DBMS PORJECT REPORT

PROJECT NAME : MUSIC STREAMING MANAGEMENT

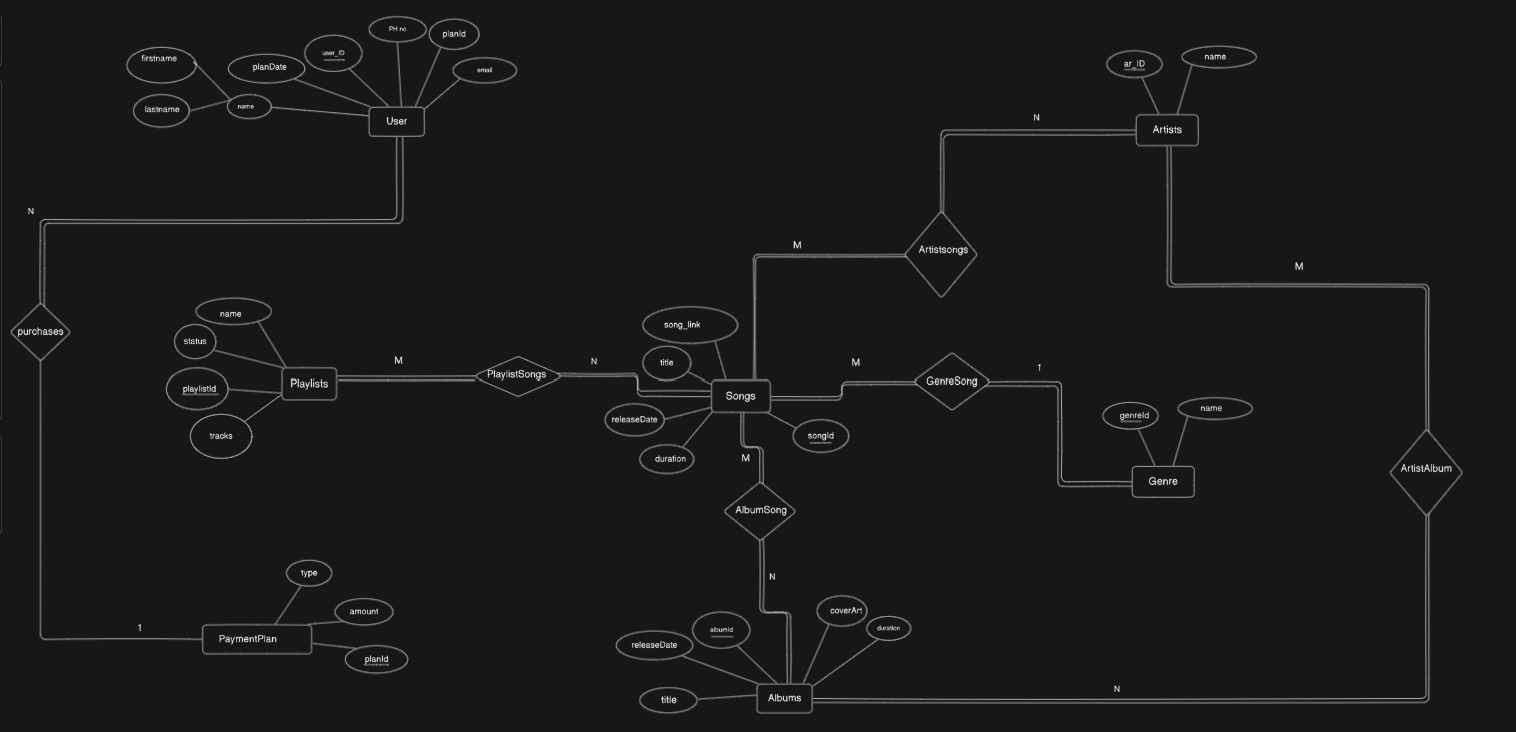
TEAM MEMBER DETAILS:

|  |  |
| --- | --- |
| NAME | SRN |
| **VARAD** | **PES1UG22CS674** |
| **VISHRUTH S MEGUR** | **PES1UG22CS703** |

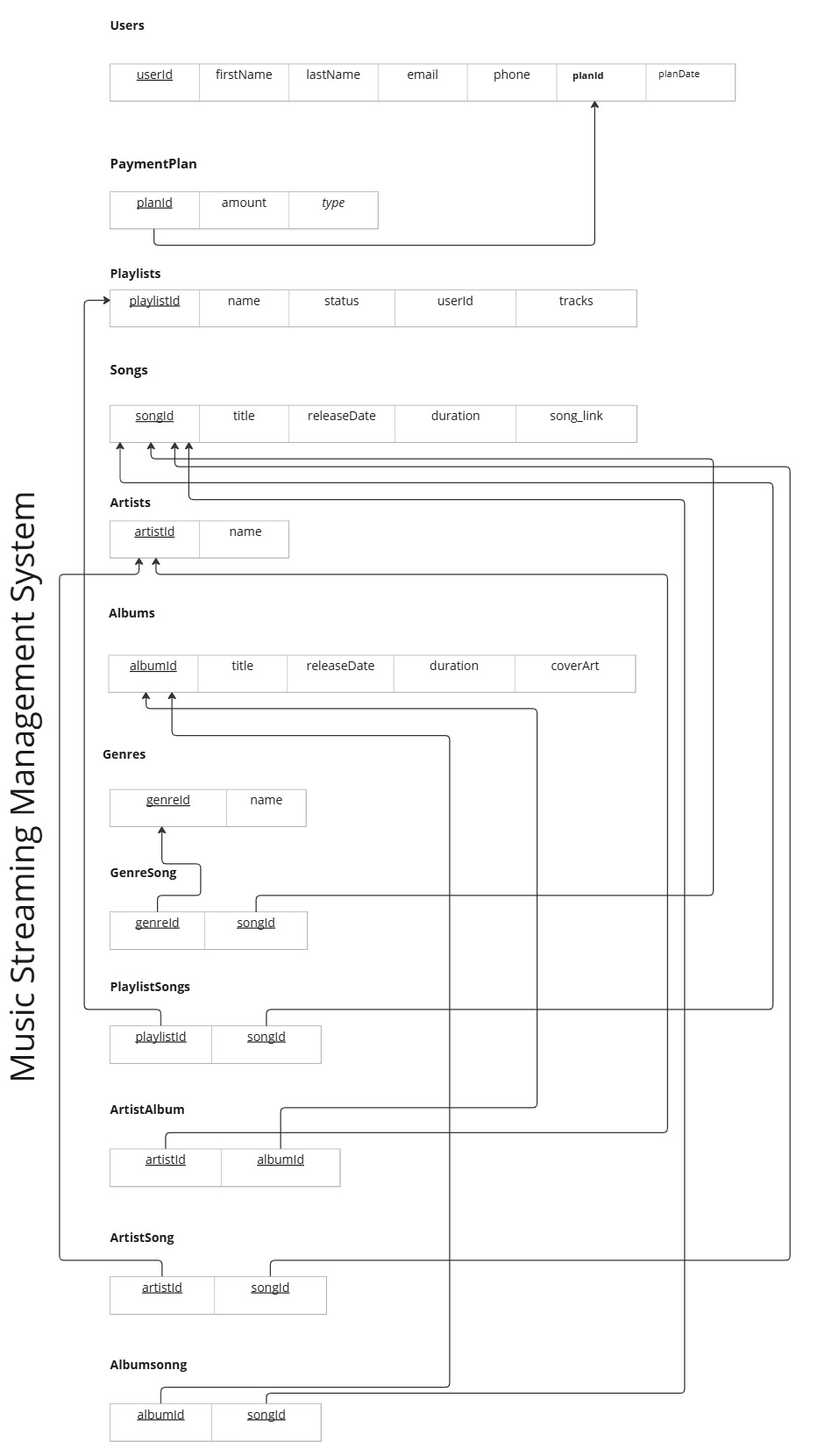
OVERVIEW:

The project, Music Streaming Management, is designed to optimize the organization and management of a music streaming platform. It allows users to interact with playlists, songs, genres, and subscription plans through a database, ensuring secure access and smooth operation.

