

REFLECTIVE REPORT ON FIND TUNE

For ST5007CEM Web Development

SUBMITTED TO

Hari Sharan Shrestha

UNIVERSITY ID

11781682

SUBMITTED BY

Aayush Pandey

YOUTUBE LINK

<https://youtu.be/PqZvN3rBjnc>

REPOSITORY LINK

<https://github.com/AlexxyQQ/FindTune>

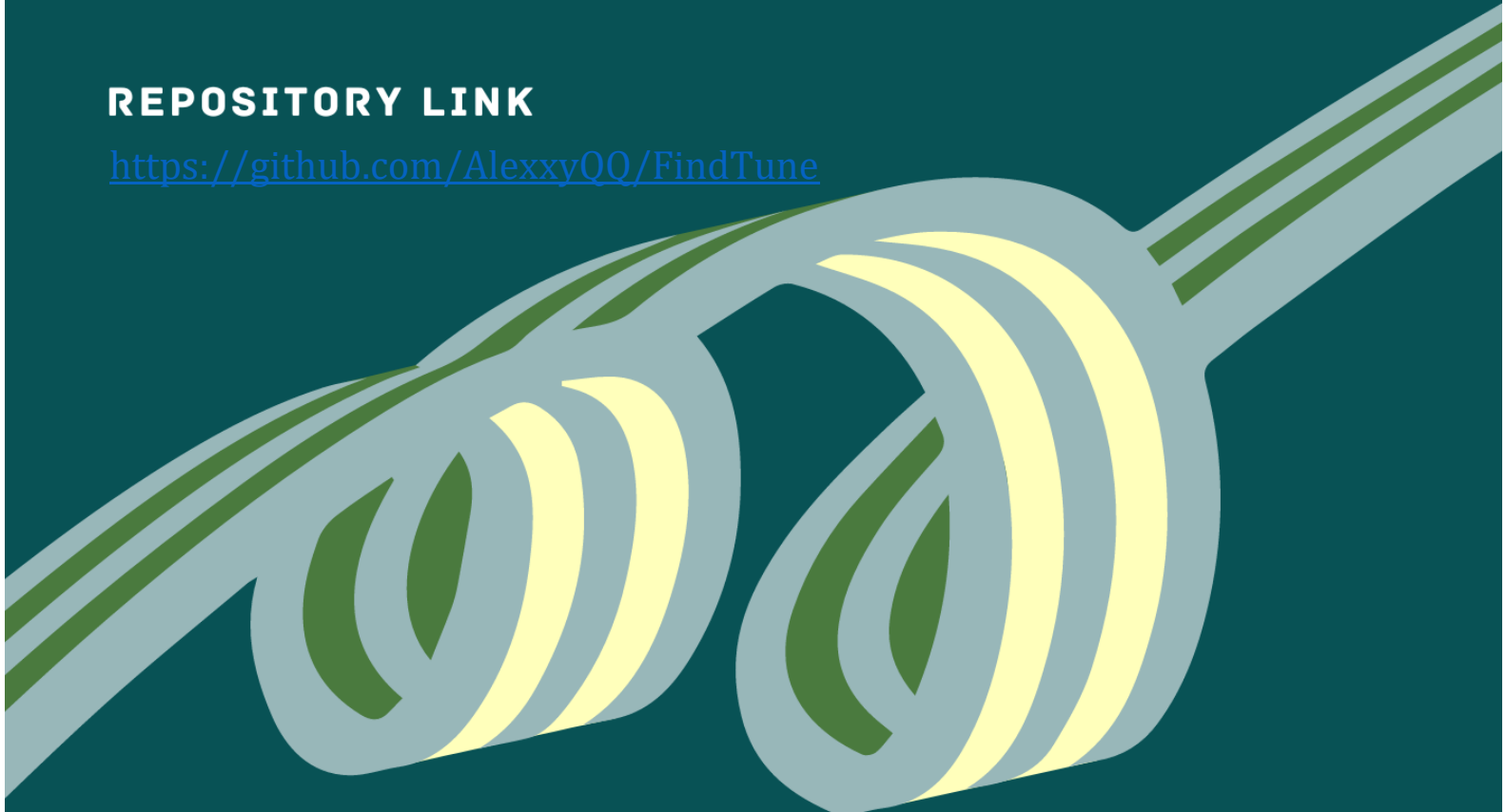


Table of Contents

Table of Figures	3
Introduction.....	4
Brief Introduction of the System	4
Feature of the System	4
Aims	5
Objectives	5
What I learned	6
Frontend.....	6
Technologies.....	6
Screenshots	8
Importance of responsive design	11
Responsive Check.....	13
Database.....	19
Introduction.....	19
Advantages and disadvantages of MySQL.....	20
Screenshots	21
Backend	21
Introduction.....	21
Technologies.....	22
Screenshots	23
Project Issues	25
Issues during the project development	25
Limitations of your project	26
Future works	26
Diagrams	27
Conclusion	28
References	28

Table of Figures

Figure 1. Frontend Homepage	8
Figure 2. Frontend Homepage 2	8
Figure 3. Frontend Song Searched	9
Figure 4. Frontend Song Searched 2	9
Figure 5. Frontend User Library	10
Figure 6. Frontend User Library 2	10
Figure 7. Frontend Search Box	11
Figure 8. Frontend Searched Content	11
Figure 9. Responsive Homepage	13
Figure 10. Responsive Song Page	14
Figure 11. Responsive Song Page 2	15
Figure 12. Responsive Hamburger menu	16
Figure 13. Responsive User Library	17
Figure 14. Responsive Login Signup Page	18
Figure 15. Database ERD	21
Figure 16. Backend Views.py	23
Figure 17. Backend Views.py 2	23
Figure 18. Backend Views.py 3	24
Figure 19. Backend Form.py	24
Figure 20. Backend auth.py	25
Figure 21. Use Cas Diagram	27
Figure 22. Rich Picture	27

Introduction

Brief Introduction of the System

Find Tune is a PWA (Progressive Web App) which makes the website behave as a mobile application and lets users install the app. Being a progressive web app helps the website to expand to a wider audience. Rather than always opening the browser and going to the link, PWA just cuts the process and lets the user install the website as an app itself, which makes it far more efficient and easier to access. PWA especially helped Find Tune as it is a quick needed app where users need to quickly record the song playing in the surroundings and let the website process the data.

Find Tune's main premise is to help user to find the song's lyrics, info, and videos if available. The song recognition is achieved by using the audio fingerprinting algorithm to find the songs that it records and tally the records to the world's leading songs information database of Shazam. The audio fingerprinting works by recording the relevant frequencies of the song and tallying it to the database. After the song is recognized, the information is stored in the local database and the user is displayed the information about the song on the website.

If the song's details are missing users can add them and help the community. Logged-in users can add the missing lyrics and can also vote on the already existing lyrics. The most voted lyrics are then displayed at the top of the lyrics section and others are displayed below accordingly. The website also supports song recognition technology by listening to a certain section of the song and finding out the information about it. Overall, the website is mainly focused on the song and their information which might be important to the users.

Feature of the System

1. Song Recognition:

Song recognition is one of the main features of the website. The user can just record the audio and then the audio is processed, and the user is greeted with the song information if available. The feature works by generating audio fingerprints and tallying it to the database. The fingerprints are the audio signals which are converted and plotted in the spectrogram. The spectrogram then is compared with other spectrograms stored in the database and when the match is found the corresponding information about the song is outputted.

2. Song Information:

Displaying song's information was the main objective of the website. This feature is achieved by the help of audio fingerprint matching and displaying out the corresponding information about the song. The website finds the match and then displays the data which is extracted from the database. The database stores the name, artist, album name, released date and lyric of the song. The data are stored in database with a normalized table which makes the extraction faster and easier.

3. User Library:

User library is a quality-of-life feature which helps user to quickly go through the songs that they have previously searched or found. This is achieved by assigning a separate database table to the user which stores all the information about user.

4. Lyrics Upload, Edit and Delete:

Displaying lyrics of the song is one of the most important part off the Find Tune experience. Find Tune supports multiple lyrics to a single song which user can vote, and the most voted lyric is displayed at the top and rest are collapsed down. User can also edit the uploaded lyrics and delete as well.

5. Lyric Vote:

The lyrics is displayed according to the vote. So voting is the crucial part of lyric displaying. One user can only vote once to the lyric of a particular song.

Aims

The main aim of Find Tune is to find the song which user is looking for and display the information of the song and let user to add and download the lyrics of the songs.

Objectives

- Develop and design efficient web application with a simple user-friendly UI
 - To store the previously identified songs in the user library
 - To let user to edit the lyrics of the song if there are any mistakes
 - To set the best voted lyric to be displayed in the song page
 - To let user to add the lyric for the songs whose lyrics are not identified
- User can add lyrics to the songs and the most voted lyric is displayed in the song info page and other lyrics are displayed below with read more button.
- To let user download lyrics for the song

What I learned

The ever-evolving field of web development is very vast and almost never ending. Find Tune was my first ever web development project. So, this was a very interesting and new experience for me. As this project is done using Flask a Python web development framework which is very lightweight and customizable, I had many fun and frustrating experiences with it. I learned a lot of new things while developing this project. As I started this project early on, I was quite focused on the frontend part of the project due to my fluency in CSS and HTML. As I went on further into the project, I got the opportunity to learn about the web scripting language JavaScript which was quite complicated due to lack of understanding of how elements work and DOM tree. This problem got me fired up to learn more about the things I was poor on. Web development gave me the opportunity to improve my HTML, CSS, and JavaScript. As the project is done using python and its web development framework Flask, I got the chance to revise my python skills and got the opportunity to learn Flask and web development on python. Flask is a lightweight web development framework that is very customizable. I have used one of the most common and properly laid out design pattern MVT (Model View Template). As I had worked on MVC (Model View Controller) pattern before I had the basic knowledge about the design pattern. I also learned how a general website is made and how it works. As the project was completed, I had a lot of knowledge and list of references about web development.

