Music Streaming Service

### Abstract:

This project aims to develop a music streaming service that provides users with personalised music recommendations, listening history, subscription management, and playlist creation. The service includes functionalities for users to make payments, view recommendations, manage playlists, and browse songs by artists and genres. The system is designed using a relational database and object-oriented programming principles in Java.

### Introduction:

The music streaming service project is designed to offer users a comprehensive platform to listen to, manage, and explore music. The service supports user authentication, subscription plans, personalized recommendations, and detailed histories of listened songs. It facilitates the management of music playlists and allows users to make payments for subscriptions. The project involves setting up a database, developing a backend in Java, and implementing various features to enhance the user experience.

### Functional Requirements:

### **-User Management**

1. Registration:
   * Users should be able to register by providing an email, username, and password.
   * The system should generate a unique UserID for each registered user.
2. Authentication:
   * Registered users should be able to log in using their username and password.
   * Users should be able to reset their password if forgotten.
3. Profile Management:
   * Users should be able to update their profile information, including email and password.

### **-Subscription Management**

1. Subscription Plans:
   * Users should be able to view available subscription plans with their names and prices.
   * Users should be able to select and subscribe to a plan.
2. Billing:
   * The system should handle billing based on the selected subscription plan.
   * Users should be able to view their billing history.

### **-Payment Management**

1. Payment Processing:
   * Users should be able to make payments for their subscriptions.
   * The system should record payment details including amount, date, and associated UserID.
2. Payment History:
   * Users should be able to view their payment history.

### **-Music Content Management**

1. Song Management:
   * The system should store information about songs including title, artist, album, genre, length, and release date.
   * Users should be able to view details of songs.
2. Artist Management:
   * The system should store information about artists including name and associated genres.
   * Users should be able to view details of artists and their associated songs and albums.
3. Album Management:
   * The system should store information about albums including title, artist, genre, and release date.
   * Users should be able to view details of albums and their associated songs.
4. Genre Management:
   * The system should store information about genres including name.
   * Users should be able to view details of genres and associated songs and artists.

### **-Playlist Management**

1. Create and Manage Playlists:
   * Users should be able to create, edit, and delete playlists.
   * Users should be able to add or remove songs from playlists.
   * Playlists should have unique IDs and names, associated with specific users.

