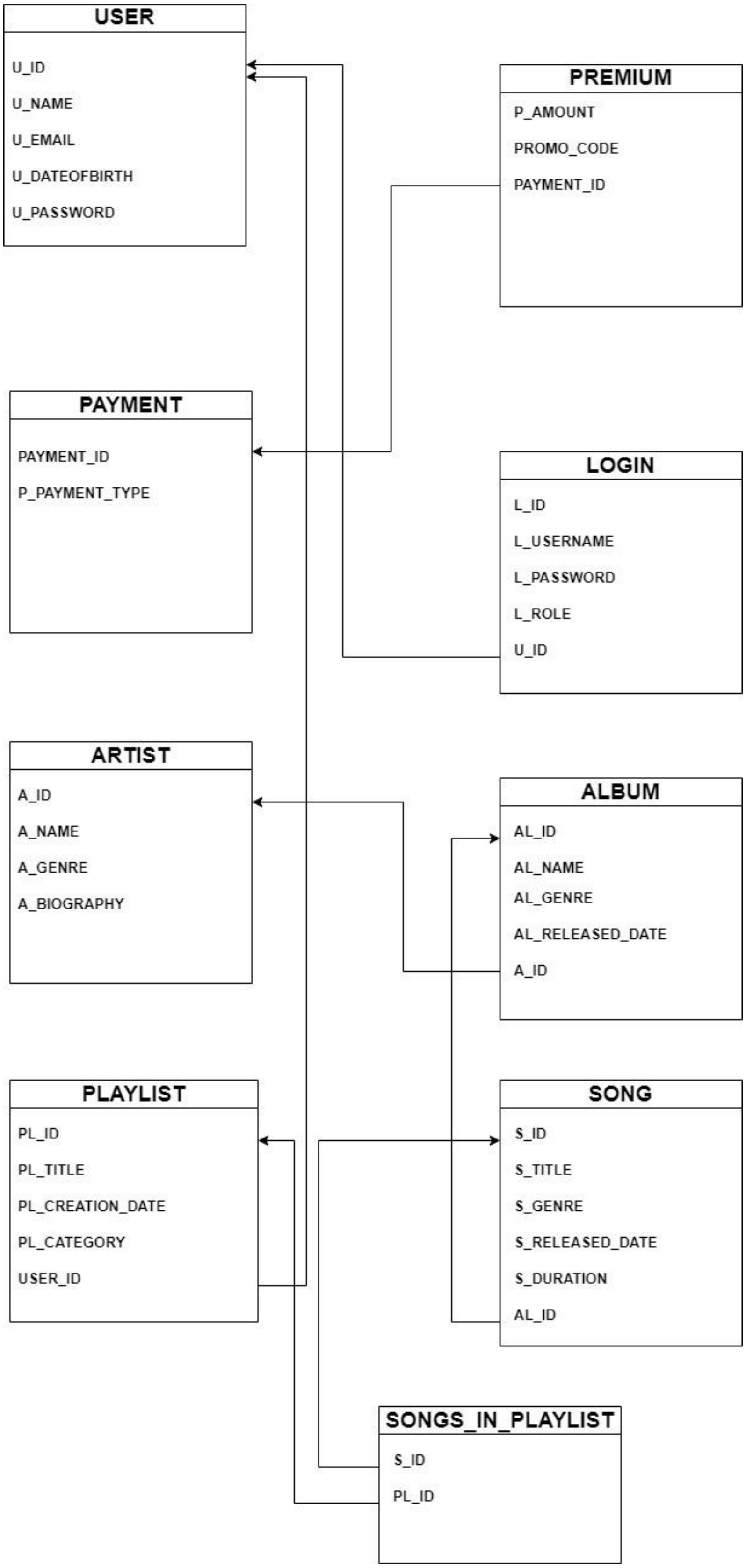
REQUIREMENTS DESIGN SPECIFICATION

- Use case diagram.
- Class diagram
- E-R diagram and Data dictionary
- User Interface design (optional)