ER DIAGRAM:



RELATION SCHEMA:



DDL QUERIES:

TABLE CREATION QUERY:

CREATE TABLE IF NOT EXISTS `StreamingDatabase`.`playlist` (

  `playlistId` INT NOT NULL AUTO\_INCREMENT,

  `name` VARCHAR(45) NOT NULL,

  `status` ENUM('Public', 'Private') NOT NULL,

  PRIMARY KEY (`playlistId`))

ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `StreamingDatabase`.`user` (

  `userId` INT NOT NULL AUTO\_INCREMENT,

  `firstName` VARCHAR(64) NOT NULL,

  `lastName` VARCHAR(64) NOT NULL,

  `email` VARCHAR(128) NOT NULL,

  `phone` VARCHAR(10) NOT NULL,

  PRIMARY KEY (`userId`),

  UNIQUE INDEX `email\_UNIQUE` (`email` ASC) VISIBLE)

CREATE TABLE IF NOT EXISTS `StreamingDatabase`.`payment\_plan` (

  `date` DATE NOT NULL,

  `paymentId` INT NOT NULL AUTO\_INCREMENT,

  `amount` INT NOT NULL,

  `type` ENUM("Monthly", "Yearly", "Free") NOT NULL,

  PRIMARY KEY (`paymentId`))

CREATE TABLE IF NOT EXISTS `StreamingDatabase`.`genre` (

  `genreId` INT NOT NULL AUTO\_INCREMENT,

  `name` VARCHAR(128) NOT NULL,

  PRIMARY KEY (`genreId`),

  UNIQUE INDEX `name\_UNIQUE` (`name` ASC) VISIBLE)

CREATE TABLE IF NOT EXISTS `StreamingDatabase`.`songs` (

  `songId` INT NOT NULL AUTO\_INCREMENT,

  `title` VARCHAR(45) NOT NULL,

  `release` DATE NOT NULL,

  `duration` INT NOT NULL,

  PRIMARY KEY (`songId`))

CREATE TABLE IF NOT EXISTS `StreamingDatabase`.`album` (

  `albumId` INT NOT NULL AUTO\_INCREMENT,

  `title` VARCHAR(45) NOT NULL,

  `release` DATE NOT NULL,

  `albumcol` VARCHAR(45) NULL,

  PRIMARY KEY (`albumId`))

CREATE TABLE IF NOT EXISTS `StreamingDatabase`.`artist` (

  `artistId` INT NOT NULL AUTO\_INCREMENT,

  `firstName` VARCHAR(128) NOT NULL,

  `lastName` VARCHAR(128) NOT NULL,

  PRIMARY KEY (`artistId`))

CREATE TABLE IF NOT EXISTS `StreamingDatabase`.`playlistSong` (

  `playlistId` INT NOT NULL,

  `songId` INT NOT NULL,

  PRIMARY KEY (`playlistId`, `songId`),

  INDEX `songId\_idx` (`songId` ASC) VISIBLE,

  CONSTRAINT `playlistId`

    FOREIGN KEY (`playlistId`)

    REFERENCES `StreamingDatabase`.`playlist` (`playlistId`)

    ON DELETE CASCADE

    ON UPDATE CASCADE,

  CONSTRAINT `songId`

    FOREIGN KEY (`songId`)

    REFERENCES `StreamingDatabase`.`songs` (`songId`)

    ON DELETE NO ACTION

    ON UPDATE NO ACTION)

DML QUERIES:

1.INSERT QUERIES:

LINSERT INTO `songs` VALUES (1,'Feel The Love','2018-06-08','00:02:45')….

INSERT INTO `paymentplans` VALUES (1,60,'Yearly')….

INSERT INTO `genresong` VALUES (2,1)….

INSERT INTO `genres` VALUES (2,'hip-hop')….

INSERT INTO `artistsong` VALUES (1,1)

INSERT INTO `artists` VALUES (1,'Kanye West')….

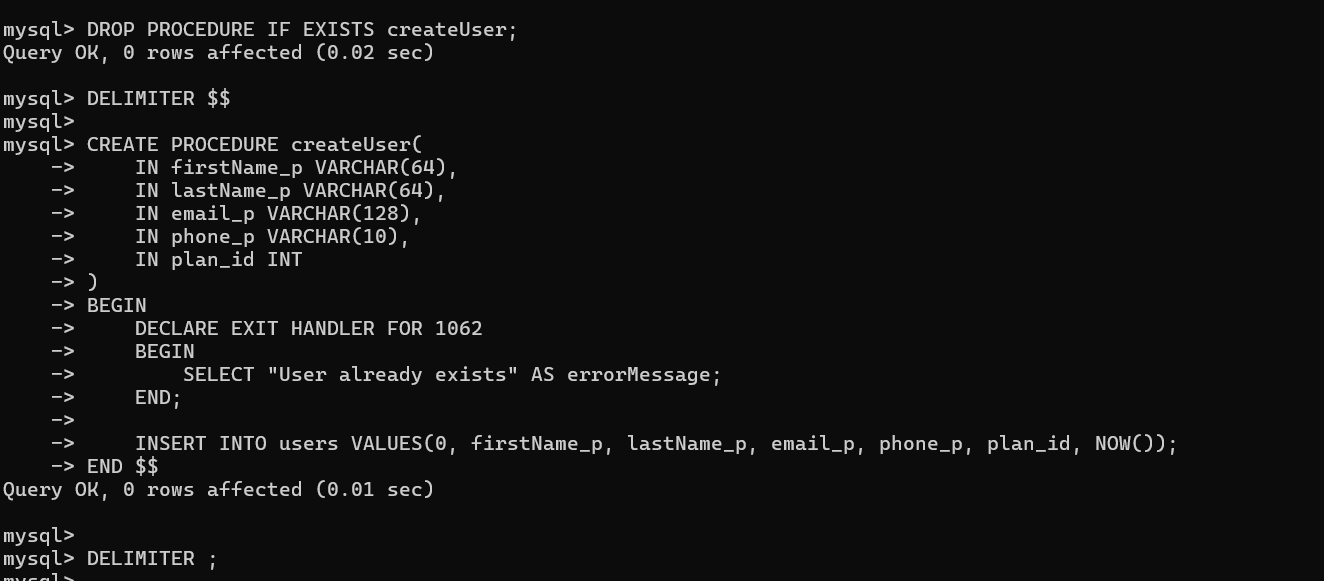
INSERT INTO `artistalbum` VALUES (1,1)….

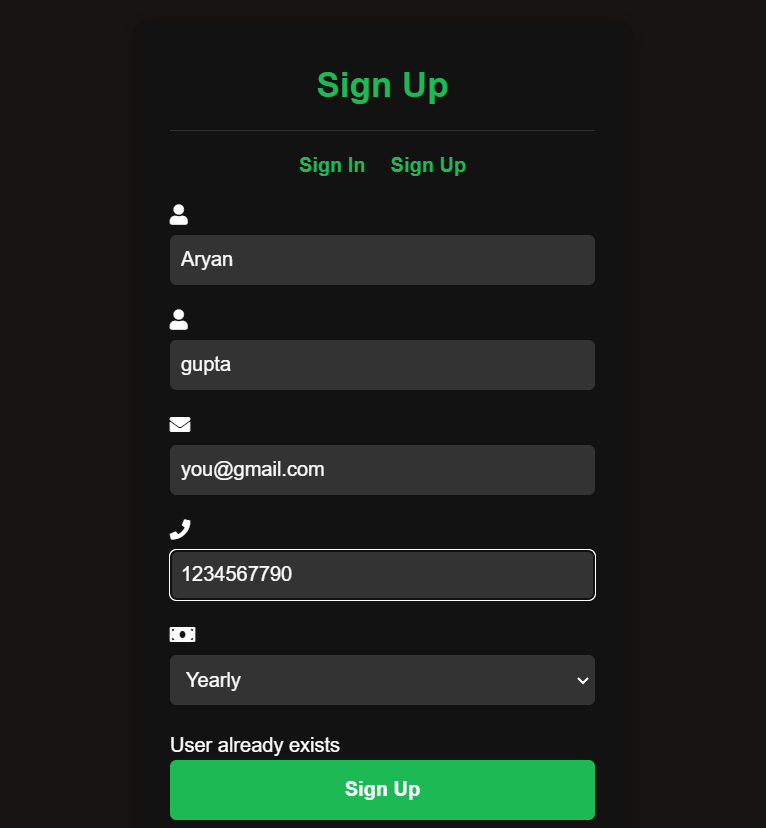
INSERT INTO `albumsong` VALUES (1,1)….

INSERT INTO `albums` VALUES (1,'Kids See Ghosts','2018-06-08','00:23:52','/covers/ksg.png')….

2.PROCEDURE:

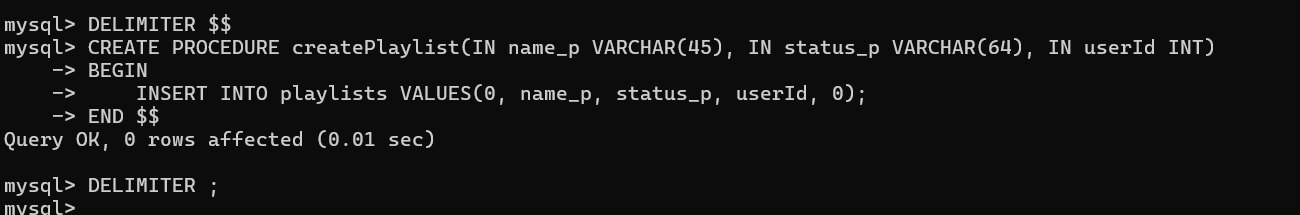
* Procedure to create user while logging into website