### **-History and Recommendations**

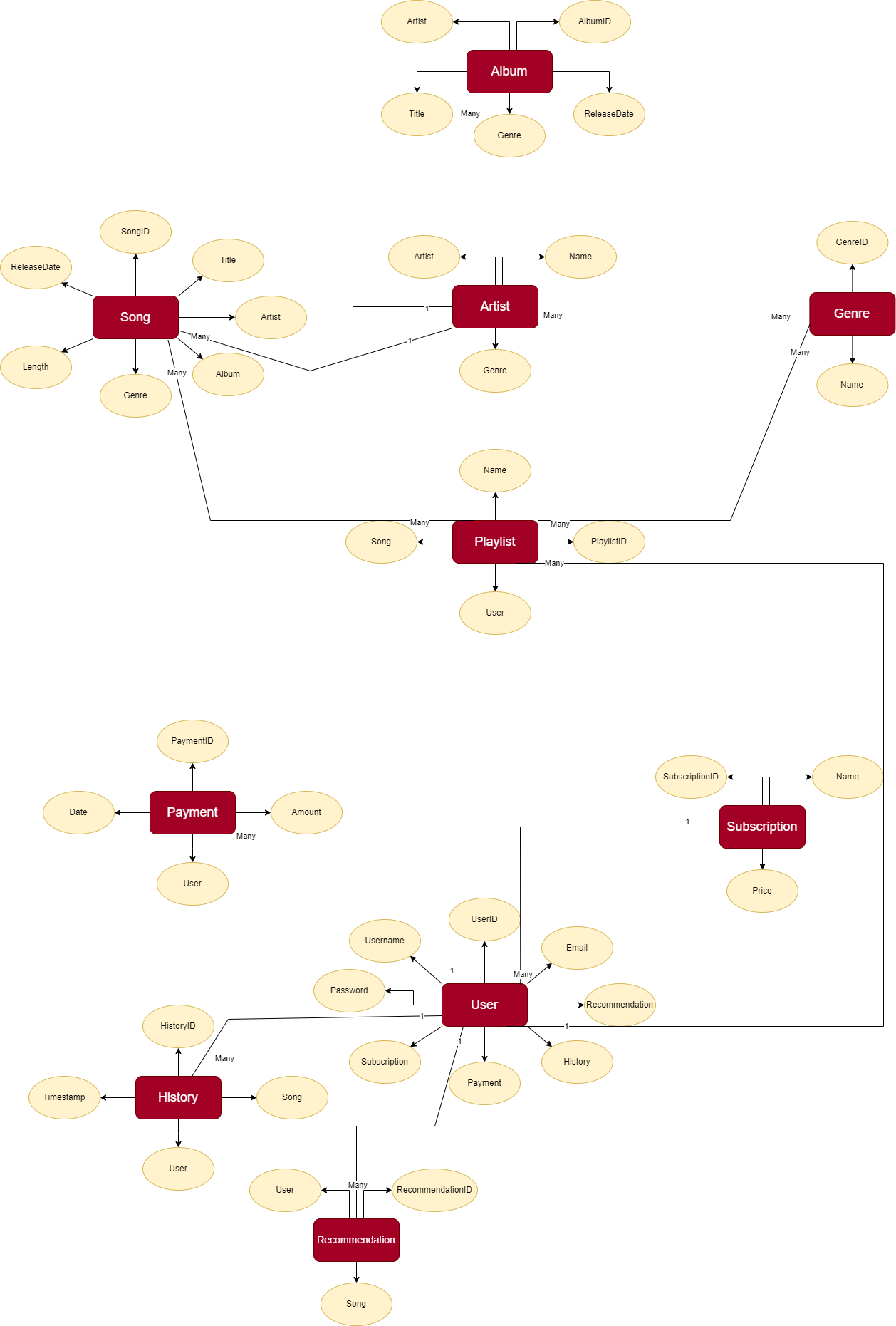
1. Listening History:
   * The system should track and store the listening history of users including songID and timestamp.
   * Users should be able to view their listening history.
2. Recommendations:
   * The system should generate song recommendations for users based on their listening history and preferences.
   * Recommendations should be stored with a unique ID and associated UserID.

**-General Functionalities**

1. Search:
   * Users should be able to search for songs, artists, albums, and genres.
2. Streaming:
   * Users should be able to stream songs.

These functional requirements cover various aspects of the music streaming service, including user management, subscription and payment handling, music content management, playlist management, history tracking, recommendations, and general functionalities like search and streaming. Fulfilling these requirements should ensure music streaming service platform .

ER Diagram of Music Streaming Service:



Queries to create the tables in the Database :

* CREATE TABLE Subscription (

subscriptionID VARCHAR(255) PRIMARY KEY,

name VARCHAR(255) NOT NULL,

price DOUBLE NOT NULL

);

* CREATE TABLE User (

userID VARCHAR(255) PRIMARY KEY,

email VARCHAR(255) NOT NULL,

username VARCHAR(255) NOT NULL,

password VARCHAR(255) NOT NULL,

subscriptionID VARCHAR(255),

FOREIGN KEY (subscriptionID) REFERENCES

Subscription(subscriptionID)

);

* CREATE TABLE Recommendation (

recommendationID VARCHAR(255) PRIMARY KEY,

userID VARCHAR(255),

songID VARCHAR(255),

FOREIGN KEY (userID) REFERENCES User(userID)

);

* CREATE TABLE History (

historyID VARCHAR(255) PRIMARY KEY,

userID VARCHAR(255),

songID VARCHAR(255),

timestamp TIMESTAMP NOT NULL,

FOREIGN KEY (userID) REFERENCES User(userID)