Frontend

Technologies

Frontend is one of the crucial parts in any part of software development. Frontend is the first thing that a user sees and interacts with, so it needs a lot of consideration and layout. There are a lot of technologies that help achieve a good and thoughtful frontend. Some of the technologies that I used for this project are listed below with their proper descriptions. (["Front End Developer – What is Front End Development, Explained in Plain English", 2022](#))

- **Figma**
Figma is a frontend design tool which helps user to design and preview the prototypes of the projects. For the development of this project Figma really helped layout the user interface and user experience. Figma provides a very easy to use interface which is very seamless.
- **Photoshop**
Photoshop is an image manipulating tool which helps to create and manipulate images. Photoshop is the tool that is used to create the icons and image in the project.
- **HTML**
Hyper Text Markup Language is the language used in creating webpages. It is a static

programming language which helps to add elements on the webpage. The elements tell the browser how to display the content on the webpage. Html is the base for any web development project and this project is no exception.

- CSS

Cascading Style Sheet is a styling language which helps to change the displaying and styling properties of the HTML elements. CSS is crucial when it comes to styling and making the front end look like the user wants. CSS contains classes and ID's which can be assigned to the elements of the HTML and according to the name of the classes or the ids unique a styling property can be given to the website. Find Tune uses the custom CSS classes to style various elements of the website. The responsiveness of the website also comes from the CSS.

- JavaScript

JavaScript is also one of the frontend technologies which is used to add scripts to the elements of the HTML. It runs in browser and every interaction is handle by the JavaScript. The recording function of the website is also handled by JavaScript. The recording is recorded and then saved into the storage and the backend processes the data and sends the output to the frontend. Other functions like button clicks and interactions are also handled by JavaScript

- Bootstrap

Bootstrap is a CSS framework, which means it has premade classes which proved the styling that general users are looking for. Bootstrap is used very rarely in the project due to it generic and hard to remember classes names.