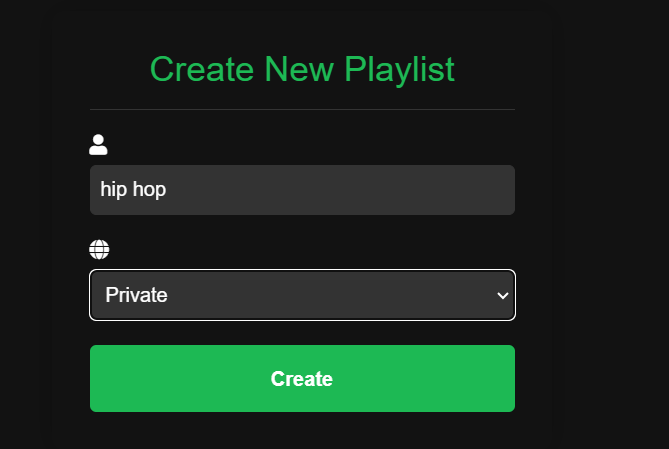




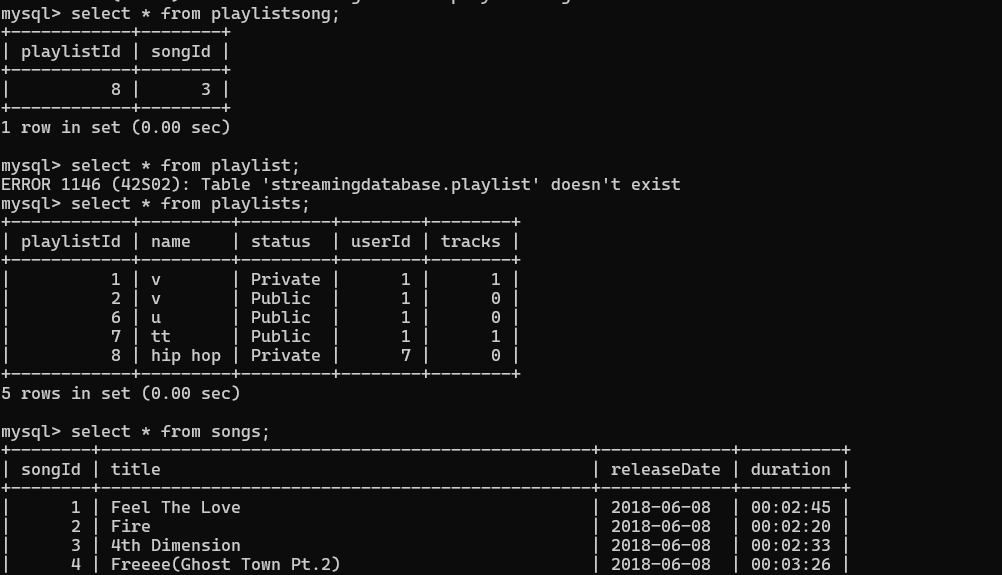
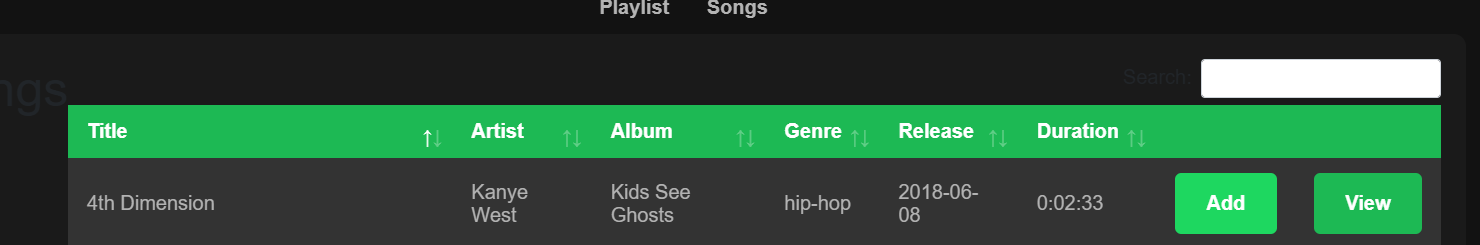
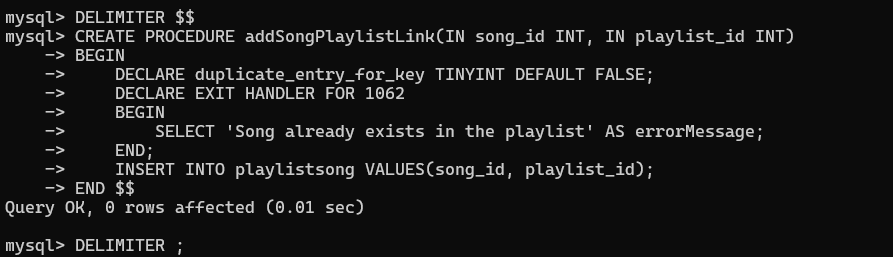


* Procedure to create a playlist





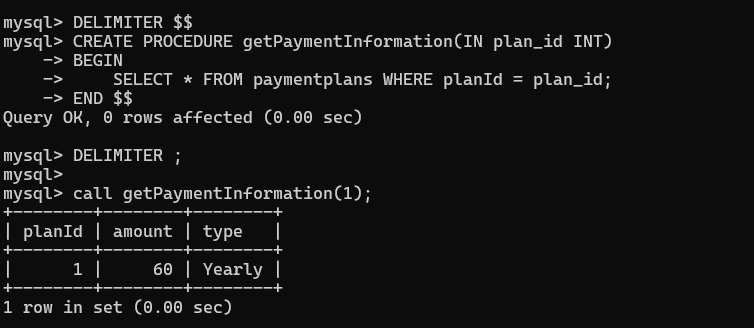
* Procedure to add song into the playlist



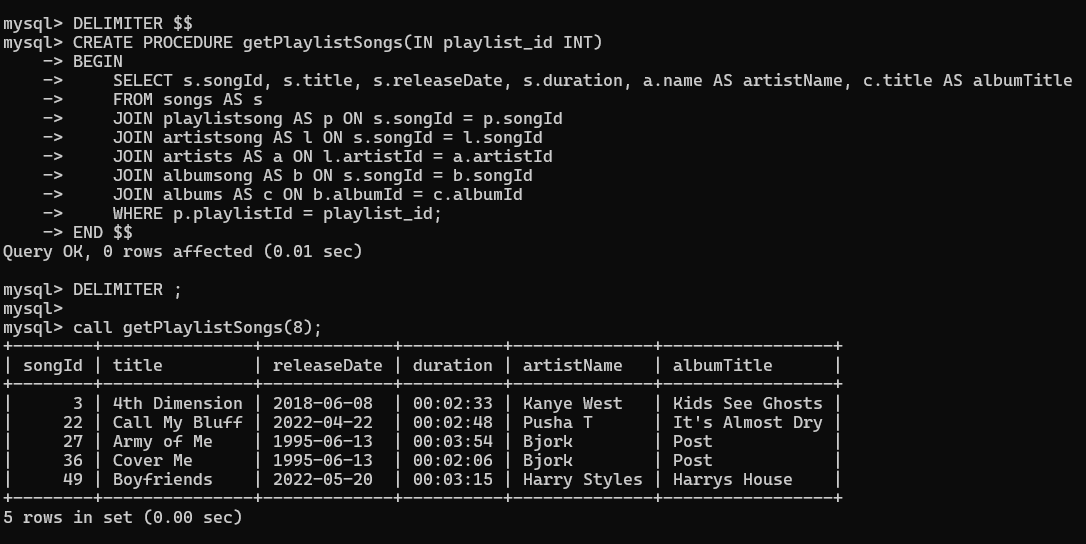
* Procedure to get information about a user from user ID



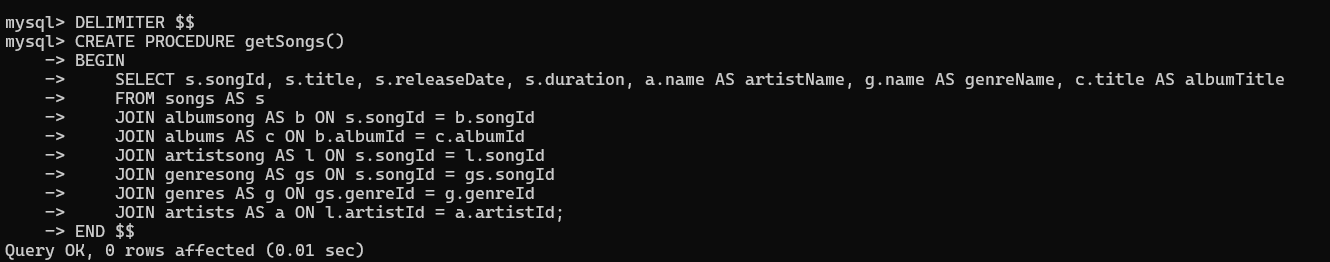
* Procedure to get information about a payment-plan from payment-plan ID