);

* CREATE TABLE Payment (

paymentID VARCHAR(255) PRIMARY KEY,

amount DOUBLE NOT NULL,

date DATE NOT NULL,

userID VARCHAR(255),

FOREIGN KEY (userID) REFERENCES User(userID)

);

* CREATE TABLE Playlist (

playlistID VARCHAR(255) PRIMARY KEY,

userID VARCHAR(255),

name VARCHAR(255),

FOREIGN KEY (userID) REFERENCES User(userID)

);

* CREATE TABLE Genre (

genreID VARCHAR(255) PRIMARY KEY,

name VARCHAR(255) NOT NULL

);

* CREATE TABLE Artist (

artistID VARCHAR(255) PRIMARY KEY,

name VARCHAR(255) NOT NULL,

genreID VARCHAR(255) ,

FOREIGN KEY (genreId) REFERENCES Genre(genreId)

);

* CREATE TABLE Album (

albumID VARCHAR(255) PRIMARY KEY,

artistID VARCHAR(255),

title VARCHAR(255) NOT NULL,

genreID VARCHAR(255),

releaseDate DATE NOT NULL,

FOREIGN KEY (artistID) REFERENCES Artist(artistID),

FOREIGN KEY (genreID) REFERENCES Genre(genreID)

);

* CREATE TABLE Song (

songID VARCHAR(255) PRIMARY KEY,

title VARCHAR(255) NOT NULL,

artistID VARCHAR(255),

albumID VARCHAR(255),

genreID VARCHAR(255),

length DOUBLE NOT NULL,

releaseDate DATE NOT NULL,

FOREIGN KEY (artistID) REFERENCES Artist(artistID),

FOREIGN KEY (albumID) REFERENCES Album(albumID),

FOREIGN KEY (genreID) REFERENCES Genre(genreID)

);

* CREATE TABLE PlaylistSong (

playlistID VARCHAR(255),

songID VARCHAR(255),

PRIMARY KEY (playlistID, songID),

FOREIGN KEY (playlistID) REFERENCES Playlist(playlistID),

FOREIGN KEY (songID) REFERENCES Song(songID)

);

* CREATE TABLE PlaylistGenre (

playlistID VARCHAR(255),

genreID VARCHAR(255),

PRIMARY KEY (playlistID, genreID),

FOREIGN KEY (playlistID) REFERENCES Playlist(playlistID),

FOREIGN KEY (genreID) REFERENCES Genre(genreID));

