Jiggly Music App

(App Dev-2 Project Report)

Author:

Name: Akash Sharma

Roll no.: 22F2000700

Email: 22f2000700@ds.study.iitm.ac.in

About: I am a diploma level student pursuing a B.S. in Data Science & Applications at IIT-Madras.

Description:

Jiggly is a multi-user music streaming web application, where a user can upload & listen to songs, create albums/playlists from the available songs and even rate, like or flag a song. It also contains admin functionality, who can remove songs or blacklist a creator according to app policy.

This app is built on Vue.js framework for client-side and Flask for server-side. Additional features include scheduled jobs, daily reminders and caching using redis and celery, and token-based authentication using flask JWT.

Technologies Used:

- Vue.js - The client side/ frontend part of the app is built on Vue.js.
- Flask - The server side/ backend part of the app is built on Flask.
- Redis and Celery are used for scheduled jobs/daily reminders via Mail Hog.
- Flask-restful for managing the API calls.
- Flask JWT for token-based authentication.

- Smtplib and MIME Multipart to send multipart messages using simple mail transfer protocol.
- SQLite3 and Flask-SQL Alchemy - to create and manage the relational database for the app.
- Chart.js - to plot the app statistics graphs for the admin dashboard.
- Bootstrap - for templates of the web pages.
- Jinja2 - for generating Monthly activity reports at backend.
- Redis for caching.

Database:

- Database models/tables for the app are created using flask-SQL alchemy.
- There are 8 Tables used in the database: Users, Role, Songs, Playlist, Songs_in_Playlist, Album, Songs_in_Album, Rating.
- Users are differentiated based on their roles using the Role table.
- Songs and Album/Playlist have many-to-many relationships.
- Songs & Rating is a one-to-many relationship.

System Design:

- This web app follows MVC architecture style: -
    - Model(M) is handled by flask. Flask interacts with the database and manages the data model.
    - View(V) is handled by vue.js. Vue components are responsible for interactive user interface.
    - Controller(C) is handled by flask. Flask routes handle all the business logic at the backend.
- backend/application/apis – the code for managing API calls.
- backend/application/models – the code for creating database tables.
- backend/application/routes – the code for login, logout, search and other app functionality.
- backend/application/worker, tasks – the code for schedule jobs & daily reminders.
- backend/instance - stores the database of the app.
- backend/celeryconfig, config – the code for celery & app configuration.
- backend/initial_data - the code of some pre saved data of the app.

- backend/main - the code for the flask app instance, celery app instance and initializing api & database for the app.
- backend/requirements.txt - store required dependencies.
- backend/Readme - contains instructions/command to run the server side.
- frontend/public/audio - stores all the audio files.
- frontend/src/assets – stores the base & main CSS files.
- frontend/src/components – contains all the vue.js components.
- frontend/src/router/index.js – contains code for all the routes & navigation guard.
- frontend/src/stores - contains store for user state-management and authorisation.
- frontend/src/main – contains code for Vue app & store initialisation.
- frontend/package – contains all the npm dependencies.
- frontend/Readme – contains instructions/command to run the client side.

Features Implemented:

- Separate login form for users and admin.
- Admin dashboard with app statistics and graphs.
- Admin can manage songs, albums, remove flags and blacklist creators.
- Admin/User/Creator can search for songs based on song title.
- Users/Creator can filter songs based on rating.
- Users/Creator can play songs, read lyrics, like/dislike, rate & flag songs, create, edit, delete playlists.
- Creators can upload, edit and delete songs and albums.
- User/Creator can change password and delete their account.
- Monthly Activity report of creator is sent to creators' email on first day of month.
- Daily notifications on email to users to visit the app if inactive for more than 24 hours.

Presentation Video: Click here to view...