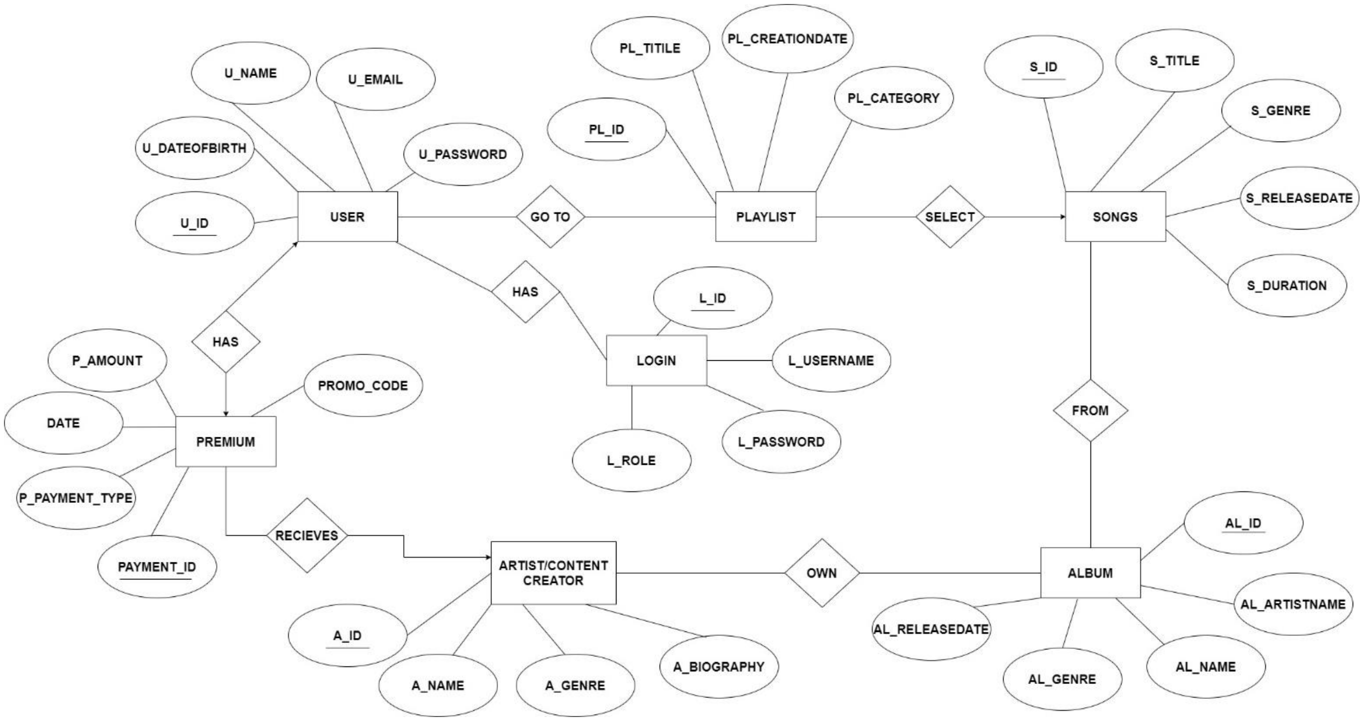
USE CASE DIAGRAM



Class diagram



E-R diagram and Data dictionary


U_NAME
U_EMAIL
PL_TITILE
PL_CREATIONDATE
PL_CATEGORY
S_ID
S_TITLE
S_GENRE
U_DATEOFBIRTH
U_PASSWORD
PL_ID
U_ID
USER
GO TO
PLAYLIST
SELECT
SONGS
S_RELEASEDATE
S_DURATION
HAS
L_ID
HAS
LOGIN
L_USERNAME
P_AMOUNT
PROMO_CODE
DATE
PREMIUM
L_PASSWORD
L_ROLE
FROM
P_PAYMENT_TYPE
RECIEVES
PAYMENT_ID
ARTIST/CONTENT CREATOR
OWN
ALBUM
AL_ID
AL_ARTISTNAME
AL_RELEASEDATE
A_ID
A_NAME
A_GENRE
A_BIOGRAPHY
AL_GENRE
AL_NAME