* CREATE TABLE ArtistGenre (

artistID VARCHAR(255),

genreID VARCHAR(255),

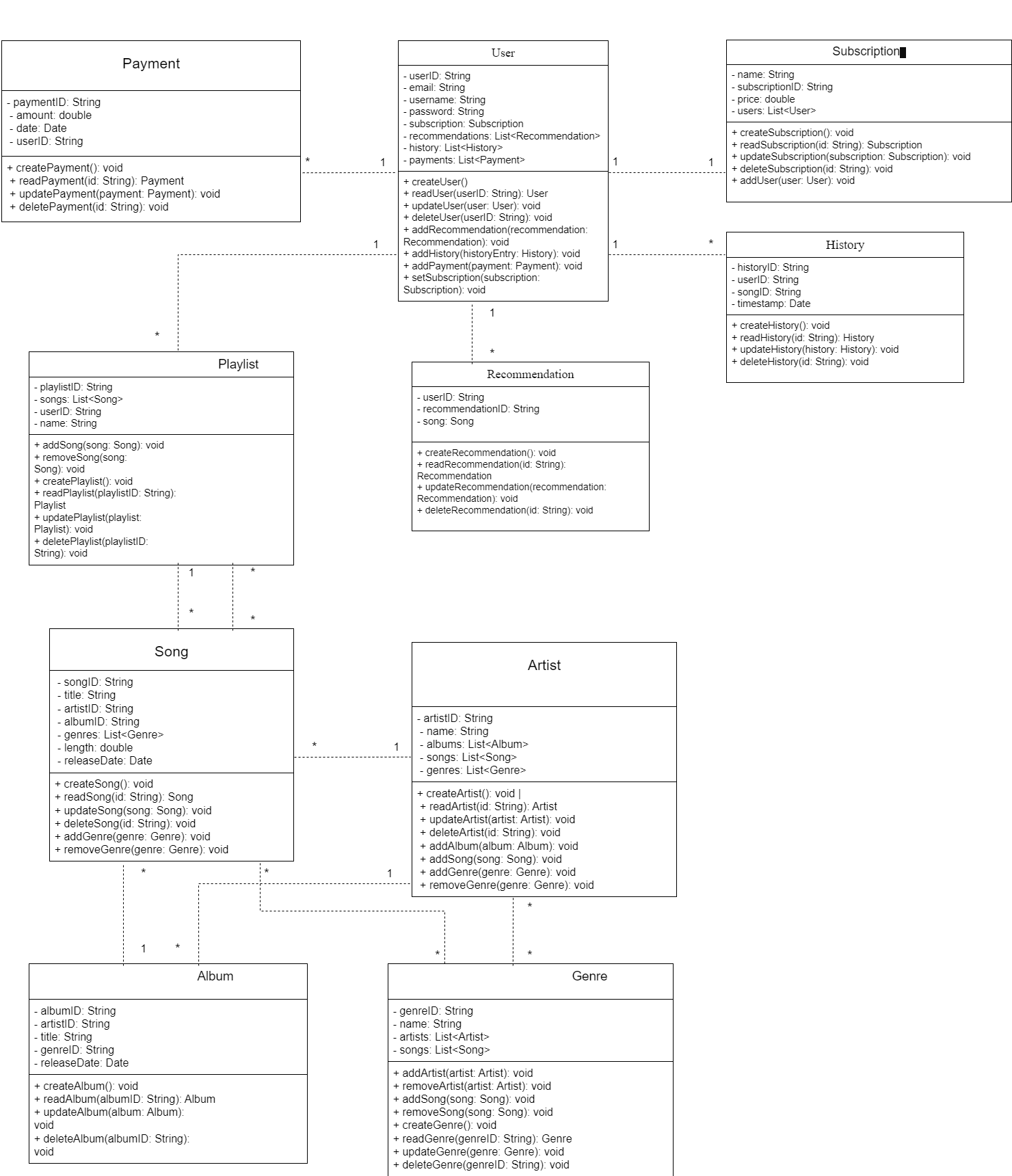
PRIMARY KEY (artistID, genreID),

FOREIGN KEY (artistID) REFERENCES Artist(artistID),

FOREIGN KEY (genreID) REFERENCES Genre(genreID)

);

UML Diagram :



All Classes Java file codes :