Screenshots

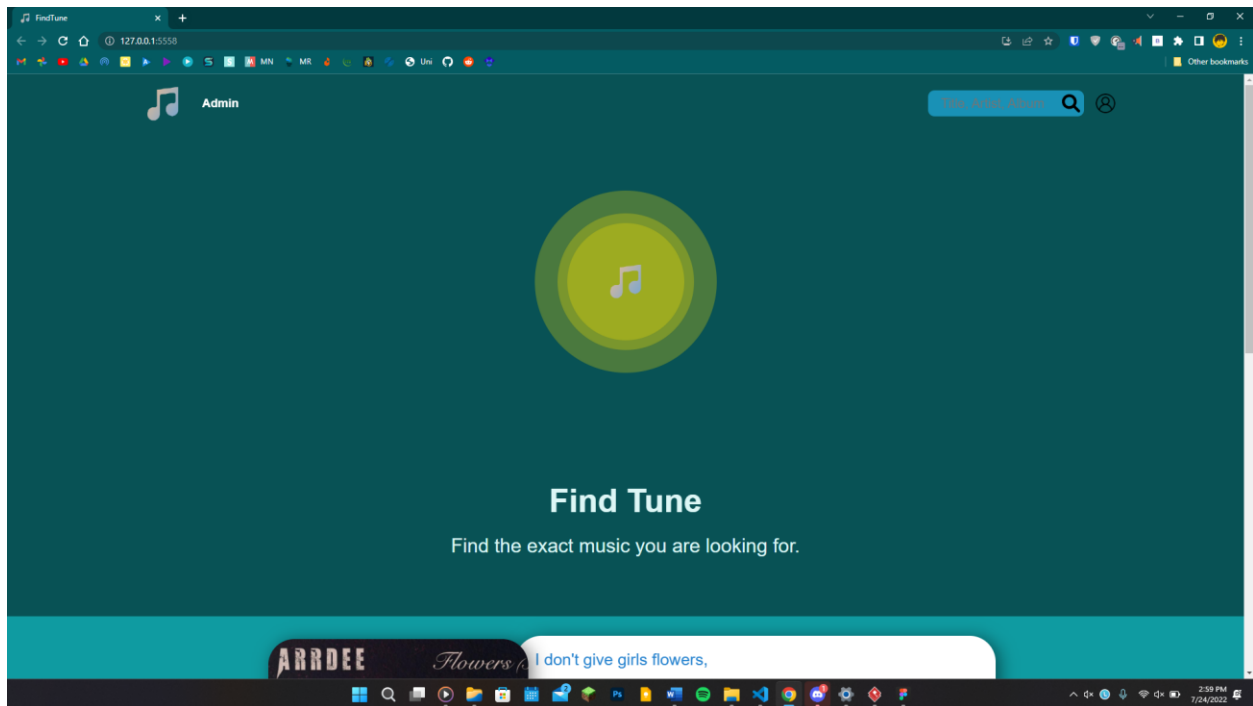


Figure 1. Frontend Homepage

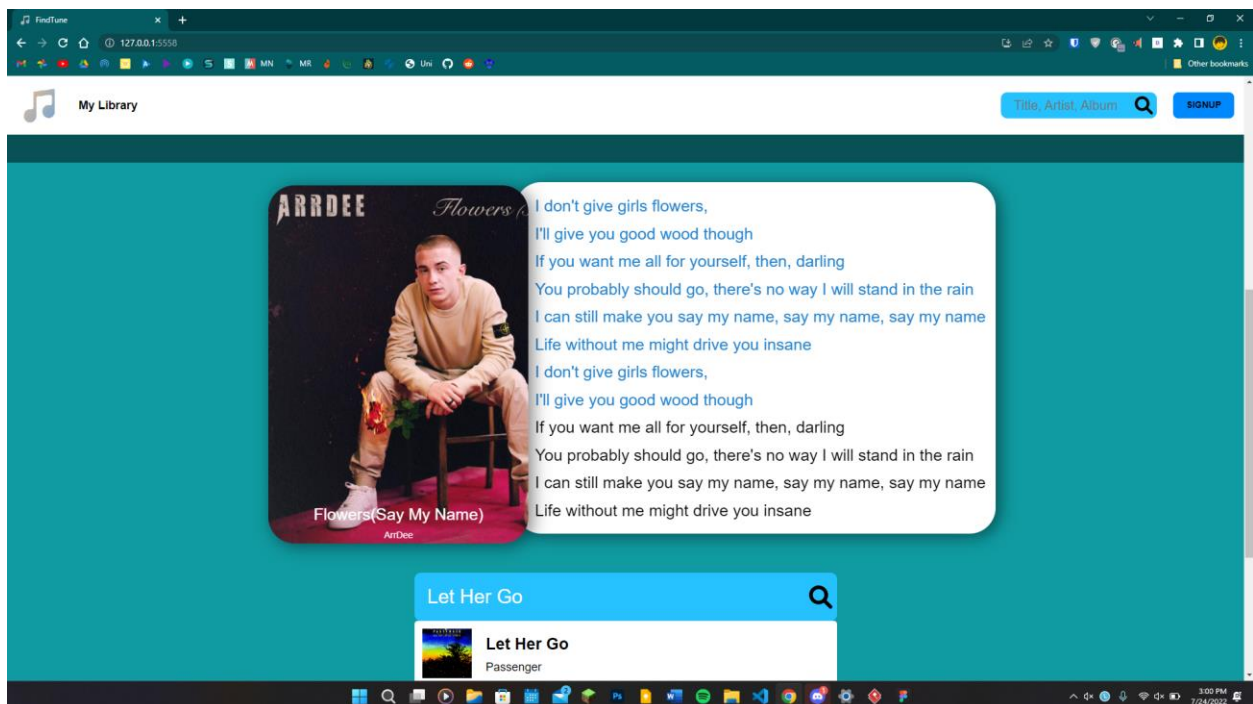


Figure 2. Frontend Homepage 2

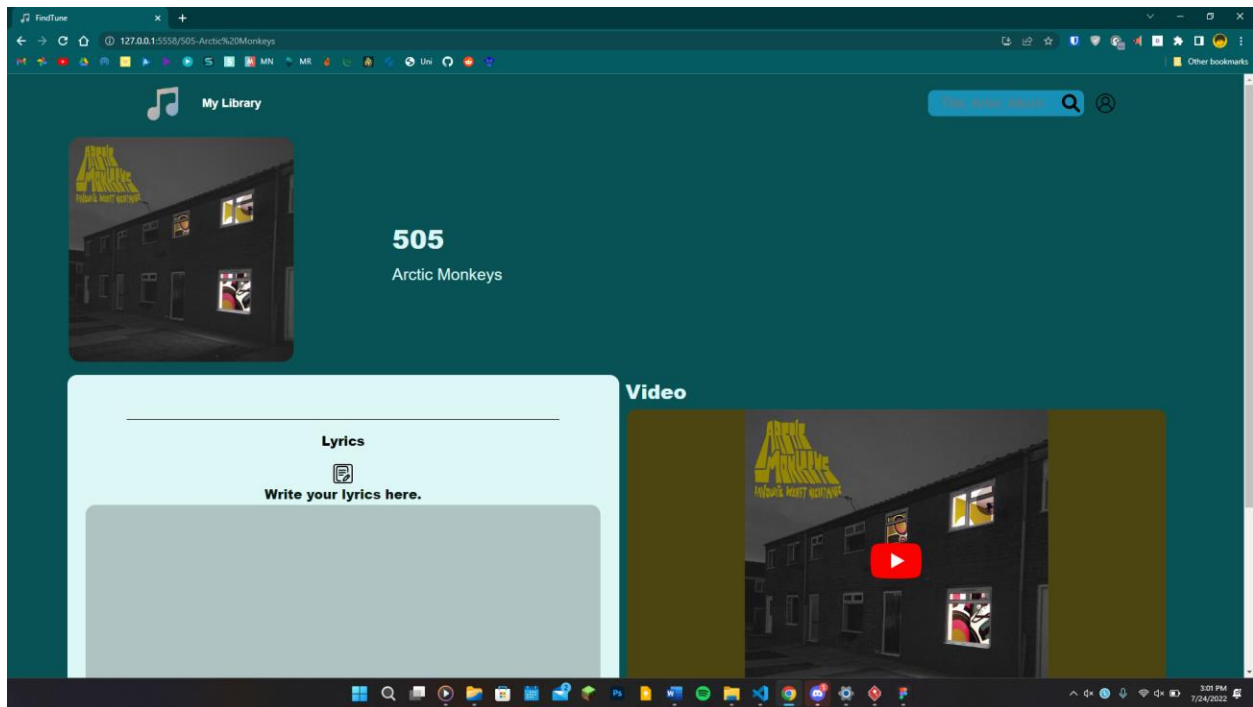


Figure 3. Frontend Song Searched

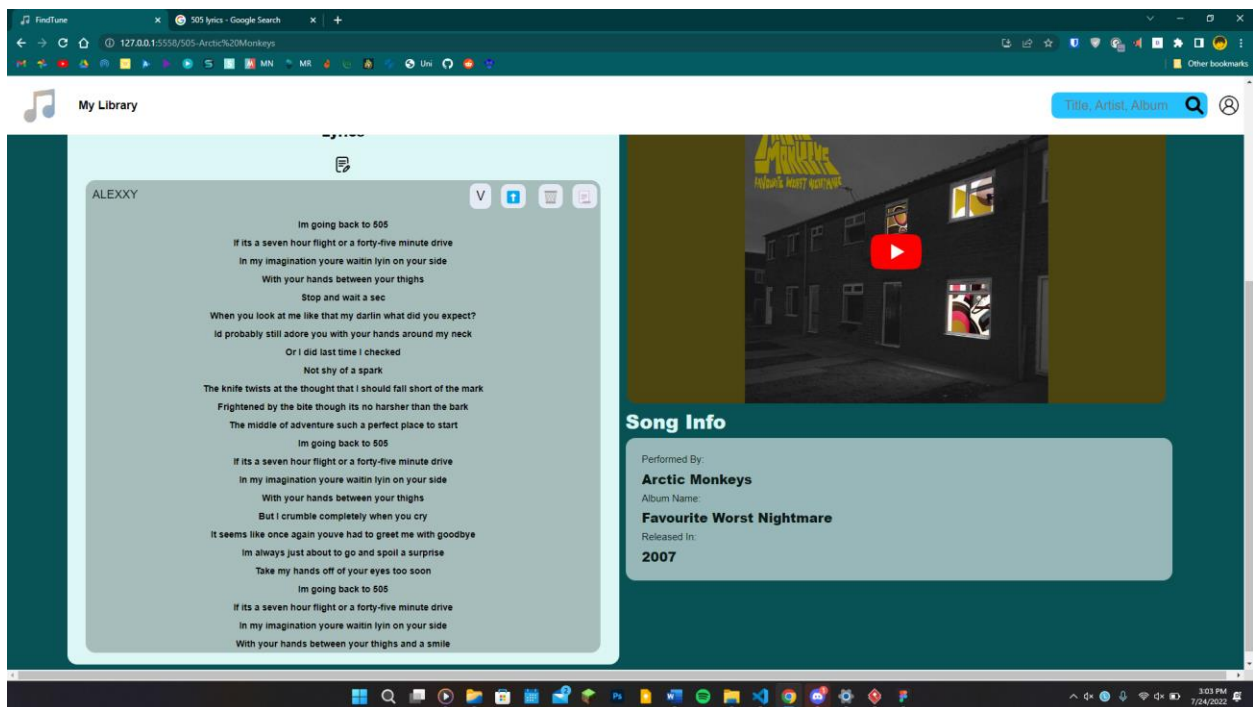


Figure 4. Frontend Song Searched 2

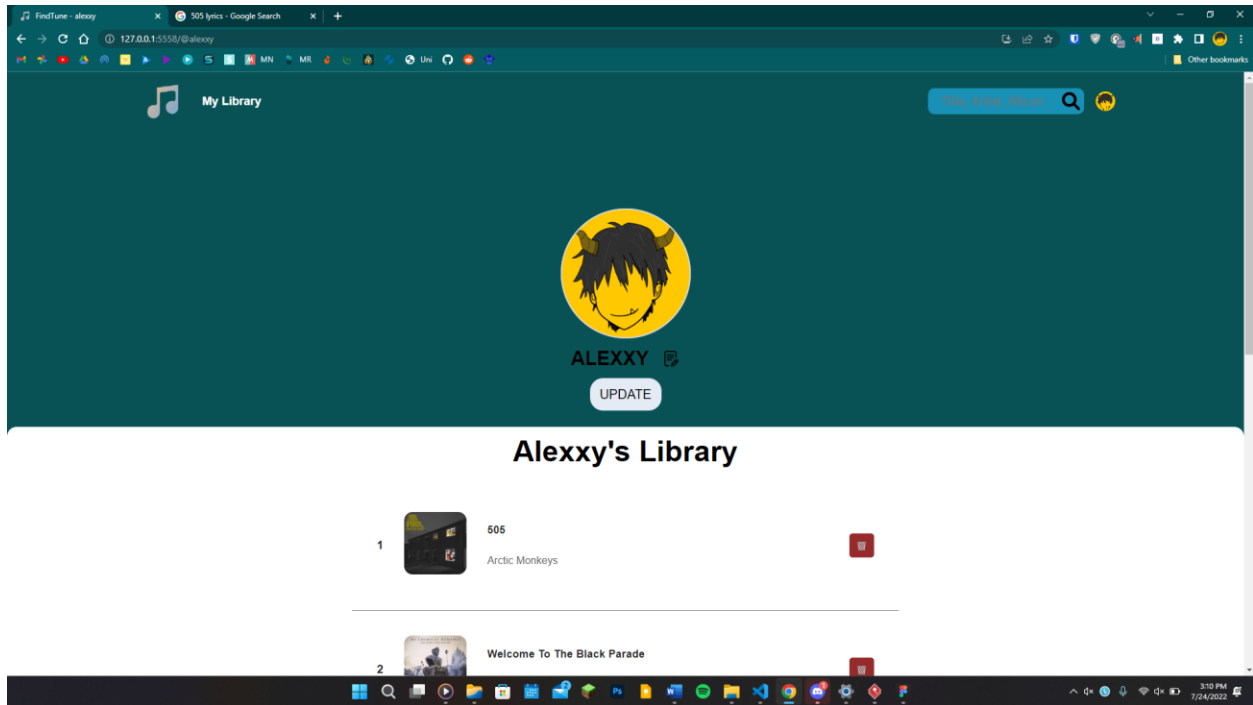


Figure 5. Frontend User Library

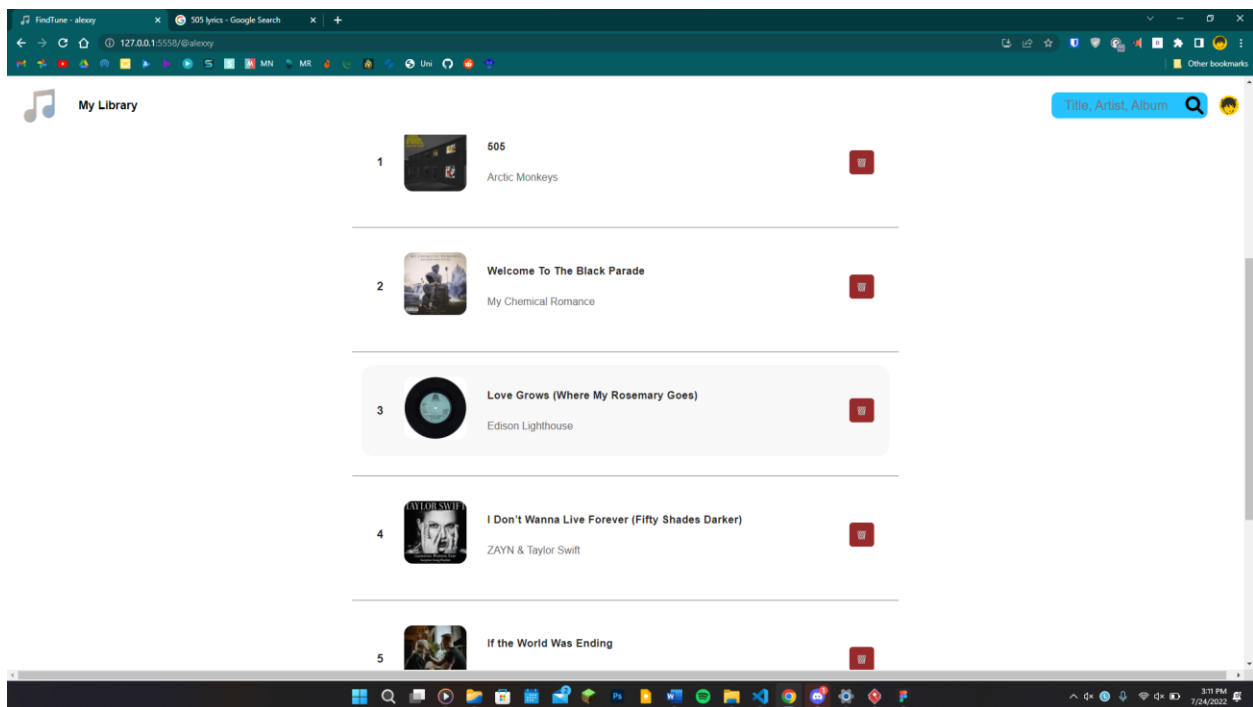


Figure 6. Frontend User Library 2

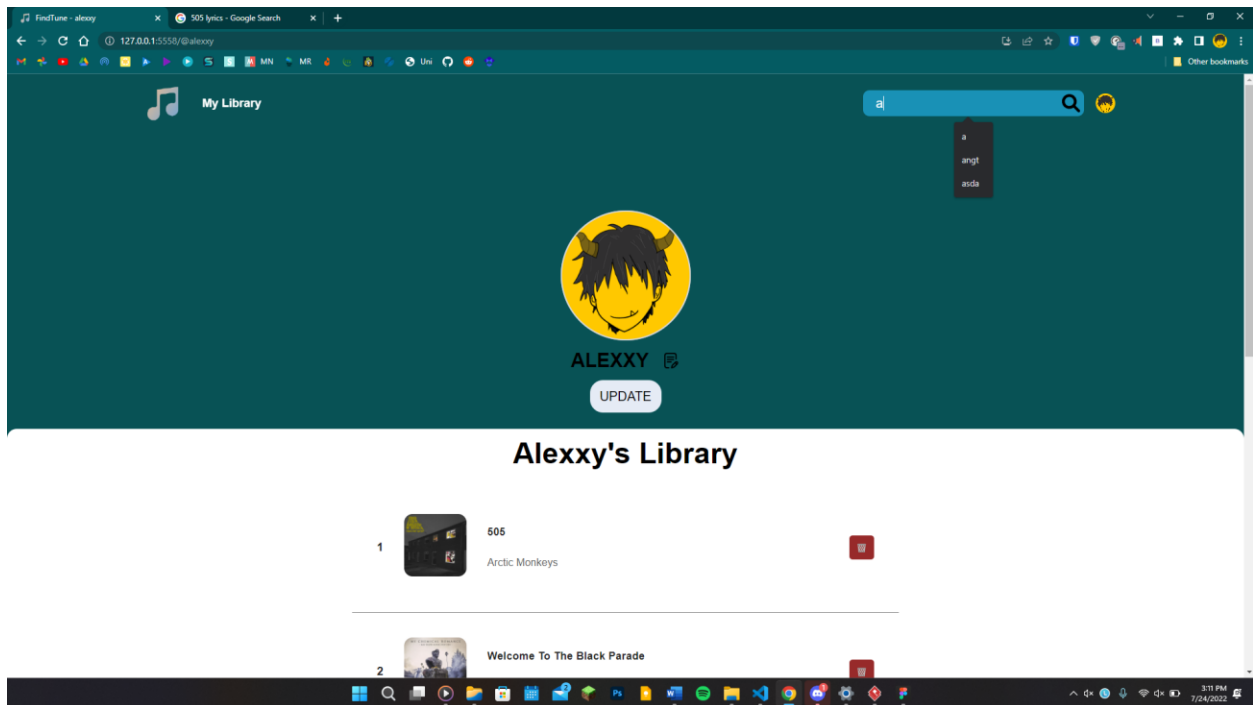


Figure 7.Frontend Search Box

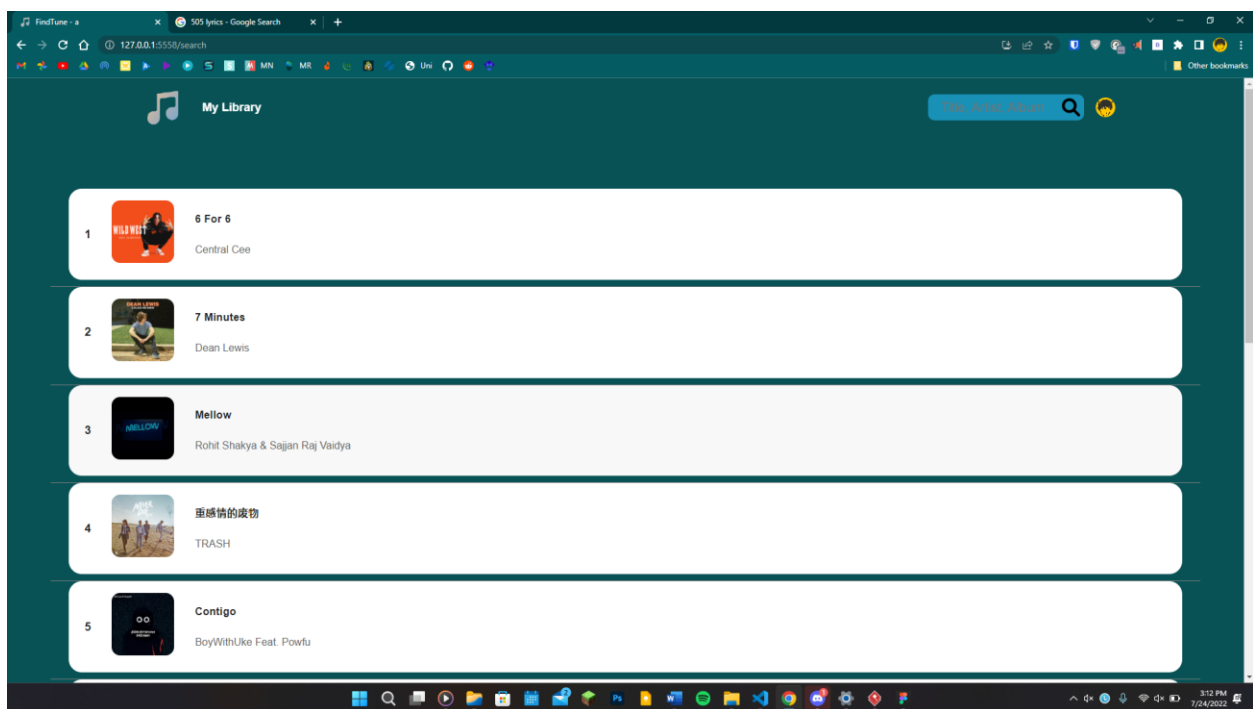


Figure 8.Frontend Searched Content

Importance of responsive design

A website is called responsive when it can adapt to the changes in screen size. A fully responsive website can look as intended in every device that the website

is opened in. The responsiveness of a website also says how much consideration is put on to the frontend development. In 2022 there are many devices around the world and almost every device got screen attached to it, Phones, Laptop, Desktop, TVs, Cars, Refrigerators etc. So, if user opens the website with any of those devices the experience should not be hindered, and the layout should resemble the original design. (["Responsive Web Design - What It Is And How To Use It — Smashing Magazine", 2022](#))

The responsiveness of a website can be achieved with the help of CSS. CSS helps to layout the HTML element on the screen and the position and size of those elements can be changed and tweaked accordingly the size of the screen. The “@media” feature of CSS helps us to set the size of the screen and the classes inside the designated screen size behaves accordingly to the reference of the class when the screen size is as same as the assigned sized.

The Find Tune also have responsive design to maximize the user experience. There are 3 different screen size limitation set up, one for the big screens, one for the phone screens and one for the rest of the devices that falls in the middle. Responsive is one of the import design solutions for maximum the user experience.

