POPIFY MUSIC APPLICATION

A person lying on a bed with headphones

Description automatically generated

FSD-08 Integration Project 2023-4

Developed By  
Shaunpaul Anthony

Dustin Ruck

Dominique Anthony

Lewis Innes-Miller

Pejman Shahmohammadhamedani

Miral Patel­

Contents

[**PROJECT DESCRIPTION:** 3](#_Toc152750785)

[**TECHNOLOGIES USED:** 4](#_Toc152750786)

[**SPECIAL FEATURES:** 4](#_Toc152750787)

[**ADDITIONAL LIBRARIES:** 4](#_Toc152750788)

[**DESIGN OF THE DATABASE:** 5](#_Toc152750789)

[**USE CASE DIAGRAM:** 6](#_Toc152750790)

[**MOCKUPS:** 7](#_Toc152750791)

[**FURTHER QUESTIONS:** 9](#_Toc152750792)

|  |
| --- |
| **PROJECT DESCRIPTION:** |

The project is a full-featured music web application that allows users to find songs, create playlists, and even upload their own content. The application has use cases for three different types of users: regular users, creators, advertisers, and administrators.

**1** — Regular users can:

* Register by filling a specific form with their personal information, uploading their picture and paying the membership fee (the registration form has appropriate validations)
* Receive Email confirmation of the payment and approves the registration
* Have a personal profile page (which is created automatically after the registration)
* Edit their personal profile page
* Choose different themes from their personal profile page
* See and remove songs from the list of their favorite songs in their profile page
* Create their personal playlists from their favorite songs, based on their genres or categories and see, edit and remove them in their profile page
* Playlists can be public or private, public playlists are viewable and listenable by other users aside from the creator of the playlist.
* Browse the application and see all the available songs in paginated format pages and listen to them (the songs are playable and users can play, pause and skip to the next/previous song)
* Filter/sort the songs based on the song name, album name, artist name, genre, category, number of likes, max/min length, year and …
* Search for a song/artist/album
* Like/dislike the available songs, leave comments for them, send them to their list of favorite songs (by clicking on the heart icon)
* Receive playback information, e.g. which songs they have listened to most and for how long, which artists are frequently listened to, and so on
* See the first 20 most globally popular songs based on the number of likes
* Logout

**2** — Creators (who are artists and want to share their songs to the world) can have all the options that regular users have (except that when they register as a creator, they should wait for the confirmation of one of the admins), and also they can:

* Upload their songs and albums so that all the users could discover them
* See, edit, and delete the songs and albums they’ve uploaded to the website
* See the number of likes and comments for each of their songs and albums, a kind of analytics for creators

3 — Advertisers can:

* Upload mp3’s of their advertisements in audio form
* Receive approval for an advertisement from administration
* Retrieve data on their ads, e.g. how many times their ads have been played etc
* Remove their uploaded ads from circulation

4 — Administrators can:

* Approve the registered creators
* Disable users (which means users can’t login/register with the same Email) and Enable them again
* Create, read, update, and delete all the users, songs, albums, comments, and subscriptions
* Manage the advertisements with approval/disapproval

|  |
| --- |
| **TECHNOLOGIES USED:** |

- Node.js  
- Vue  
- JavaScript

- Sequelize or TypeORM  
- SQL Database (Azure)  
- Blob storage (Azure)  
- Jira   
- GitHub  
- CICD  
- Google ads (mock ads)