**-Album.java**

**package package1;**

**import java.util.Date;**

**public class Album {**

**private int albumId;**

**private int artistId;**

**private String title;**

**private int genreId;**

**private Date releaseDate;**

**public Album(int albumId, int artistId, String title, int genreId, Date releaseDate) {**

**this.albumId = albumId;**

**this.artistId = artistId;**

**this.title = title;**

**this.genreId = genreId;**

**this.releaseDate = releaseDate;**

**}**

**public int getAlbumId() {**

**return albumId;**

**}**

**public void setAlbumId(int albumId) {**

**this.albumId = albumId;**

**}**

**public int getArtistId() {**

**return artistId;**

**}**

**public void setArtistId(int artistId) {**

**this.artistId = artistId;**

**}**

**public String getTitle() {**

**return title;**

**}**

**public void setTitle(String title) {**

**this.title = title;**

**}**

**public int getGenreId() {**

**return genreId;**

**}**

**public void setGenreId(int genreId) {**

**this.genreId = genreId;**

**}**

**public Date getReleaseDate() {**

**return releaseDate;**

**}**

**public void setReleaseDate(Date releaseDate) {**

**this.releaseDate = releaseDate;**

**}**

**}**

**-Artist.java**

**package package1;**

**import java.util.List;**

**import java.util.ArrayList;**

**public class Artist {**

**private int artistId;**

**private String name;**

**private List<Integer> genreIds;**

**public Artist(int artistId, String name) {**

**this.artistId = artistId;**

**this.name = name;**

**this.genreIds = new ArrayList<>();**

**}**

**public int getArtistId() {**

**return artistId;**

**}**

**public void setArtistId(int artistId) {**

**this.artistId = artistId;**

**}**

**public String getName() {**

**return name;**

**}**

**public void setName(String name) {**

**this.name = name;**

**}**

**public List<Integer> getGenreIds() {**

**return genreIds;**

**}**

**public void addGenreId(int genreId) {**

**this.genreIds.add(genreId);**

**}**

**}**

**-Genre.java**

**package package1;**

**public class Genre {**

**private int genreId;**

**private String name;**

**public Genre(int genreId, String name) {**

**this.genreId = genreId;**

**this.name = name;**

**}**

**public int getGenreId() {**

**return genreId;**

**}**

**public void setGenreId(int genreId) {**

**this.genreId = genreId;**

**}**

**public String getName() {**

**return name;**

**}**

**public void setName(String name) {**

**this.name = name;**

**}**

**}**

**-History.java**

**package package1;**

**import java.util.Date;**

**public class History {**

**private int historyId;**

**private int userId;**

**private int songId;**

**private Date timestamp;**

**public History(int historyId, int userId, int songId, Date timestamp) {**

**this.historyId = historyId;**

**this.userId = userId;**

**this.songId = songId;**

**this.timestamp = timestamp;**

**}**

**public int getHistoryId() {**

**return historyId;**

**}**

**public void setHistoryId(int historyId) {**

**this.historyId = historyId;**

**}**

**public int getUserId() {**

**return userId;**

**}**

**public void setUserId(int userId) {**

**this.userId = userId;**

**}**

**public int getSongId() {**

**return songId;**

**}**

**public void setSongId(int songId) {**

**this.songId = songId;**

**}**

**public Date getTimestamp() {**

**return timestamp;**

**}**

**public void setTimestamp(Date timestamp) {**

**this.timestamp = timestamp;**

**}**

**}**

**-Payment.java**

**package package1;**

**import java.util.Date;**

**public class Payment {**

**private int paymentId;**

**private double amount;**

**private Date date;**

**private int userId;**

**public Payment(int paymentId, double amount, Date date, int userId) {**

**this.paymentId = paymentId;**

**this.amount = amount;**

**this.date = date;**

**this.userId = userId;**

**}**

**public int getPaymentId() {**

**return paymentId;**

**}**

**public void setPaymentId(int paymentId) {**

**this.paymentId = paymentId;**

**}**

**public double getAmount() {**

**return amount;**

**}**

**public void setAmount(double amount) {**

**this.amount = amount;**

**}**

**public Date getDate() {**

**return date;**

**}**

**public void setDate(Date date) {**

**this.date = date;**

**}**

**public int getUserId() {**