* Procedure to display a playlist

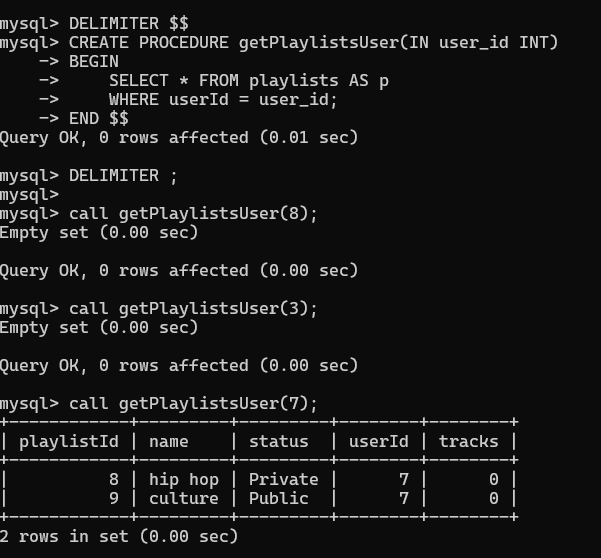


* Procedure to call all the songs

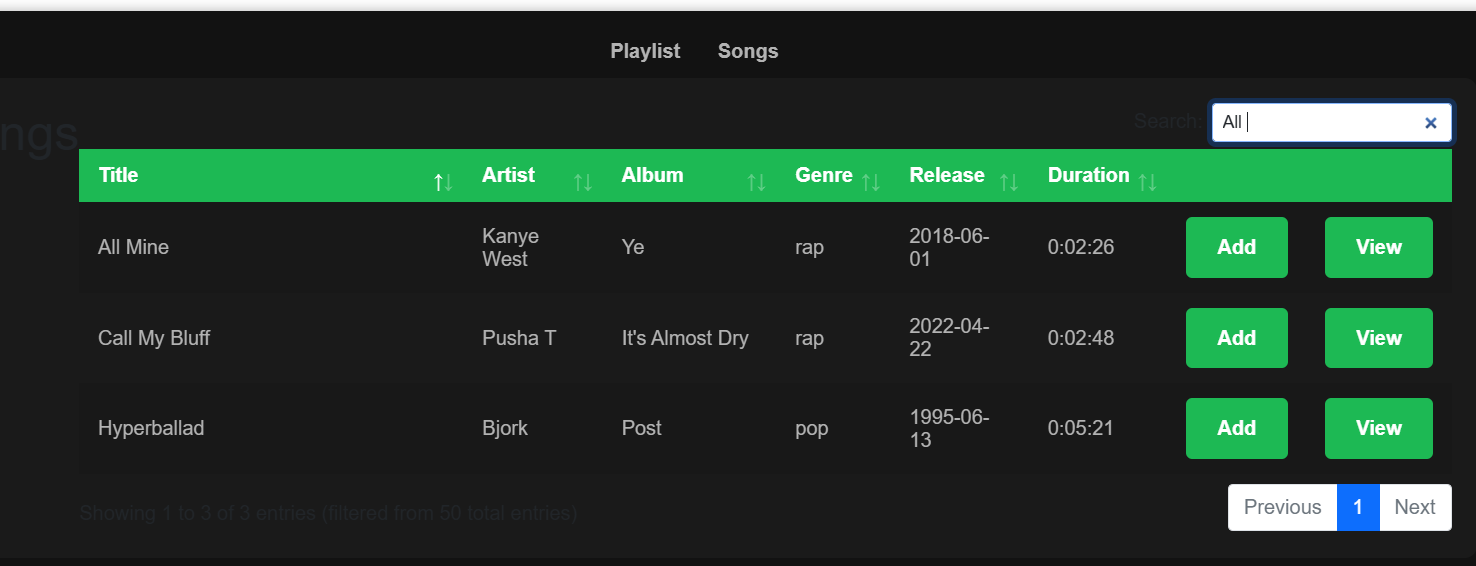
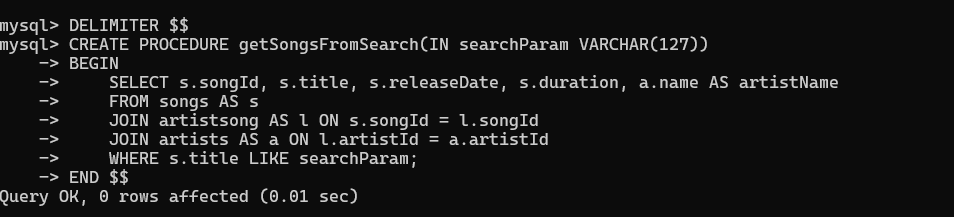




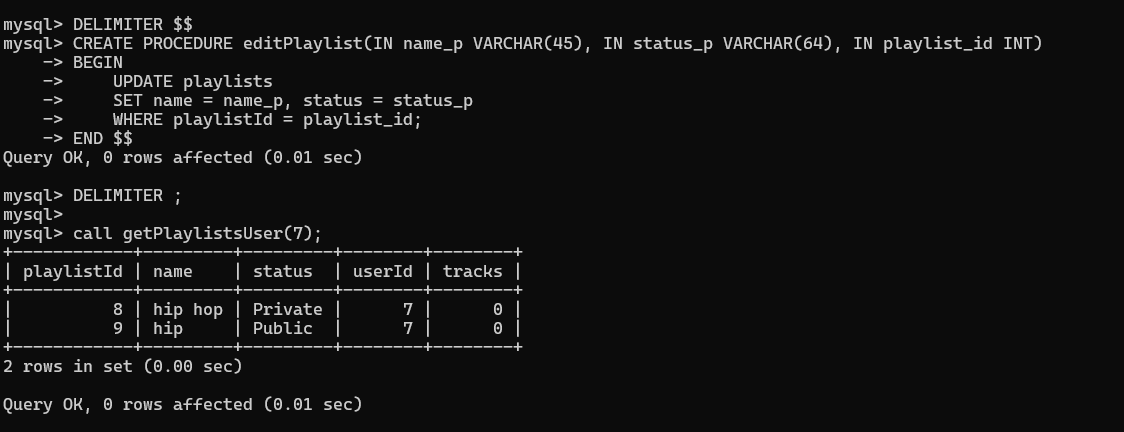
* Procedure to return the playlist created by the user