|  |
| --- |
| **SPECIAL FEATURES:** |

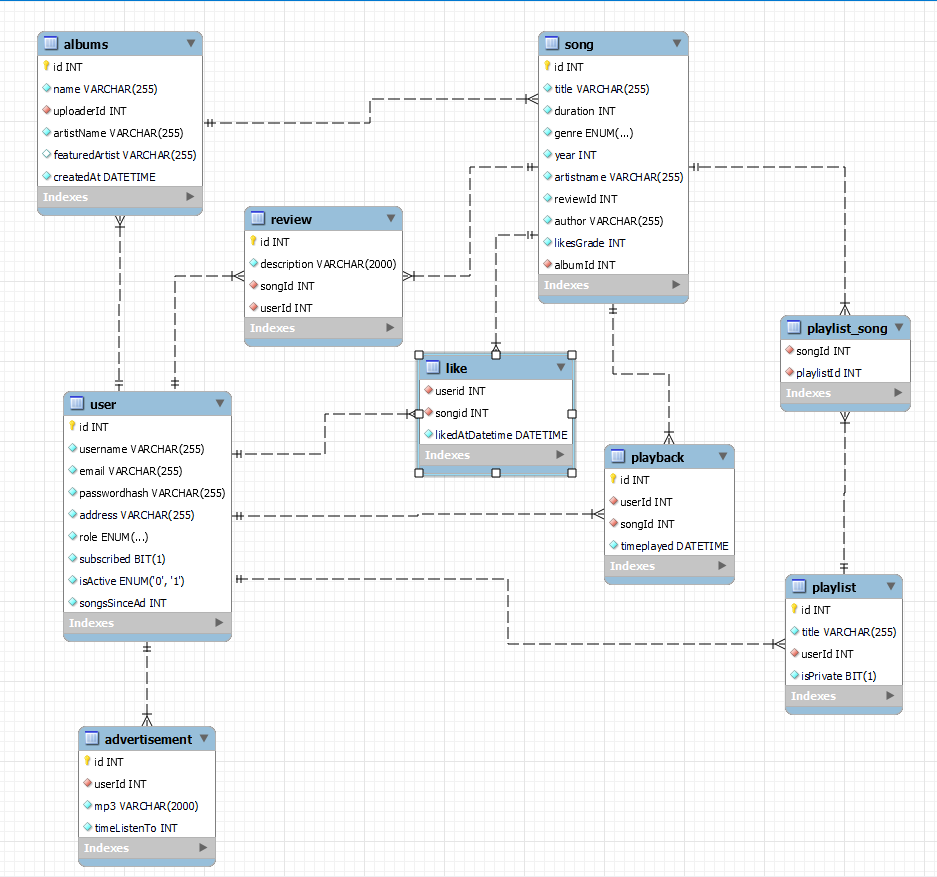
- Advertisements  
- Email confirmation  
- Google Charts  
- D3.js  
- Burndown chart