Responsive Check

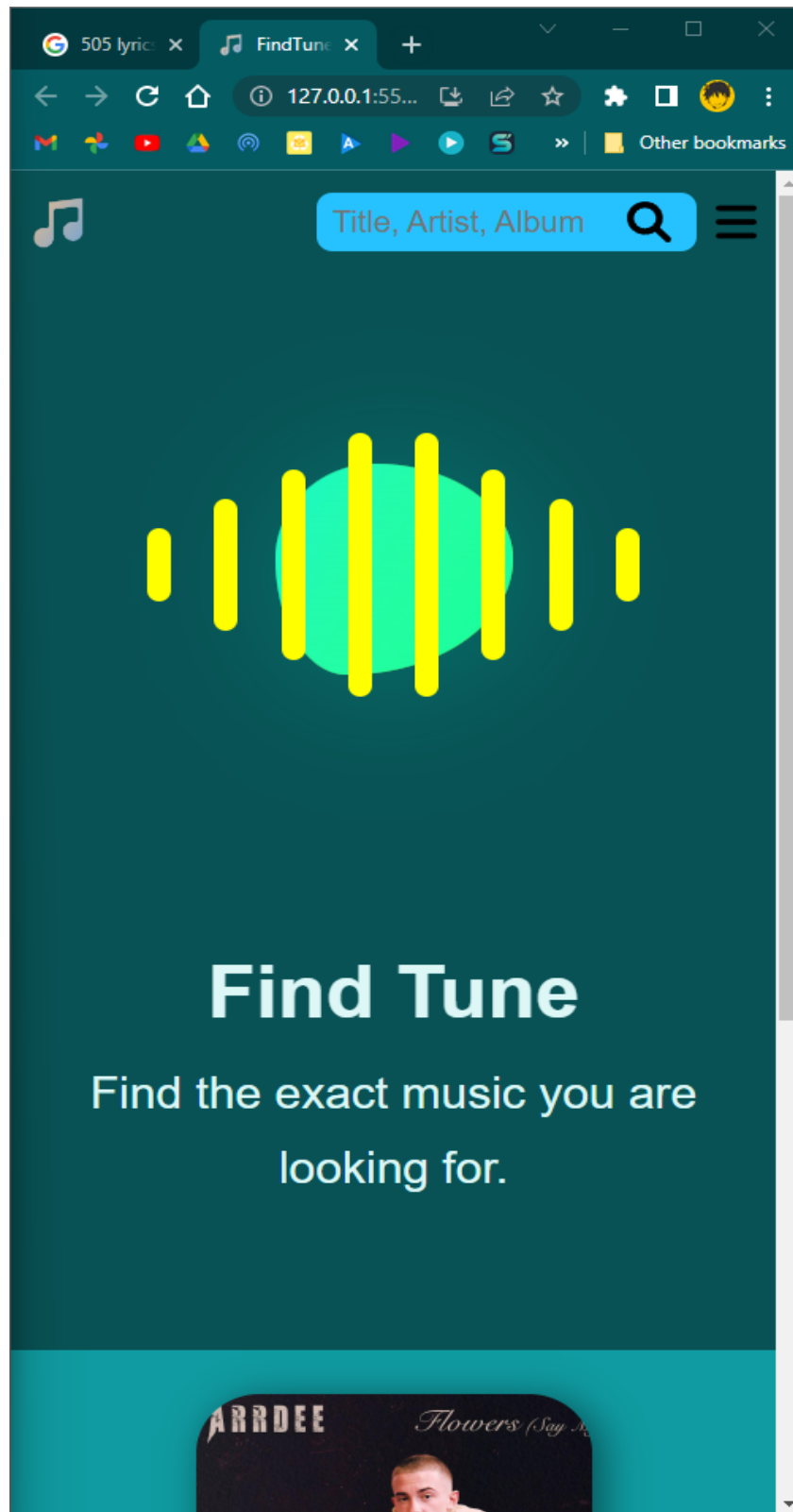


Figure 9. Responsive Homepage

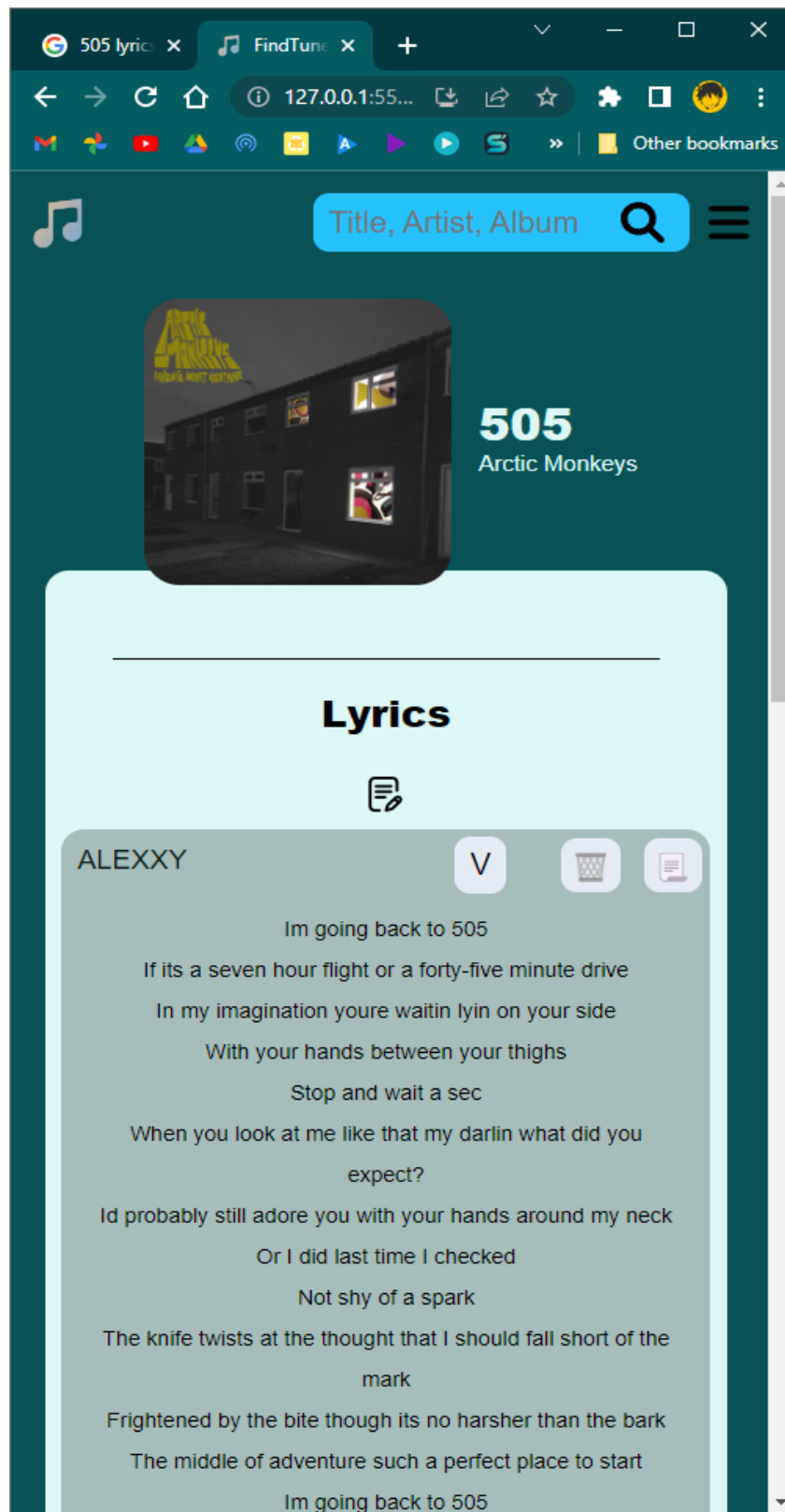



Figure 10. Responsive Song Page

505 lyric x FindTune x +

127.0.0.1:55...

Other bookmarks



Song Info

Performed By:

Arctic Monkeys

Album Name:

Favourite Worst Nightmare

Released In:

2007

<https://youtu.be/qU9mHegkTc4?autoplay=1>

Figure 11. Responsive Song Page 2

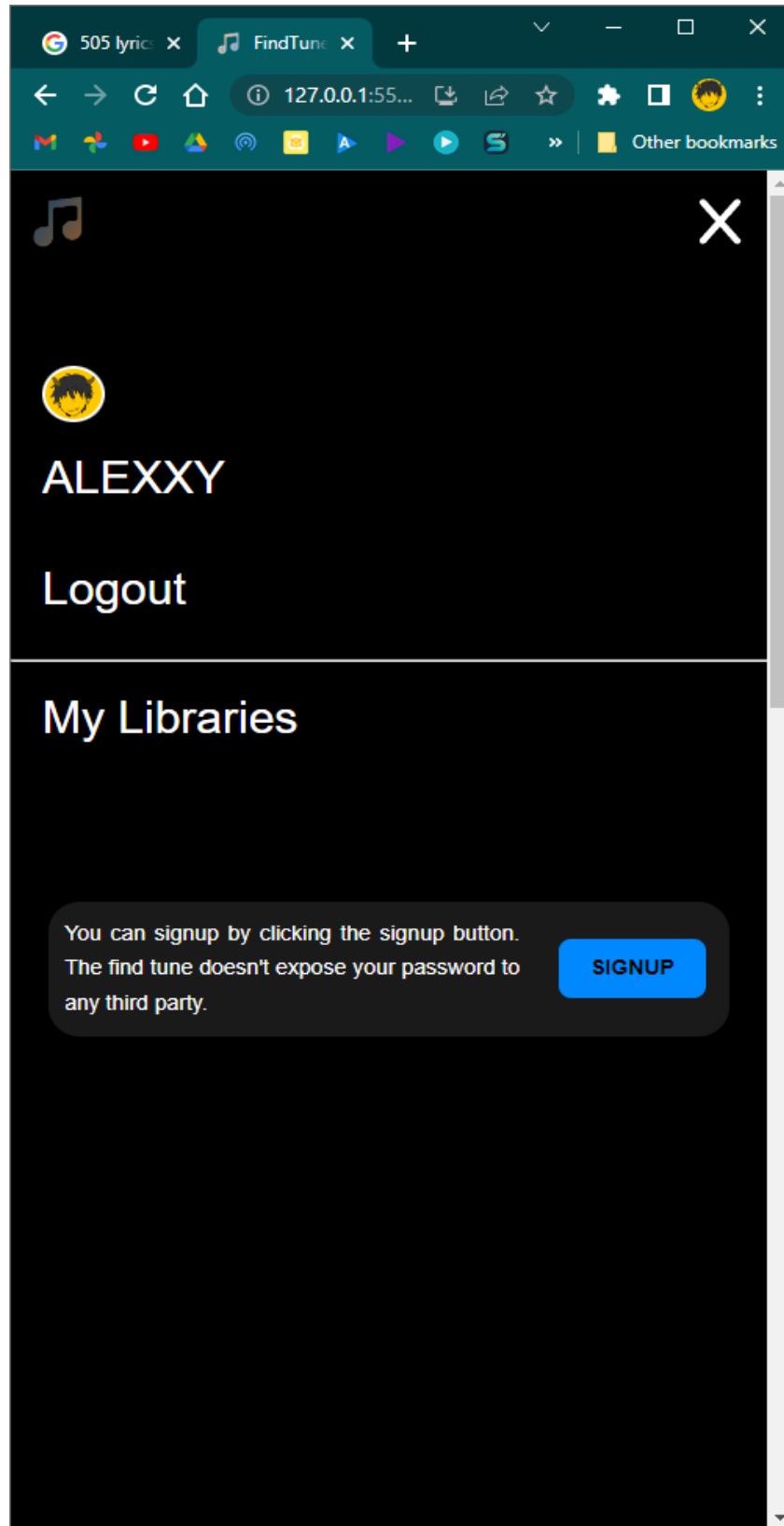


Figure 12. Responsive Hamburger menu

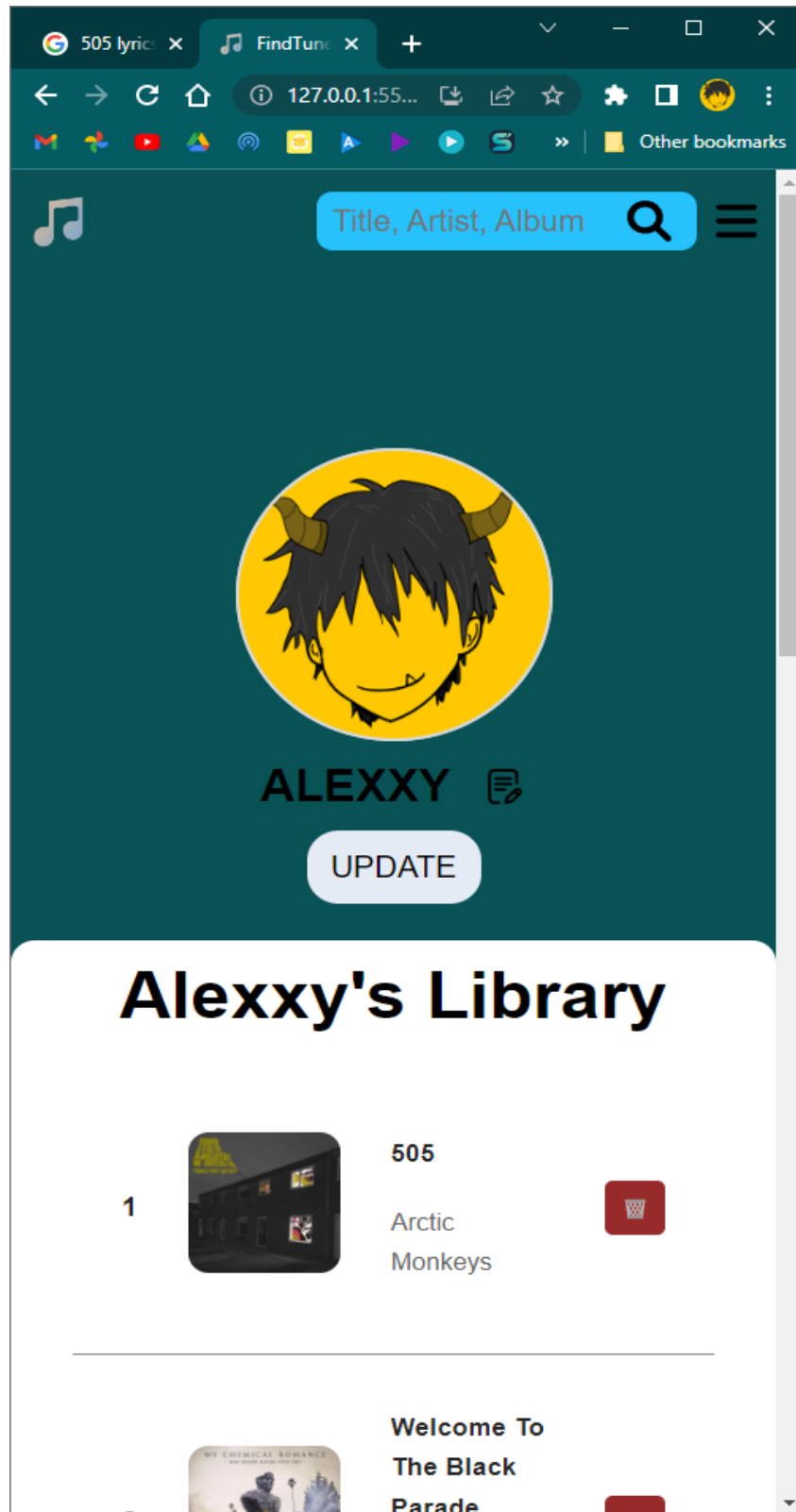


Figure 13. Responsive User Library

505 lyric x FindTune x +

127.0.0.1:55... Other bookmarks

X

Login Signup

Fullname *

Eg: Will Smith

Email *

Eg: apple@gmail.com

Username *

Eg: WILLS

Password *

Confirm Password *

SIGNUP

Figure 14. Responsive Login Signup Page

Database

Introduction

Database is the organized collection of data of any form. A database is very important in managing and storing the data. It is a structured that the data can be easily retrieved and stored. Database can be local or be hosted online in a cloud server either way a database makes it easy to store the data. The database can be visualized as the tables and inside the table there are rows and columns. Where columns define what types of data can be stored in. Columns can only hold single type of data. Whereas rows in the database are used to populate the columns. (["What is a Database Management System \(DBMS\)? - Definition from Techopedia", 2022](#))