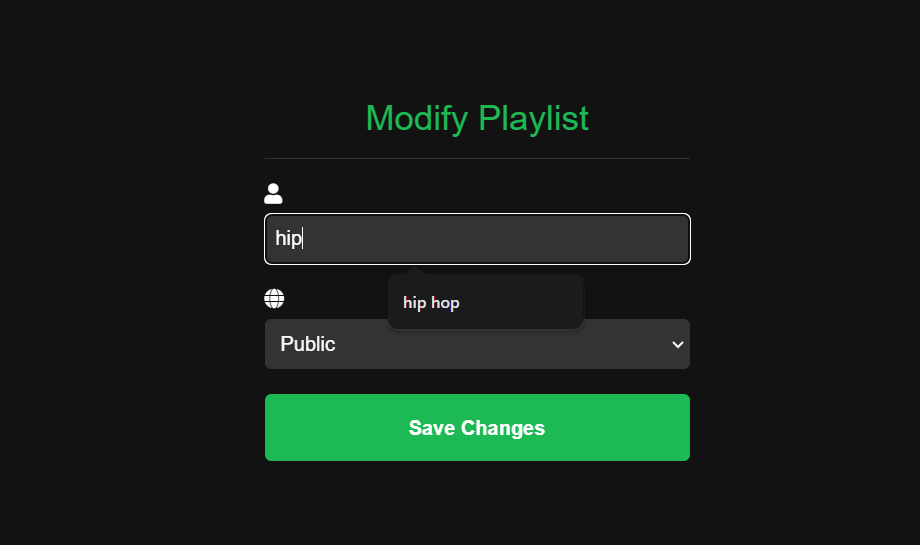


* Procedure to search a song in search bar

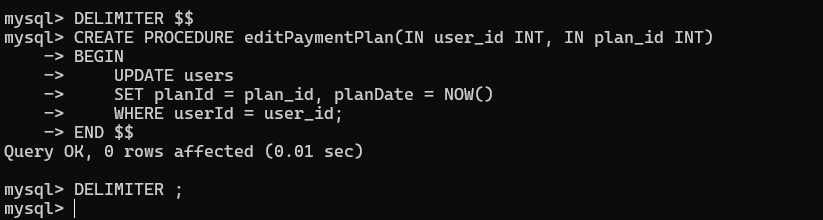


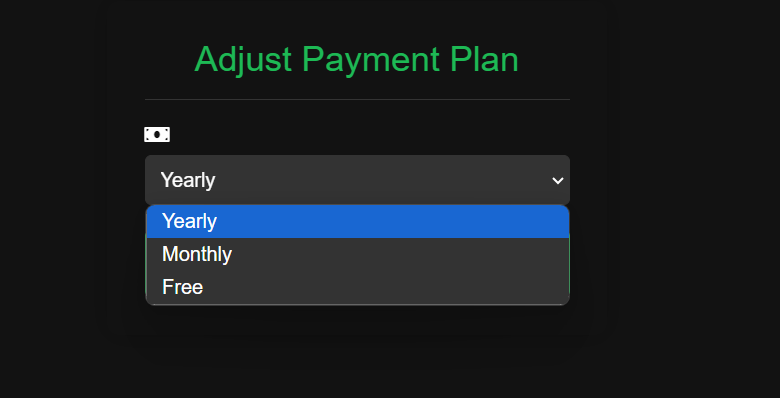
* Procedure to change a playlist name



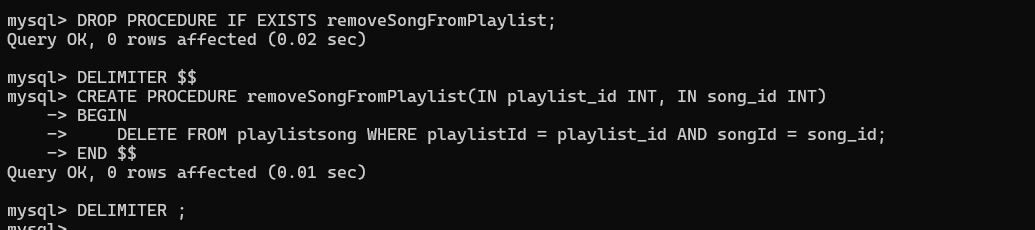


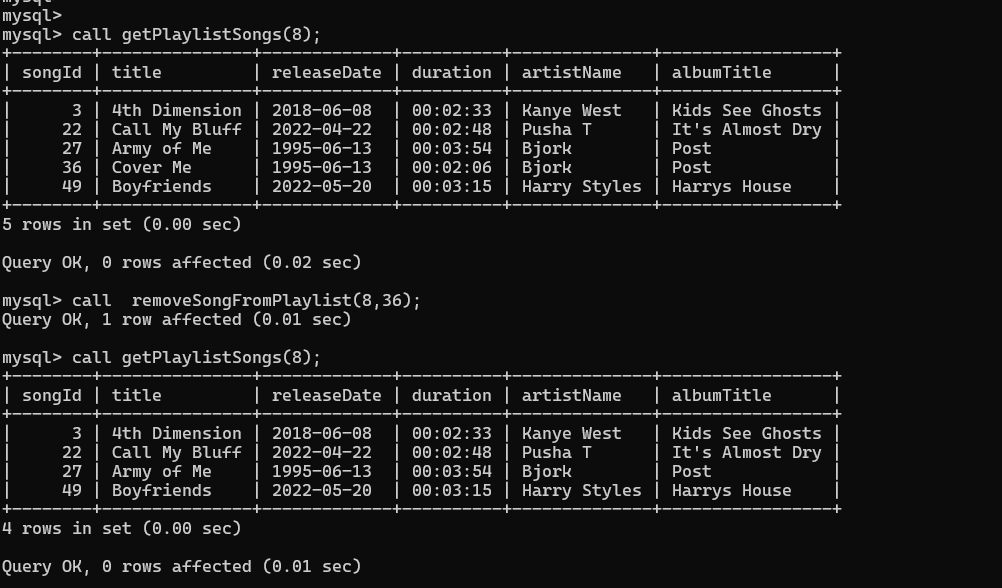
* Procedure to change the payment plan



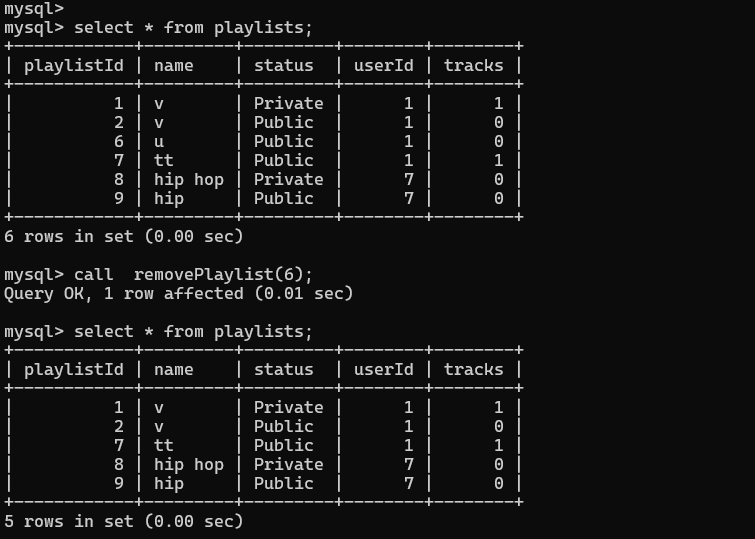
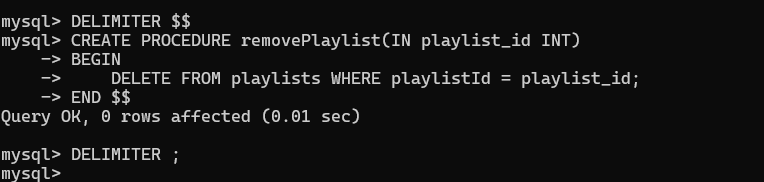


* Procedure to remove song from playlist

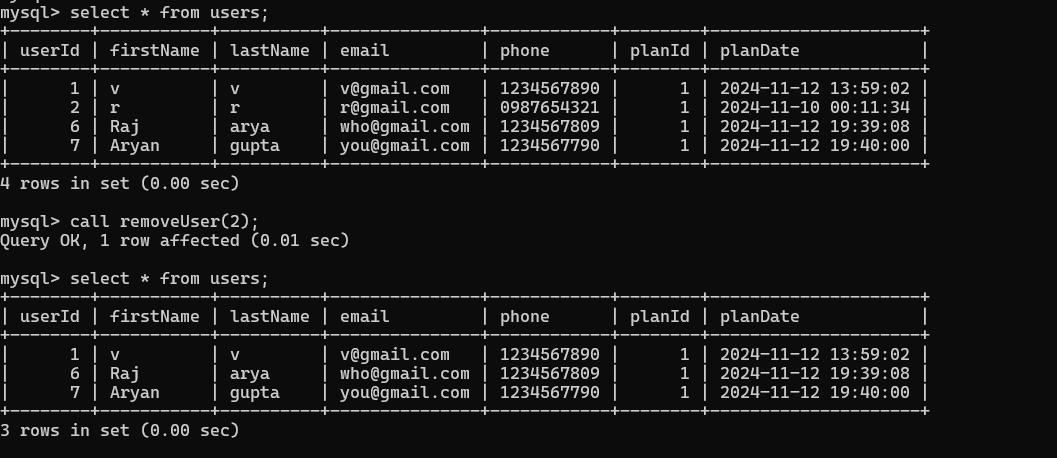
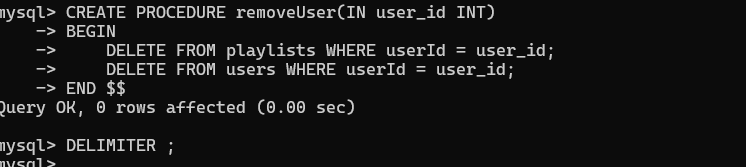