**return userId;**

**}**

**public void setUserId(int userId) {**

**this.userId = userId;**

**}**

**}**

**-Playlist.java**

**package package1;**

**import java.util.List;**

**import java.util.ArrayList;**

**public class Playlist {**

**private int playlistId;**

**private int userId;**

**private String name;**

**private List<Integer> songIds;**

**public Playlist(int playlistId, int userId, String name) {**

**this.playlistId = playlistId;**

**this.userId = userId;**

**this.name = name;**

**this.songIds = new ArrayList<>();**

**}**

**public int getPlaylistId() {**

**return playlistId;**

**}**

**public void setPlaylistId(int playlistId) {**

**this.playlistId = playlistId;**

**}**

**public int getUserId() {**

**return userId;**

**}**

**public void setUserId(int userId) {**

**this.userId = userId;**

**}**

**public String getName() {**

**return name;**

**}**

**public void setName(String name) {**

**this.name = name;**

**}**

**public List<Integer> getSongIds() {**

**return songIds;**

**}**

**public void addSongId(int songId) {**

**this.songIds.add(songId);**

**}**

**}**

**-Recommendation.java**

**package package1;**

**public class Recommendation {**

**private int userId;**

**private int recommendationId;**

**private String song;**

**public Recommendation(int userId, int recommendationId, String song) {**

**this.userId = userId;**

**this.recommendationId = recommendationId;**

**this.song = song;**

**}**

**public int getUserId() {**

**return userId;**

**}**

**public void setUserId(int userId) {**

**this.userId = userId;**

**}**

**public int getRecommendationId() {**

**return recommendationId;**

**}**

**public void setRecommendationId(int recommendationId) {**

**this.recommendationId = recommendationId;**

**}**

**public String getSong() {**

**return song;**

**}**

**public void setSong(String song) {**

**this.song = song;**

**}**

**}**

**-Song.java**

**package package1;**

**import java.util.Date;**

**import java.util.List;**

**import java.util.ArrayList;**

**public class Song {**

**private int songId;**

**private String title;**

**private int artistId;**

**private int albumId;**

**private List<Integer> genreIds;**

**private int length; // Length in seconds**

**private Date releaseDate;**

**public Song(int songId, String title, int artistId, int albumId, int length, Date releaseDate) {**

**this.songId = songId;**

**this.title = title;**

**this.artistId = artistId;**

**this.albumId = albumId;**

**this.genreIds = new ArrayList<>();**

**this.length = length;**

**this.releaseDate = releaseDate;**

**}**

**public int getSongId() {**

**return songId;**

**}**

**public void setSongId(int songId) {**

**this.songId = songId;**

**}**

**public String getTitle() {**

**return title;**

**}**

**public void setTitle(String title) {**

**this.title = title;**

**}**

**public int getArtistId() {**

**return artistId;**

**}**

**public void setArtistId(int artistId) {**

**this.artistId = artistId;**

**}**

**public int getAlbumId() {**

**return albumId;**

**}**

**public void setAlbumId(int albumId) {**

**this.albumId = albumId;**

**}**

**public List<Integer> getGenreIds() {**

**return genreIds;**

**}**

**public void addGenreId(int genreId) {**

**this.genreIds.add(genreId);**

**}**

**public int getLength() {**

**return length;**

**}**

**public void setLength(int length) {**

**this.length = length;**

**}**

**public Date getReleaseDate() {**

**return releaseDate;**

**}**

**public void setReleaseDate(Date releaseDate) {**

**this.releaseDate = releaseDate;**

**}**

**}**

**-Subsciption.java**

**package package1;**

**public class Subscription {**

**private int subscriptionId;**

**private String name;**

**private double price;**

**public Subscription(int subscriptionId, String name, double price) {**

**this.subscriptionId = subscriptionId;**

**this.name = name;**

**this.price = price;**

**}**

**public int getSubscriptionId() {**

**return subscriptionId;**

**}**

**public void setSubscriptionId(int subscriptionId) {**

**this.subscriptionId = subscriptionId;**

**}**

**public String getName() {**

**return name;**

**}**

**public void setName(String name) {**

**this.name = name;**

**}**

**public double getPrice() {**