|  |
| --- |
| **ADDITIONAL LIBRARIES:** |

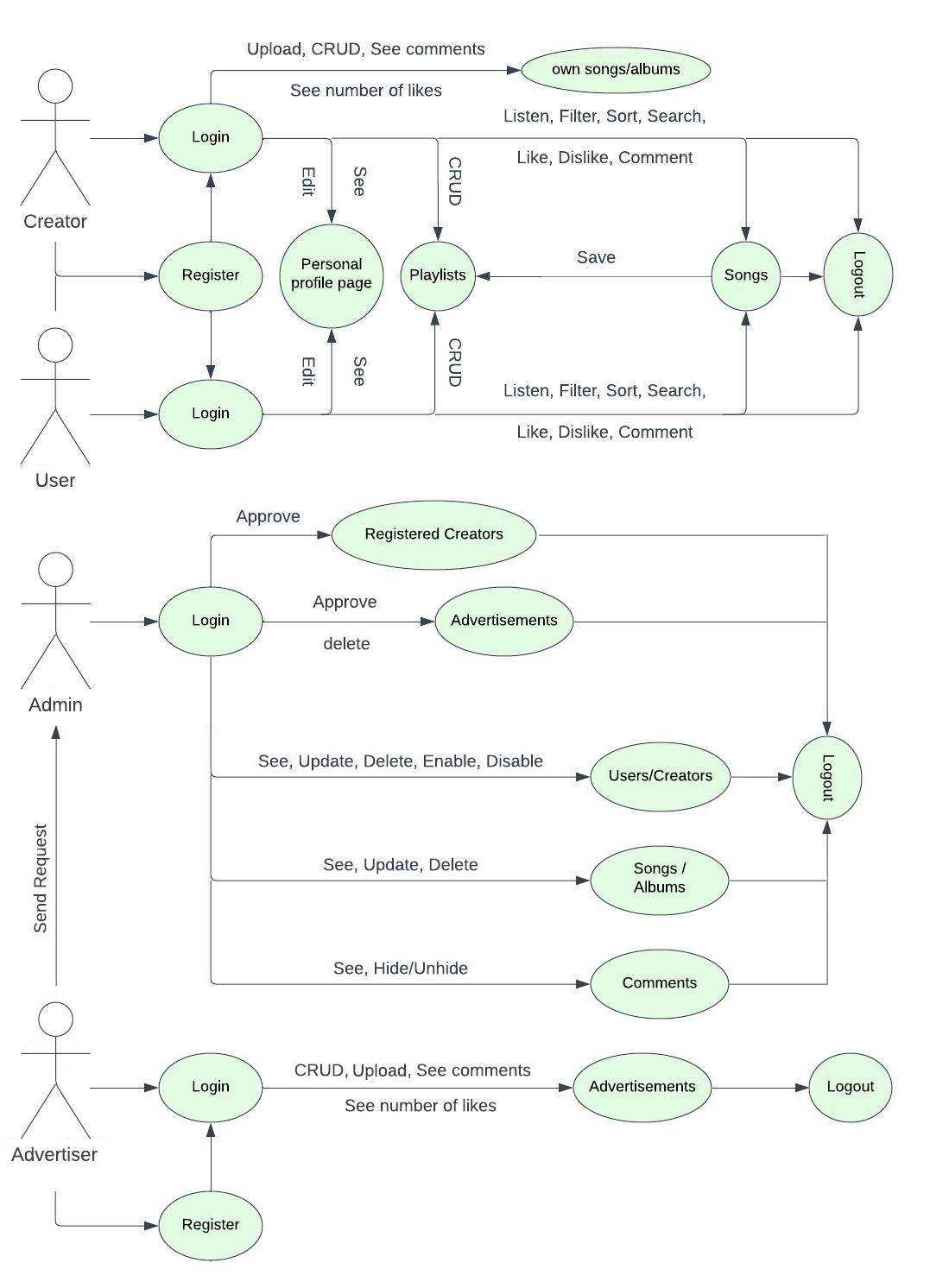
- Stripe API/Paypal  
- Express.js  
- Multer  
- JSX  
- TypeScript  
- Vite  
- Vue Router  
- Prettier  
- Bootstrap Vue

-if time allows, we would like to add libraries which enable multiple languages on the application.