* Procedure to remove playlist

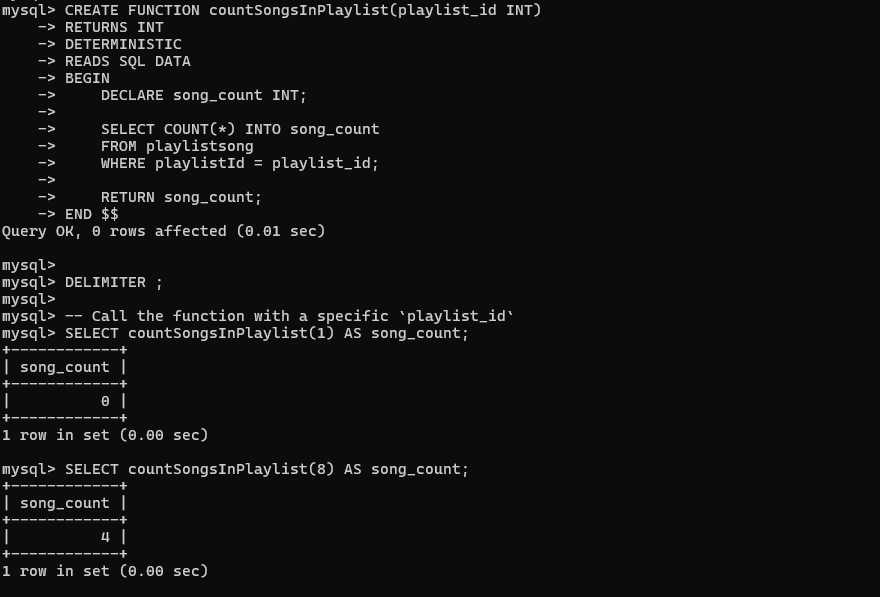


* Procedure to delete a users account

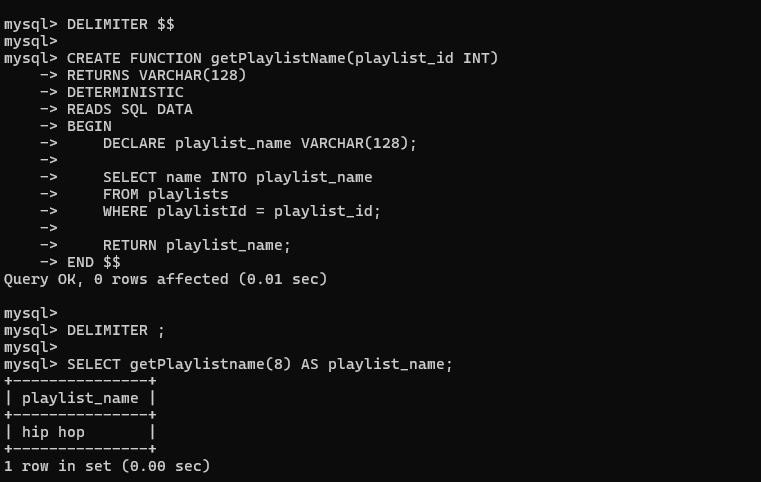


3.FUNCTIONS

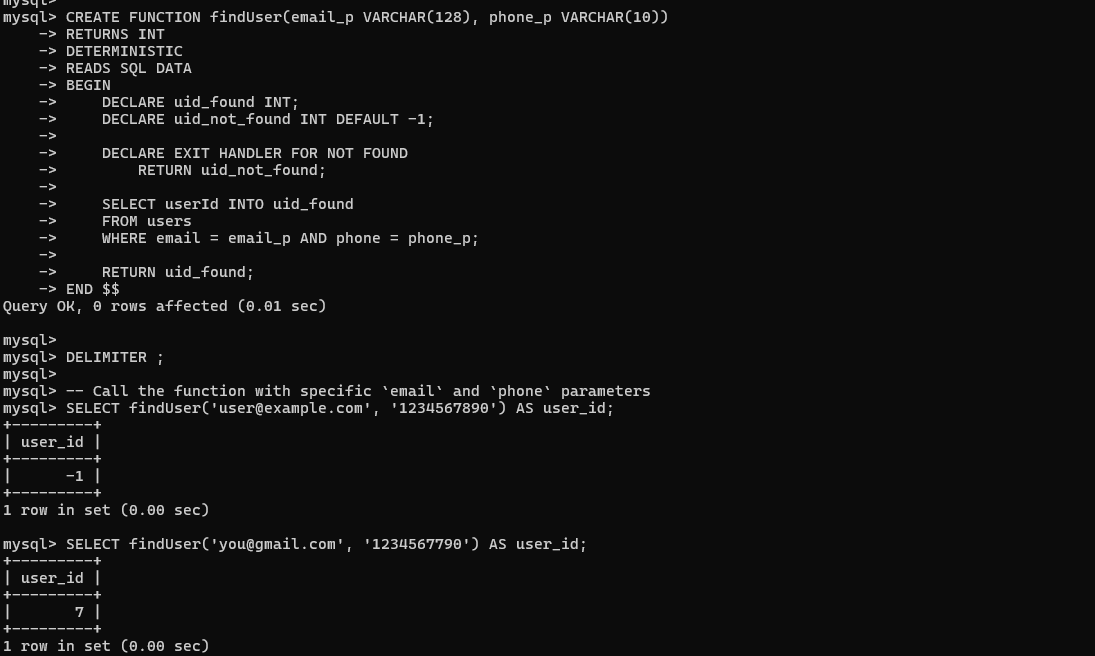
* Function to count the number of songs in a playlist



* Function to get a playlist name from playlist ID



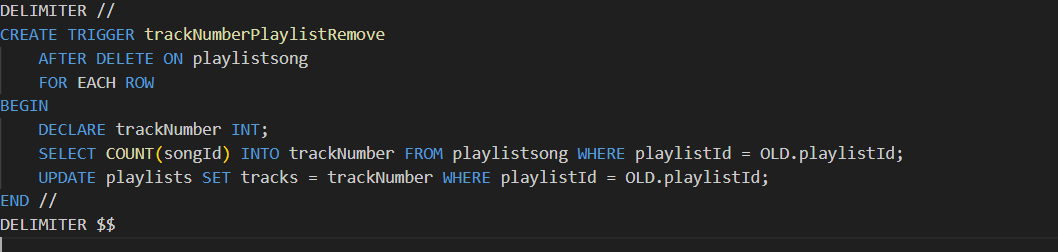
* Function to find is a user exists or not in database



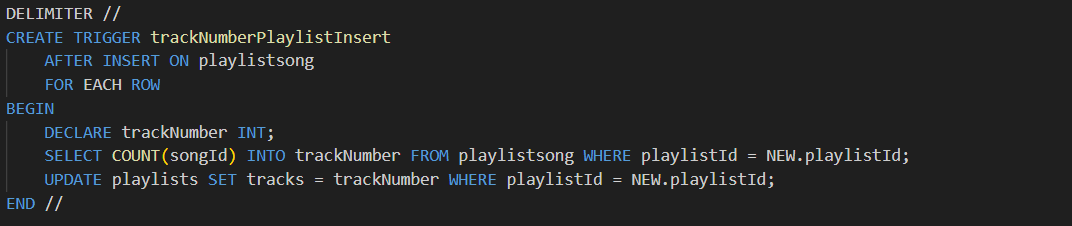
.

4.TRIGGERS

* This trigger is activated when ever we remove a song from playlist , this trigger decreases the count of tracks in playlist by one.

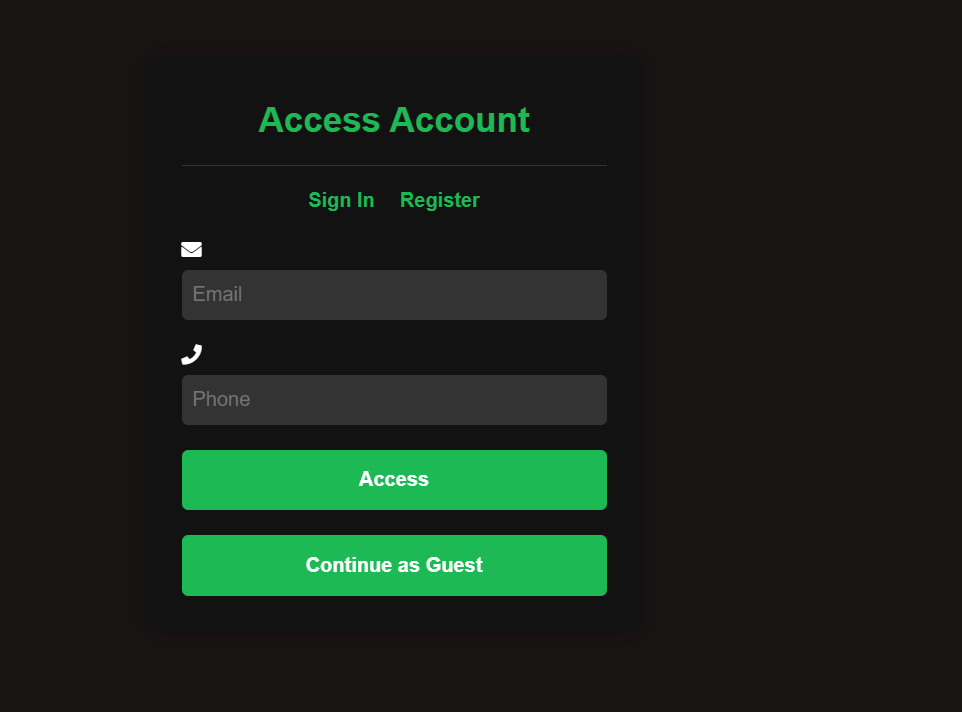


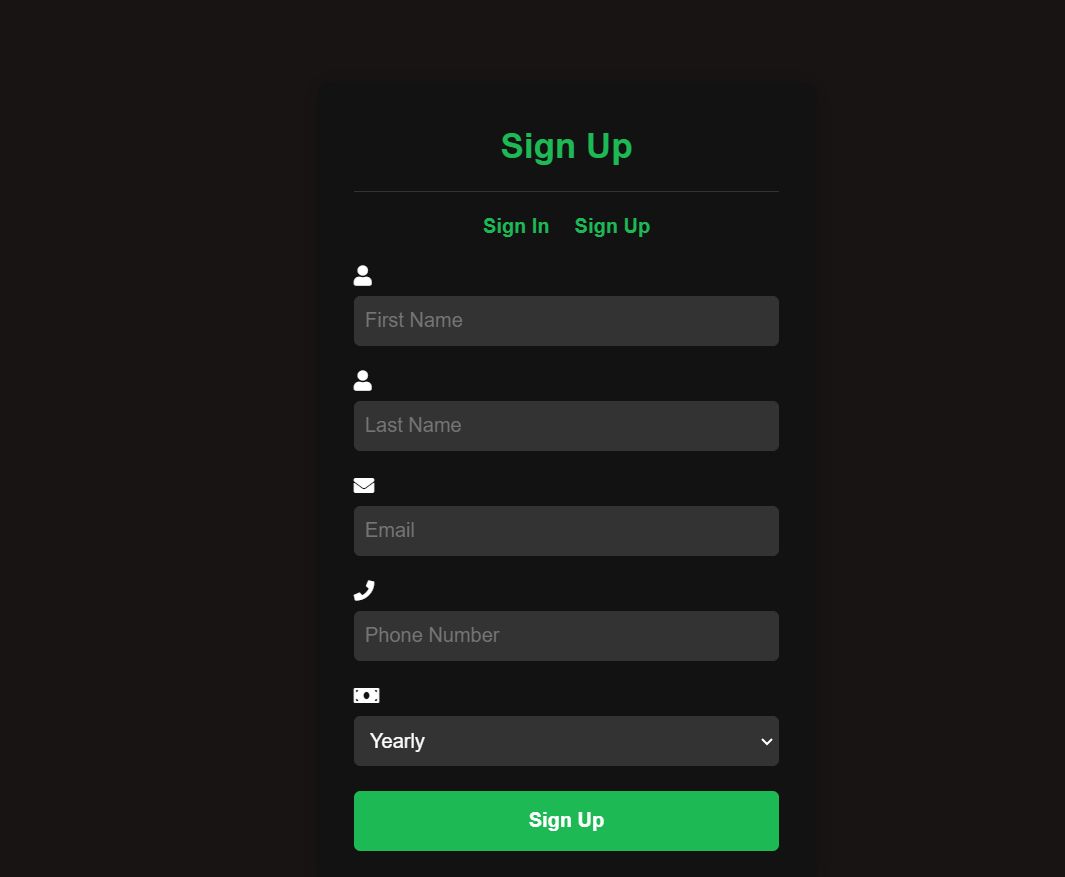
* This trigger is activated when ever we add a song to the playlist , this trigger increases the count of tracks in playlist by one.