**return price;**

**}**

**public void setPrice(double price) {**

**this.price = price;**

**}**

**}**

**-User.java**

**package package1;**

**import java.util.Date;**

**public class Payment {**

**private int paymentId;**

**private double amount;**

**private Date date;**

**private int userId;**

**public Payment(int paymentId, double amount, Date date, int userId) {**

**this.paymentId = paymentId;**

**this.amount = amount;**

**this.date = date;**

**this.userId = userId;**

**}**

**public int getPaymentId() {**

**return paymentId;**

**}**

**public void setPaymentId(int paymentId) {**

**this.paymentId = paymentId;**

**}**

**public double getAmount() {**

**return amount;**

**}**

**public void setAmount(double amount) {**

**this.amount = amount;**

**}**

**public Date getDate() {**

**return date;**

**}**

**public void setDate(Date date) {**

**this.date = date;**

**}**

**public int getUserId() {**

**return userId;**

**}**

**public void setUserId(int userId) {**

**this.userId = userId;**

**}**

**}**

**-Main.java**

**package package1;**

**import java.util.Date;**

**public class Main {**

**public static void main(String[] args) {**

**Subscription basicSubscription = new Subscription(1, "Basic", 9.99);**

**Subscription premiumSubscription = new Subscription(2, "Premium", 19.99);**

**User user1 = new User(1, "user1@example.com", "user1", "password1", basicSubscription.getSubscriptionId());**

**User user2 = new User(2, "user2@example.com", "user2", "password2", premiumSubscription.getSubscriptionId());**

**Artist artist1 = new Artist(1, "Artist One");**

**Artist artist2 = new Artist(2, "Artist Two");**

**artist1.addGenreId(1);**

**artist2.addGenreId(2);**

**Genre genre1 = new Genre(1, "Rock");**

**Genre genre2 = new Genre(2, "Pop");**

**Album album1 = new Album(1, artist1.getArtistId(), "Album One", genre1.getGenreId(), new Date());**

**Album album2 = new Album(2, artist2.getArtistId(), "Album Two", genre2.getGenreId(), new Date());**

**Song song1 = new Song(1, "Song One", artist1.getArtistId(), album1.getAlbumId(), 240, new Date());**

**song1.addGenreId(genre1.getGenreId());**

**Song song2 = new Song(2, "Song Two", artist2.getArtistId(), album2.getAlbumId(), 180, new Date());**

**song2.addGenreId(genre2.getGenreId());**

**Playlist playlist1 = new Playlist(1, user1.getUserId(), "Playlist One");**

**playlist1.addSongId(song1.getSongId());**

**playlist1.addSongId(song2.getSongId());**

**History history1 = new History(1, user1.getUserId(), song1.getSongId(), new Date());**

**user1.addHistory(history1);**

**History history2 = new History(2, user2.getUserId(), song2.getSongId(), new Date());**

**user2.addHistory(history2);**

**Recommendation rec1 = new Recommendation(1, 1, "Recommended Song One");**

**user1.addRecommendation(rec1);**

**Recommendation rec2 = new Recommendation(2, 2, "Recommended Song Two");**

**user2.addRecommendation(rec2);**

**Payment payment1 = new Payment(1, 9.99, new Date(), user1.getUserId());**

**user1.addPayment(payment1);**

**Payment payment2 = new Payment(2, 19.99, new Date(), user2.getUserId());**

**user2.addPayment(payment2);**

**System.out.println("User 1 Subscription: " + basicSubscription.getName());**

**System.out.println("User 1 Recommendations: " + user1.getRecommendations().size());**

**System.out.println("User 1 History: " + user1.getHistory().size());**

**System.out.println("User 1 Payments: " + user1.getPayments().size());**

**System.out.println("Artist 1 Name: " + artist1.getName());**

**System.out.println("Artist 1 Genres: " + artist1.getGenreIds().size());**

**System.out.println("Playlist 1 Songs: " + playlist1.getSongIds().size());**

**System.out.println("Song 1 Title: " + song1.getTitle());**

**System.out.println("Song 1 Genres: " + song1.getGenreIds().size());**

**}**

**}**

Challenges list :

* Designing a normalised and efficient database schema to handle complex relationships like many-to-many (e.g., Song-Genre, Song-Playlist, Artist-Genre).
* Creating the tables in a specific order and inserting the values accordingly .
* Implementing and managing the various one-to-many and many-to-many relationships in Java objects and ensuring data integrity.
* Ensuring that the data entered into the system is valid and consistent.