Project Report

Introduction

This report provides an overview of the system design and architecture for a web application developed using Flask REST API, Vue.js, JWT authentication, SQLite database, Celery, and Redis. The system aims to automate email reminders for users who have not visited the website recently and generate monthly reports.

System Components

Flask REST API

- Utilized for building the backend of the application.
- Handles HTTP requests and responses.
- Implements JWT authentication for user authorization.

Vue.js

- Frontend framework used to create interactive user interfaces.
- Enables dynamic rendering of content and seamless user interactions.

SQLite Database

- Lightweight and efficient relational database management system.
- Stores data related to users, songs, albums, comments, playlists, etc.

Celery and Redis

- Celery is used for asynchronous task execution.
- Redis acts as a message broker to facilitate communication between Flask and Celery.
- Tasks include sending reminder emails to users and generating monthly reports.

JWT Authentication

- JSON Web Tokens are employed for secure user authentication.
- Users receive tokens upon successful login, which are then used to access protected endpoints.

Model Overview

User

- Stores user information such as name, email, password hash, and role.
- Associated with roles to determine access permissions.

Role

- Defines user roles such as admin, creator, or regular user.
- Used for role-based access control (RBAC) within the application.

Song, Album, Comment, Playlist

- Models representing various entities in the application.
- Store relevant attributes and relationships between entities.

RecentlyPlayedSongs, SongRating, ReportedSong

- Additional models capturing user interactions with songs and reporting functionality.

System Workflow

Users interact with the frontend Vue.js application to browse songs, albums, and playlists.
Flask REST API handles incoming requests, authenticates users using JWT, and interacts with the SQLite database to retrieve or manipulate data.
Celery tasks are triggered asynchronously to perform background operations such as sending reminder emails and generating reports.
Redis acts as a message broker, facilitating communication between Flask and Celery for task execution.
Monthly reports are generated based on user activity and emailed to relevant stakeholders