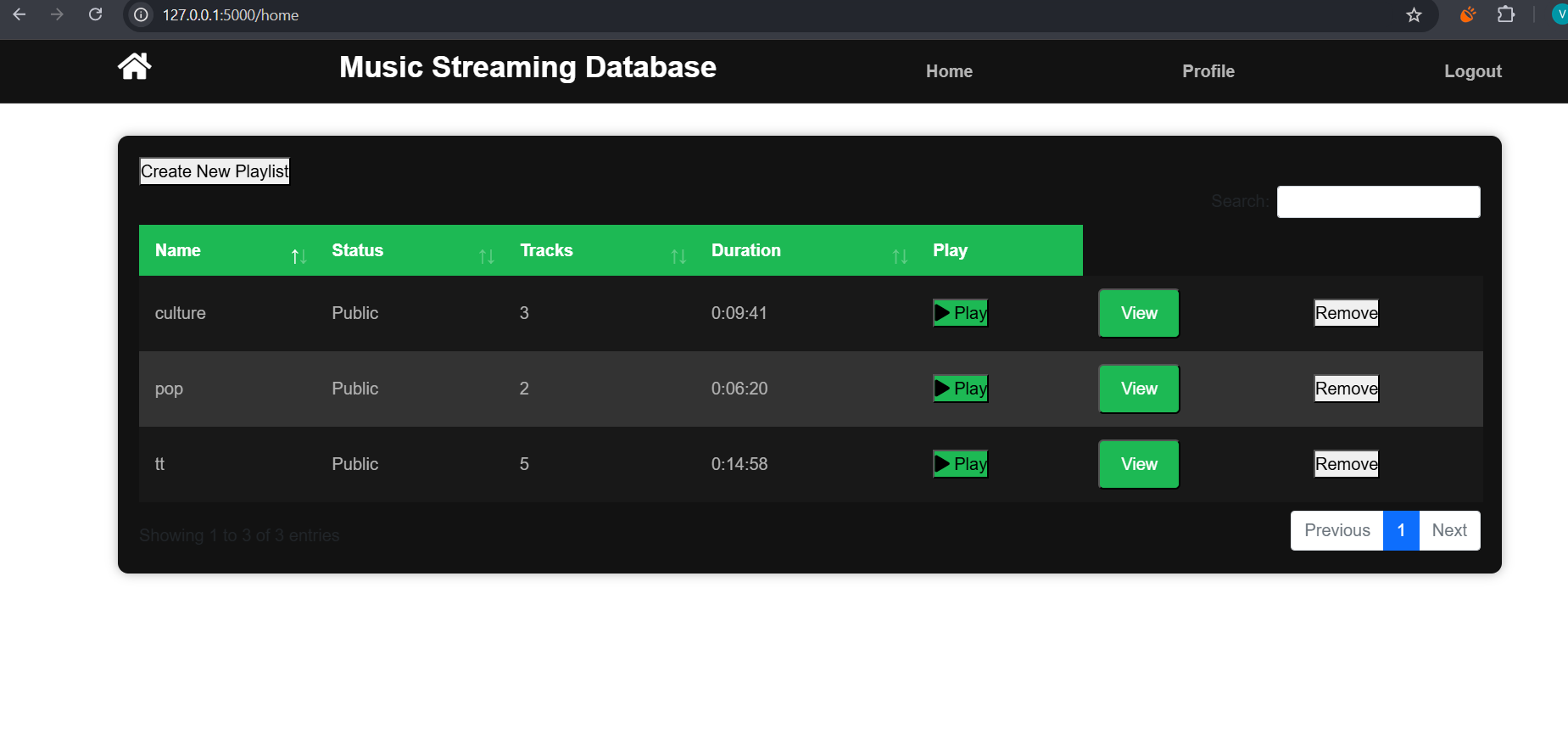
**FRONTEND:**

**Login:**

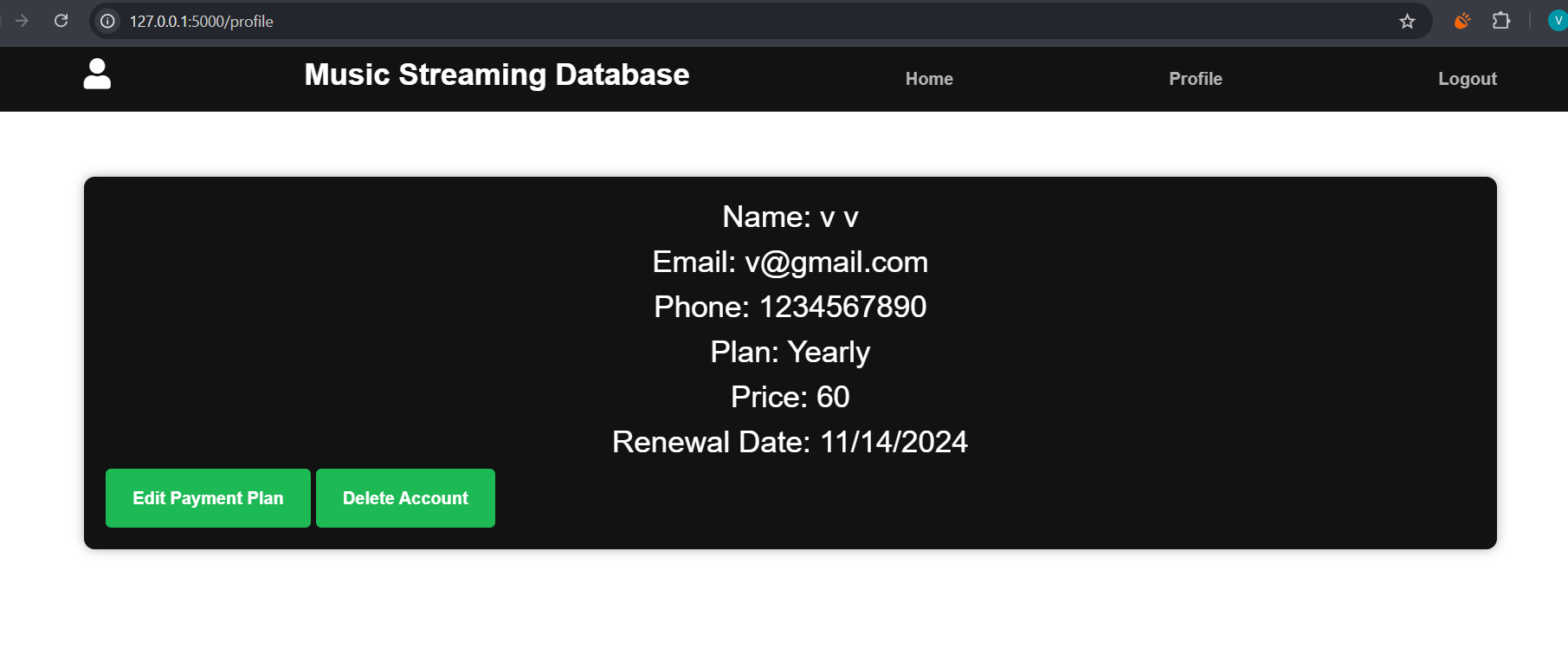
****

**REGISTER:**

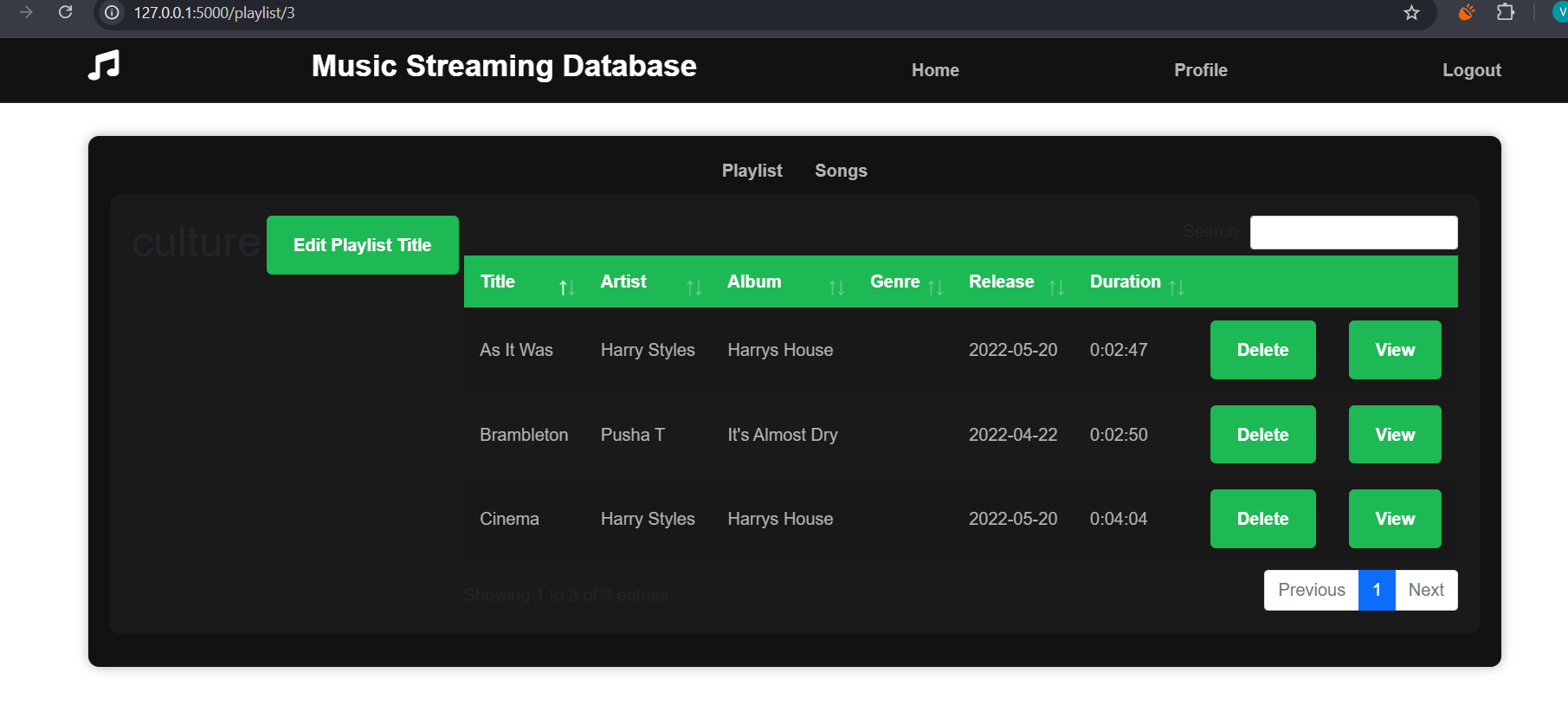