There are 2 types of databases:

Relational

Database:

In this database there are tables which are normalized and are related to each other. The tables are connected to each other with the keys with both the tables contains this helps the table to find the correct and related data to each other. In this type of database each specific type of data domain is stored separately.

Non-Relational

Database:

In this database there are just table which are not normalized and contains repeating values. Normalization reduces the reputation of the data and spates the table in different and connects it with logic. But in non-relational database it doesn't occurs. The data in this database usually stored in key value pairs like JASO.

Find Tune uses relational database to store all its data. There are multiple tables which have their specific function to store data. There are five tables: User, Songs, Lyrics, Votes and User Library present in the database of Find Tune. First the User table as per name stores the information of the user. The table is created with the all the values with constraints like not nullable and unique. The User table's username and email are set to unique so the user can't create account with same email which is a security flaw. The User table also have relationship with User Library table. The user library is set up with a constraint if the user is deleted the data related to the user is also deleted on the User Library table. The Song table store the song details and every song is assigned with a unique tag so that the songs are not repeatedly store in the database. The link to the cover images, YouTube URL and thumbnail is also stored in database. The Song table is in relationship with the Lyrics table in such a way that if a song is deleted from the database the lyrics associated with the song are also deleted. The Lyrics table stores all the lyrics, the song lyrics is associated with, the user who posted the lyrics and the how much vote the song received. The song_id and user_id all is the foreign key in the table which connects the Lyrics table to song and User table respectively. The vote in the Lyrics table is the relation which is setup in a way when the lyric is deleted the votes associated with the lyrics are also deleted. The Vote table contains all the votes of the lyrics of a song and the info of the user who voted. Finally, the User Library table contains the info

of the song and the info of the user associated with it. The visual representation of the table is in the image below.

Advantages and disadvantages of MySQL

The name of the database used in the project is MySQL. It is an open-source software that creates and manages the database. It is a relational database which stores data in tabular format and links the tables with foreign keys.

[\("MySQL :: MySQL Documentation", 2022\)](#)

Advantages:

- It is a relational database management system, so it stores and shows data in tabular format which are organized in rows and columns.
- It is open-sourced and free to all users.
- It is compatible with almost all operating systems which makes it quite demanding and popular.
- It facilitates user to create and run the server from the same computer or different computers.
- It has a fast storing and data retrieving algorithm.
- It has a very simple and easy to use interface and a user with basic knowledge of MySQL can easily write queries.

Disadvantages:

- It is quite inefficient for very large databases.
- It does not have good debugging tools compared to other database management software.
- It doesn't support for the check constraints. [\("Advantages and Disadvantages of using MySQL", 2022\)](#)

Screenshots

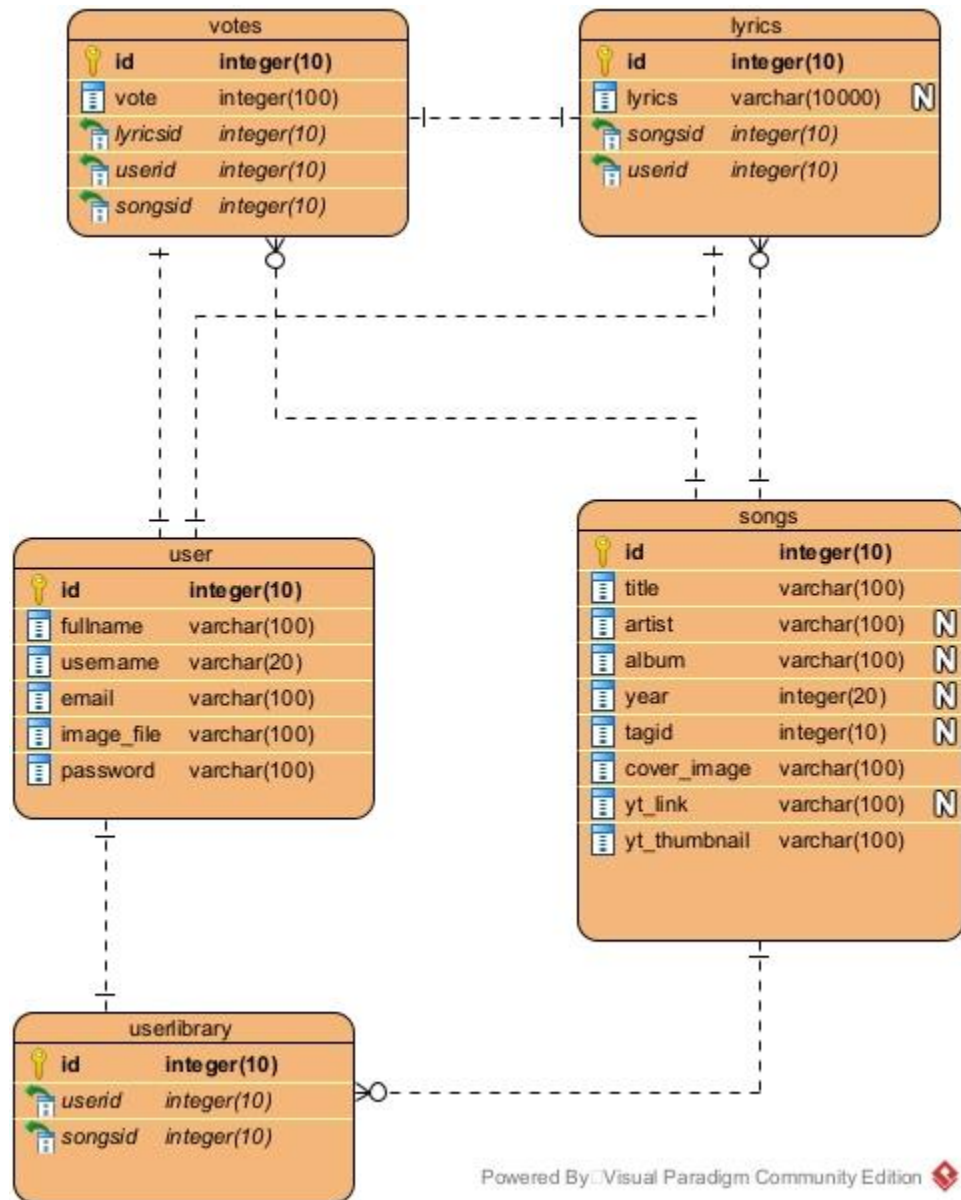


Figure 15. Database ERD

Backend

Introduction

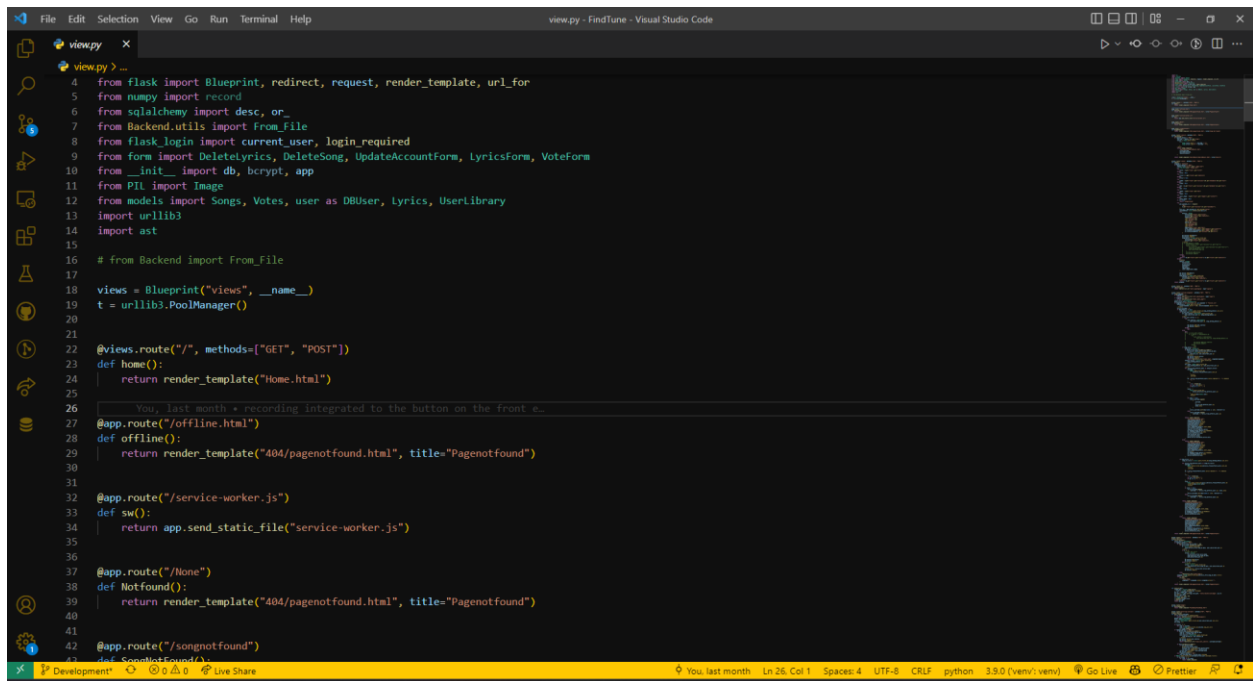
In lamente's term backend is considered as something a user doesn't directly access and is hidden for the front. Backend is the part of program that generally handles the user input and processed and generates the required output. It is also called server side. Backend helps to retrieve the data and process the data which the user can't see and sends out the desired output to the frontend.