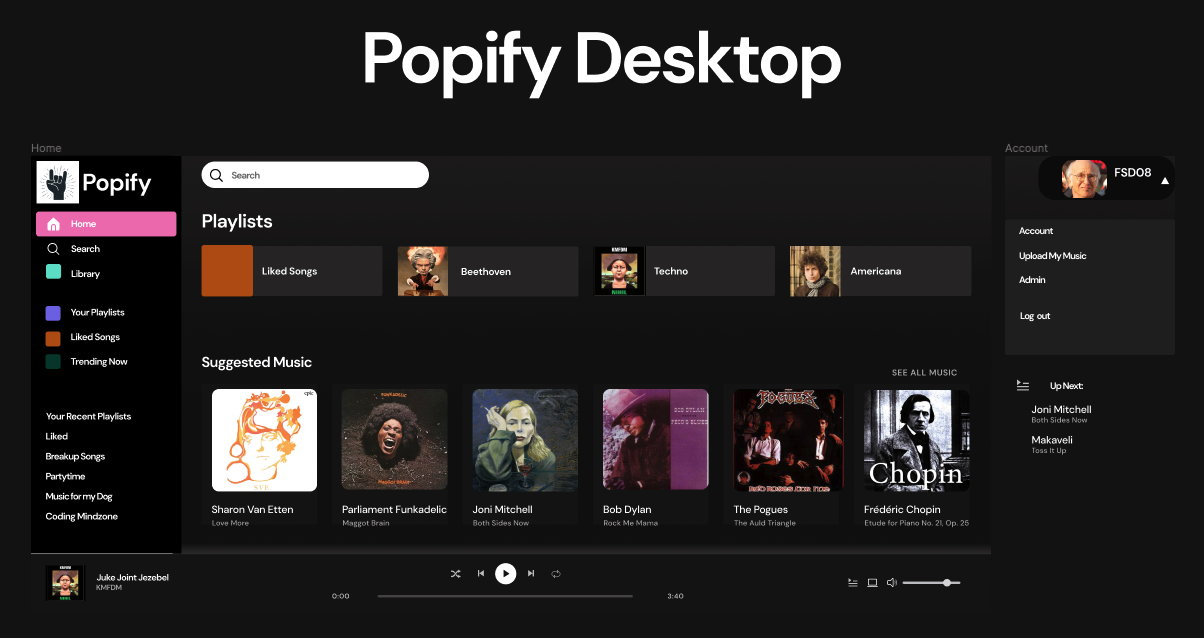
|  |
| --- |
| **DESIGN OF THE DATABASE:** |



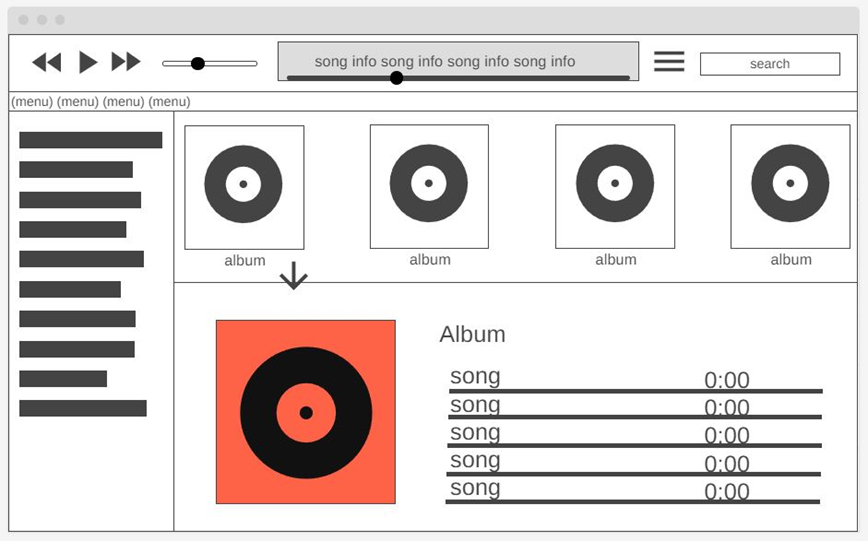
|  |
| --- |
| **USE CASE DIAGRAM:** |



|  |
| --- |
| **MOCKUPS:** |



A screenshot of a music player

Description automatically generated  
  
  
  
  


|  |
| --- |
| **FURTHER QUESTIONS:** |

1. We want to have a field in the songs table which is recalculated every time a user likes the song. LikesGrade – this will be a ratio of likes to dislikes, and so will be the result of SELECT count(\*) WHERE songId=… query on the likes table. Should we just send these queries to calculate LikesGrade on the Frontend or should we have the calculation occur in the database and store the result in the songs table?

2. What should we plan for payment and transaction storage? Should we store transaction information in our database, and if so, how much transaction information should we store? How much information can we expect to get from Stripe?

3. What is the relationship between docker and CI/CD technologies like Jenkins and GitLab?

4. What is a DTO?