**HOME:**

****

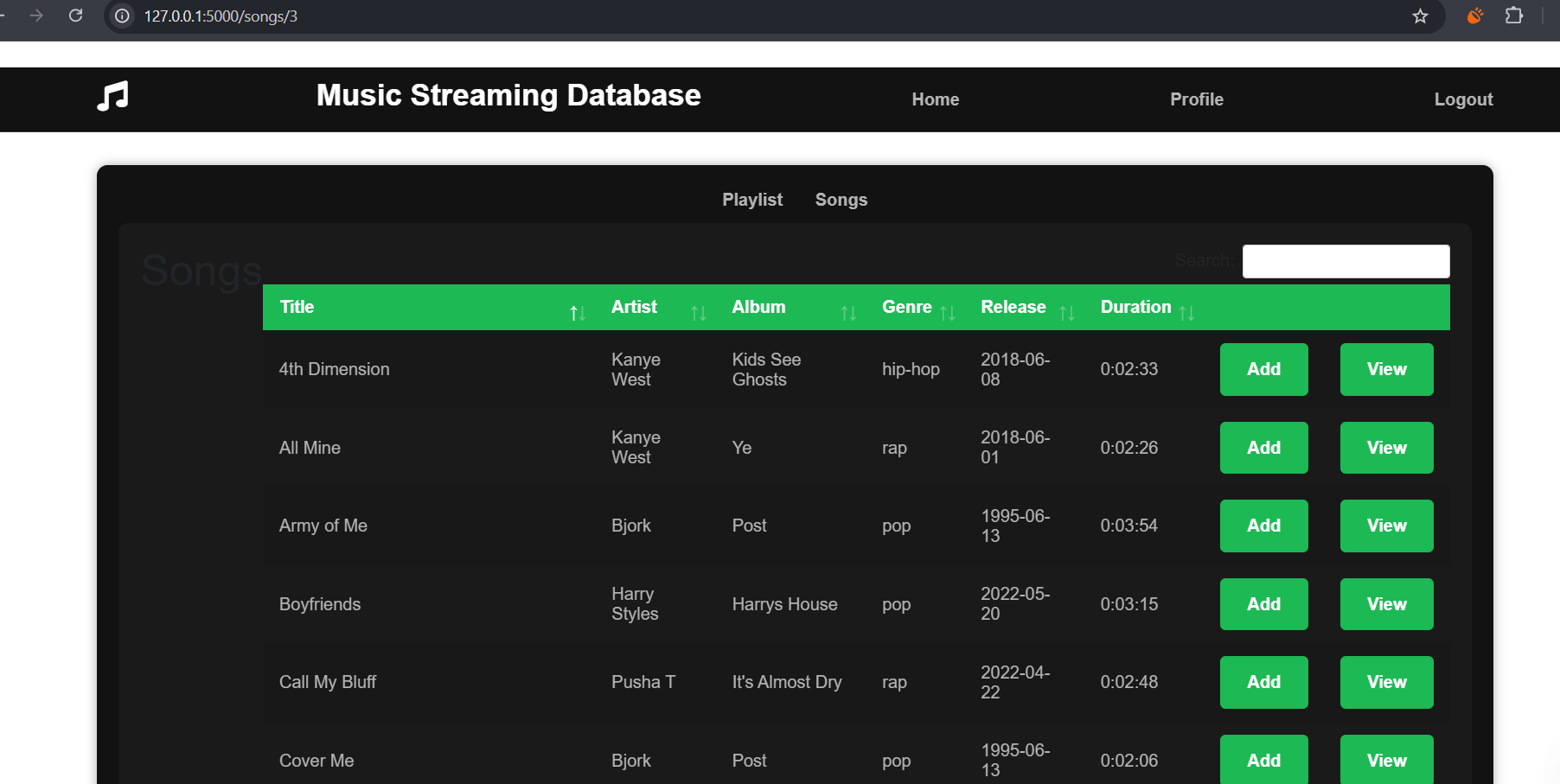
**PROFILE:**

****

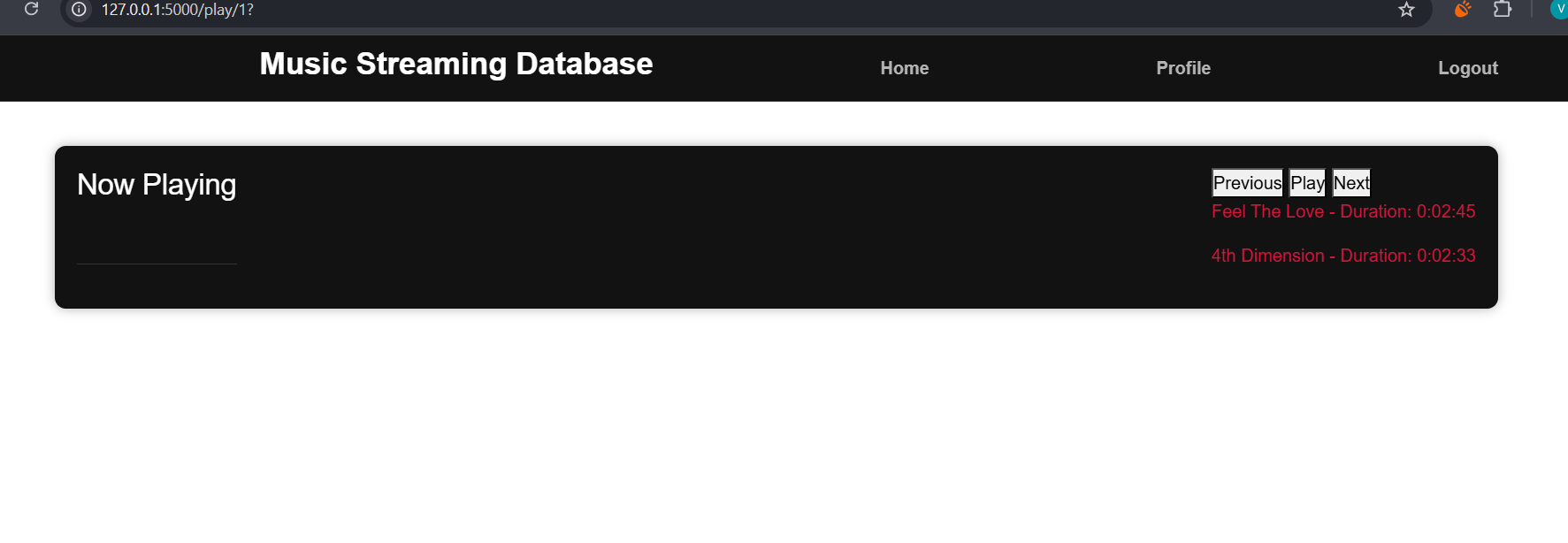
**PLAYLIST:**



**SONGS:**

****

**MUSIC PALYING PAGE:**

****