The backend is achieved by using flask. Flask is the framework of the python programming language which is used for the web development process. The flask generally follows MVC (Model View Controller) pattern where the file structure is divided into Model, View and Controller. But the Find Tune is structured in MVT (Model View Temple) pattern. MVT is the mostly used software design pattern in the web development. Model View Temple separates the application into three components where each component does their sperate role that helps drive the application. The Model is the part which handles the database and its creation. Model stores the creation of tables in database and the constraints and variables of the database. The Views as per its name handles the interface of the website. In views all the routs and the logics of the page are stored. Form processing the data from the model and retrieving data from the page views does it all which makes it one of the important parts of the design pattern. At last, the Template, it contains all the templates like HTML, CSS and JS that is used to render the page on the website. (["What Is the Difference Between Front-End and Back-End Development?", 2022](#))

Technologies

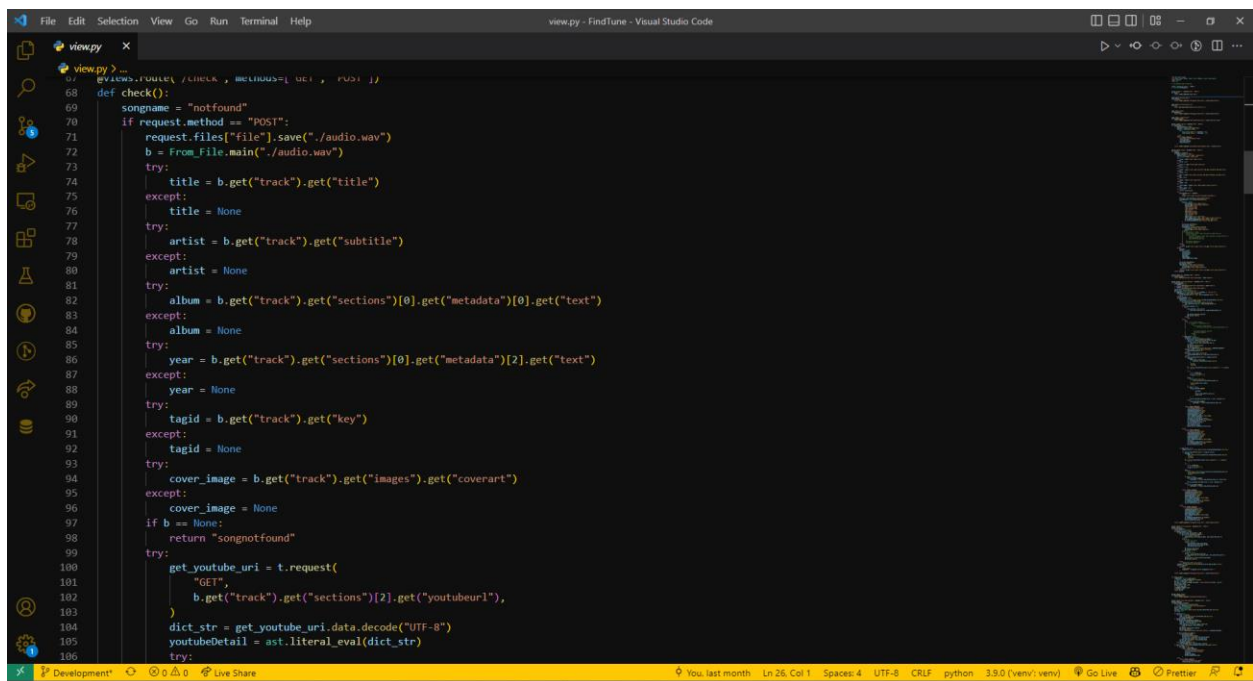
The technology used in the backend is flask a python library. Flask is a lightweight web development framework which helped this project to be what it is now. Flask handles all the rendering of the webpages with all the logic and components. Flask uses MVT architecture to divide the program into 3 specific parts that controls the website individually so that the code structure is clean and easy to understand. Flask is not a programming language rather it is a framework so having a basic knowledge of python is good enough to get started. Flask being a lightweight framework it lacks a bit of features, and the developer is required to invent the wheel and create the required feature themselves. (["Welcome to Flask — Flask Documentation \(2.1.x\)", 2022](#))

Screenshots



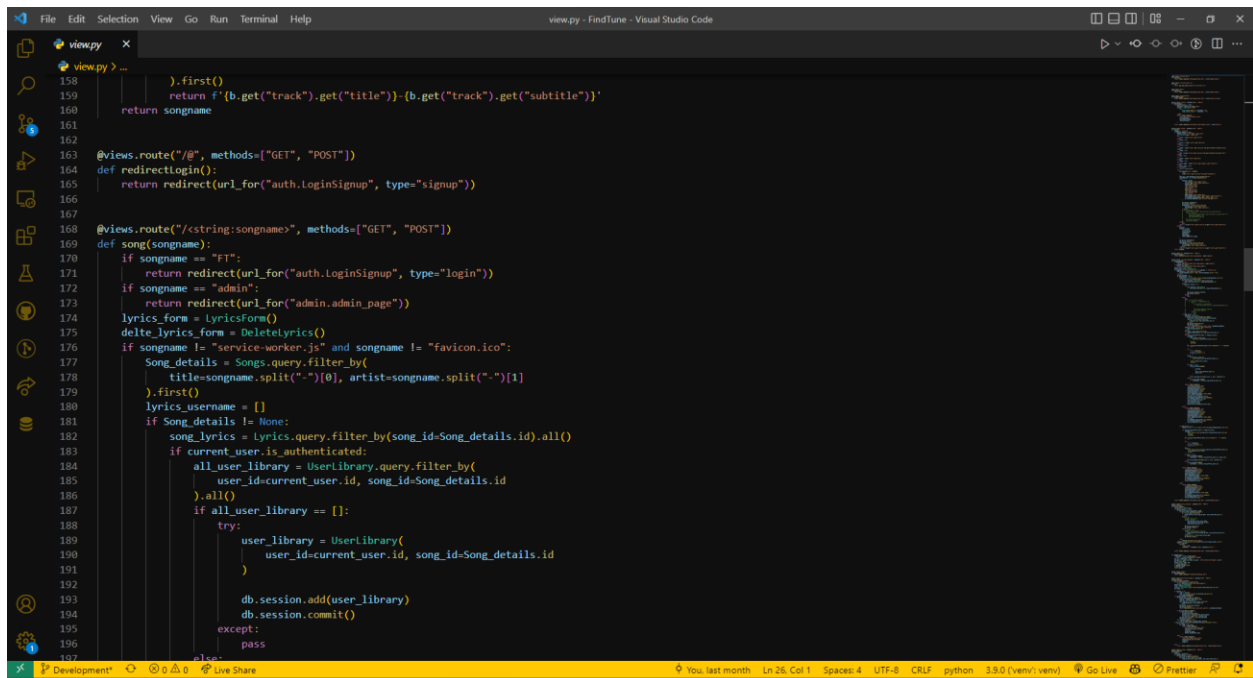
```
view.py x
view.py > ...
4 from flask import Blueprint, redirect, request, render_template, url_for
5 from numpy import record
6 from sqlalchemy import desc, or_
7 from Backend.utils import From_File
8 from flask_login import current_user, login_required
9 from form import DeleteLyrics, DeleteSong, UpdateAccountForm, LyricsForm, VoteForm
10 from __init__ import db, bcrypt, app
11 from PIL import Image
12 from models import Songs, Votes, user as DBUser, Lyrics, UserLibrary
13 import urllib3
14 import ast
15
16 # from Backend import From_File
17
18 views = Blueprint("views", __name__)
19 t = urllib3.PoolManager()
20
21
22 @views.route("/", methods=["GET", "POST"])
23 def home():
24     return render_template("Home.html")
25
26 # You, last month + recording integrated to the button on the front e.
27 @app.route("/offline.html")
28 def offline():
29     return render_template("404/pagenotfound.html", title="Pagenotfound")
30
31
32 @app.route("/service-worker.js")
33 def sw():
34     return app.send_static_file("service-worker.js")
35
36
37 @app.route("/None")
38 def NotFound():
39     return render_template("404/pagenotfound.html", title="Pagenotfound")
40
41
42 @app.route("/songnotfound")
43 def SongnotFound():
```

