Project Report

Modern Application Development - II

Introduction

The project was aimed at creating a multiuser music streaming application to meet the given requirements.

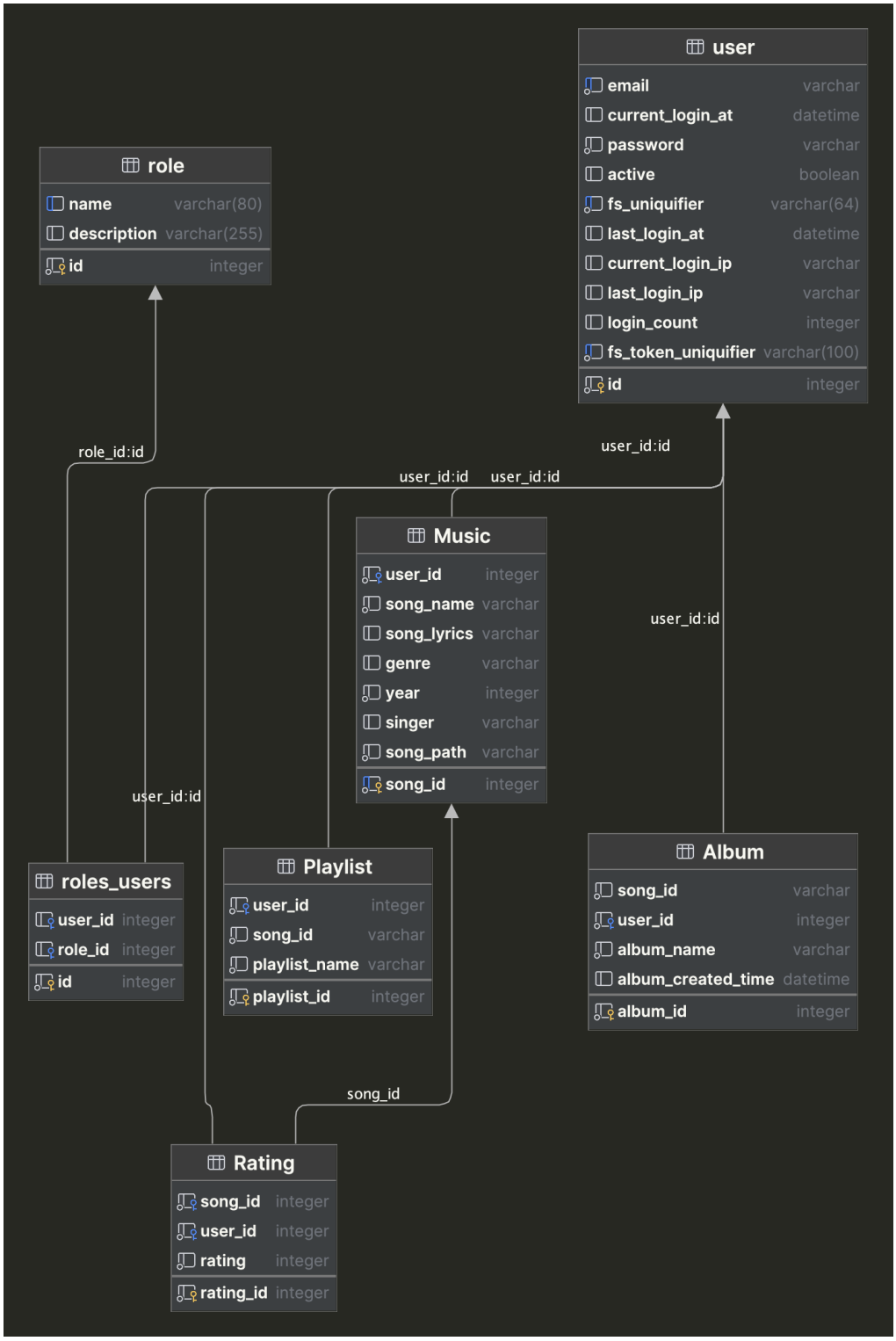
Requirements

The following are the key requirements for the music streaming application:

- Support multiple users with role based access.
- Provide administrator access and controls
- Provide a user-friendly interface for music browsing and playback
- Enable users to create and edit playlists
- Implement search functionality
- SQLite for data storage
- Redis for caching
- Celery for batch jobs
- Token based authentication
- VueJS based UI

Design

The application consists of a client-server architecture, with the client application running on the user's device and the server application responsible for managing user data and music files. The client application will communicate with the server using APIs to perform various operations, such as user authentication, music search, and playlist management. SQLite will be used on the server-side to store user profiles, playlists, and music data, etc



The user table consists of email and id which is unique to every user, it also contains the login time, login ip, hashed password and unique identifier.

The album table contains user id of the creator who created the album, a unique album id, album name and a list of songs which make up the album.

The music table consists of song name, a unique song id, song lyrics, year, singer/artist and path of the song stored in file-system.

The playlist table contains user id of the user who created the playlist, a list of song_id which makes up that playlist, playlist name and a unique playlist id.

The rating table consists of rating id, user id and song id. A user a rate the same song any number of times and the existing rating of the user will be replaced by the new one. The average rating for the song is calculated by averaging the ratings for that particular song by all users.

The roles table is used to list the role id, role name and the role description

The roles_users tables is used to map the user with the role they have and used to implement role based access.

Implementation

The application is developed Vue Components written using HTML 5 and bootstrap for CSS. The server-side application is built using FLASK (Python web framework) .SQLite is integrated into the server-side application using FlaskSQLalchemy. The graphs are created using Matplotlib .Token based authentication is implemented using flask security and tokens are also used to identify and implement role based access. Caching and batch jobs are handled using celery and redis. CDN based Vue Js is used for front end and UI. REST API are written using flask-restful.

A report is sent to the creator every month through email detailing the new albums they have created in the past month, their current average rating and a few stat graph.

Every user will get a reminder to login if they have not logged-in in the past 24hrs.

A user will receive an email on becoming a creator

Presentation video: https://drive.google.com/file/d/1EZQ7_AZmgzhXYirbcYgORE8I02tvUWdx/view?usp=sharing