Spring Final Project

Youtube Url: https://youtu.be/QlAJt-Mz010

GitHub Url: https://github.com/Luckyboy-1/SpringFinalProject.git

Project Participants:

Moses Charo

Title:

Online Music Store

Executive Summary:

This project is focused on creating an online music store with a database design that consists of three entities: Artist, Album, and Track. These entities are represented by their respective tables, each containing specific attributes to store relevant information. The Artist table includes attributes such as artist_id (primary key), name, bio, and genre. The Album table consists of album_id (primary key), title, release_date, and artist_id (foreign key) to establish a one-to-many relationship with the Artist table. Lastly, the Track table contains track_id (primary key), title, duration, and album_id (foreign key) to support the many-to-many relationship between albums and tracks. To implement the RESTful Web API server, we have chosen Advanced Rest Client (ARC) as the testing tool. We have defined the API endpoints for each feature, allowing users to interact with the system.

Initial Features:

Create a bulleted list of Initial Features. This will be a list of features that meet the requirements of the final project.

Database Design:

1. Entities: Artist, Album, Track
2. Tables:

    a. Artist:

    - artist_id (Primary Key)
    - artist_name
    - bio
    - genre

    b. Album:

    - album_id (Primary Key)
    - album_title

    - release_date
    - artist_id (Foreign Key)

  c. Track:

    - track_id (Primary Key)
    - track_title
    - duration
    - album_id (Foreign Key)

There is a one-to-many relationship that exists between the Artist and Album tables, where an artist can have multiple albums. The foreign key "artist_id" in the Album table establishes this relationship. The many-to-many relationship exists between the Album and Track tables, where an album can have multiple tracks, and a track can belong to multiple albums.

CRUD Operations:

1. Artist:
    - Create: Create a new artist record with name, bio, and genre.
    - Read: Retrieve artist details based on artist_id or search for artists by name or genre.
    - Update: Update artist details such as name, bio, or genre based on artist_id.
    - Delete: Delete an artist record based on artist_id.
2. Album:
    - Create: Create a new album record with title, release_date, and artist_id.
    - Read: Retrieve album details based on album_id or search for albums by title or artist.
3. Track:
    - Create: Create a new track record with title, duration, and album_id.
    - Read: Retrieve track details based on track_id or search for tracks by title or album.

REST Web API Server: ARC

- Artists:
    - GET /artists: Retrieve a list of all artists.
    - POST /artists: Create a new artist.
    - GET /artists/{artist_id}: Retrieve details of a specific artist.
    - PUT /artists/{artist_id}: Update details of a specific artist.
    - DELETE /artists/{artist_id}: Delete a specific artist.

- Albums:
    - GET /albums: Retrieve a list of all albums.
    - POST /albums: Create a new album.
    - GET /albums/{album_id}: Retrieve details of a specific album.
- Tracks:
    - GET /tracks: Retrieve a list of all tracks.
    - POST /tracks: Create a new track.
    - GET /tracks/{track_id}: Retrieve details of a specific track.