Figure 16. Backend Views.py



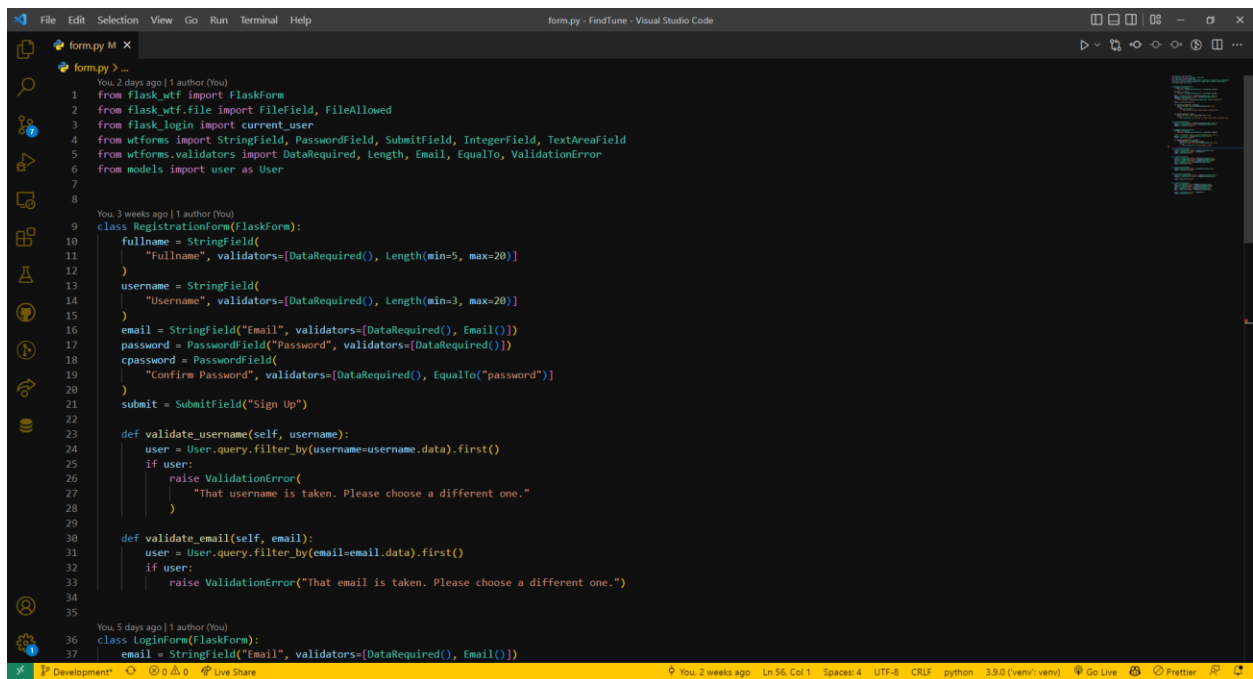
```
view.py x
view.py > ...
67 @views.route("/check", methods=["GET", "POST"])
68 def check():
69     songname = "notfound"
70     if request.method == "POST":
71         request.files["file"].save("./audio.wav")
72         b = From_File.main("./audio.wav")
73         try:
74             title = b.get("track").get("title")
75         except:
76             title = None
77         try:
78             artist = b.get("track").get("subtitle")
79         except:
80             artist = None
81         try:
82             album = b.get("track").get("sections")[0].get("metadata")[0].get("text")
83         except:
84             album = None
85         try:
86             year = b.get("track").get("sections")[0].get("metadata")[2].get("text")
87         except:
88             year = None
89         try:
90             tagid = b.get("track").get("key")
91         except:
92             tagid = None
93         try:
94             cover_image = b.get("track").get("images").get("coverart")
95         except:
96             cover_image = None
97         if b == None:
98             return "songnotfound"
99         try:
100             get_youtube_uri = t.request(
101                 "GET",
102                 b.get("track").get("sections")[2].get("youtubeurl"),
103             )
104             dict_str = get_youtube_uri.data.decode("UTF-8")
105             youtubeDetail = ast.literal_eval(dict_str)
106             try:
```

Figure 17.Backend Views.py 2



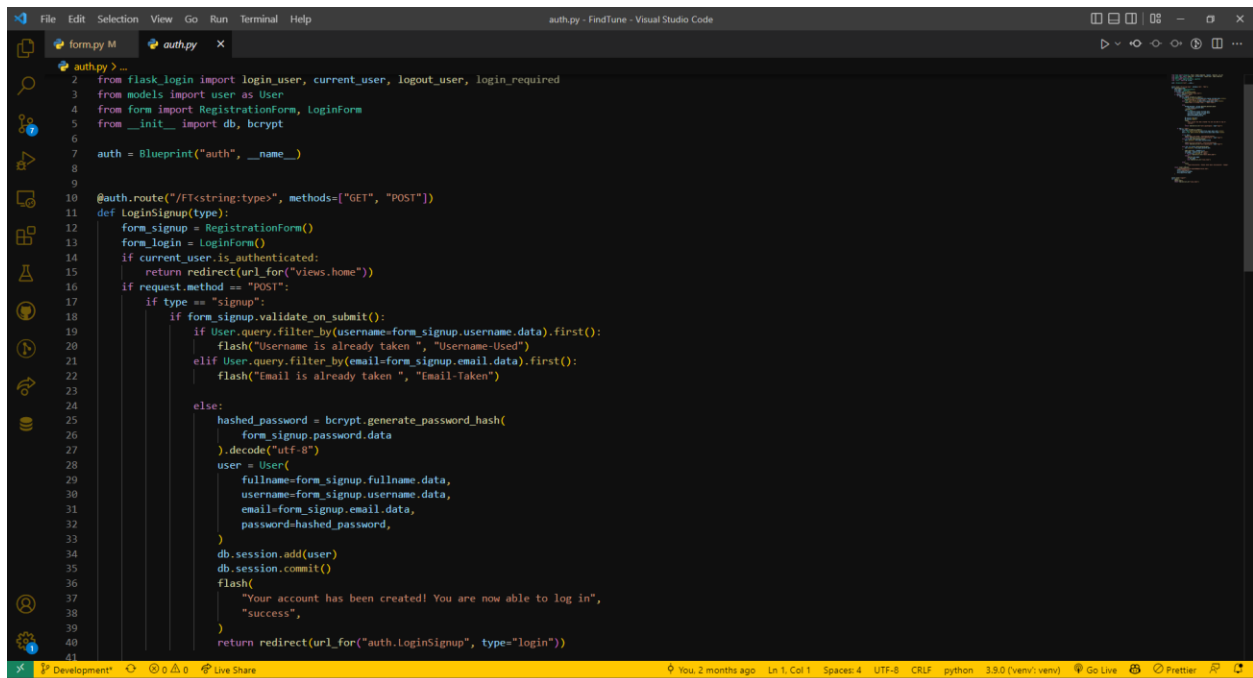
```
158         ).first()
159         return f'{b.get("track").get("title")}-{b.get("track").get("subtitle")}'
160     return songname
161
162
163 @views.route("/@", methods=["GET", "POST"])
164 def redirectlogin():
165     return redirect(url_for("auth.LoginSignup", type="signup"))
166
167
168 @views.route("/<string:songnames", methods=["GET", "POST"])
169 def song(songname):
170     if songname == "FT":
171         return redirect(url_for("auth.LoginSignup", type="login"))
172     if songname == "admin":
173         return redirect(url_for("admin.admin_page"))
174     lyrics_form = LyricsForm()
175     delete_lyrics_form = DeleteLyrics()
176     if songname != "service-worker.js" and songname != "favicon.ico":
177         Song_details = Songs.query.filter_by(
178             title=songname.split("-")[0], artist=songname.split("-")[1]
179         ).first()
180         lyrics_username = []
181         if Song_details != None:
182             song_lyrics = Lyrics.query.filter_by(song_id=Song_details.id).all()
183             if current_user.is_authenticated:
184                 all_user_library = UserLibrary.query.filter_by(
185                     user_id=current_user.id, song_id=Song_details.id
186                 ).all()
187                 if all_user_library == []:
188                     try:
189                         user_library = UserLibrary(
190                             user_id=current_user.id, song_id=Song_details.id
191                         )
192                     except:
193                         pass
194                     db.session.add(user_library)
195                     db.session.commit()
196                 else:
197                     pass
```

Figure 18.Backend Views.py 3



```
1 from flask_wtf import FlaskForm
2 from flask_wtf.file import FileField, FileAllowed
3 from flask_login import current_user
4 from wtforms import StringField, PasswordField, SubmitField, IntegerField, TextareaField
5 from wtforms.validators import DataRequired, Length, Email, EqualTo, ValidationError
6 from models import user as User
7
8
9 You 2 days ago | 1 author (You)
10 class RegistrationForm(FlaskForm):
11     fullname = StringField(
12         "Fullname", validators=[DataRequired(), Length(min=5, max=20)]
13     )
14     username = StringField(
15         "Username", validators=[DataRequired(), Length(min=3, max=20)]
16     )
17     email = StringField("Email", validators=[DataRequired(), Email()])
18     password = PasswordField("Password", validators=[DataRequired()])
19     cpassword = PasswordField(
20         "Confirm Password", validators=[DataRequired(), EqualTo("password")]
21     )
22     submit = SubmitField("Sign Up")
23
24 def validate_username(self, username):
25     user = User.query.filter_by(username=username.data).first()
26     if user:
27         raise ValidationError(
28             "That username is taken. Please choose a different one."
29         )
30
31 def validate_email(self, email):
32     user = User.query.filter_by(email=email.data).first()
33     if user:
34         raise ValidationError("That email is taken. Please choose a different one.")
35
36 You 5 days ago | 1 author (You)
37 class LoginForm(FlaskForm):
38     email = StringField("Email", validators=[DataRequired(), Email()])
```

Figure 19. Backend Form.py



```
1 from flask_login import login_user, current_user, logout_user, login_required
2 from models import user as User
3 from form import RegistrationForm, LoginForm
4 from __init__ import db, bcrypt
5
6 auth = Blueprint("auth", __name__)
7
8
9
10 @auth.route("/loginSignup", methods=["GET", "POST"])
11 def loginSignup():
12     form_signup = RegistrationForm()
13     form_login = LoginForm()
14     if current_user.is_authenticated:
15         return redirect(url_for("views.home"))
16     if request.method == "POST":
17         if type == "signup":
18             if form_signup.validate_on_submit():
19                 if User.query.filter_by(username=form_signup.username.data).first():
20                     flash("Username is already taken ", "Username-Used")
21                 elif User.query.filter_by(email=form_signup.email.data).first():
22                     flash("Email is already taken ", "Email-Taken")
23             else:
24                 hashed_password = bcrypt.generate_password_hash(
25                     form_signup.password.data
26                 ).decode("utf-8")
27                 user = User(
28                     fullname=form_signup.fullname.data,
29                     username=form_signup.username.data,
30                     email=form_signup.email.data,
31                     password=hashed_password,
32                 )
33                 db.session.add(user)
34                 db.session.commit()
35                 flash(
36                     "Your account has been created! You are now able to log in",
37                     "success",
38                 )
39                 return redirect(url_for("auth.LoginSignup", type="login"))
40
41
```

Figure 20. Backend auth.py

Project Issues

Issues during the project development

The web development is one of the most complex and long-winded process, and it being my first web development project it was quite the challenge. The project is created using HTML, CSS, JavaScript, and Bootstrap. The Frontend development was a big issue due to constant conflicts of the designs and non-responsive elements. Frontend development being one of the crucial and most interactive part for the user I wanted a properly laid out webpage which doesn't hinders the user experience. As I heavily used CSS and very few Bootstrap, I had to create the basic elements and classes from scratch, and it was time consuming. The great issue about the frontend development was how long it took, as frontend has lot of elements and pages, I had to take my time to create animations, responsiveness, and transitions for each of them.

Not only frontend backend also had lot of issue of its own. As this was the first time, I have used any web development framework so learning and development simultaneously was quite hard and frustrating. Flask is one of the web frameworks which is light weight and due to its lightweight feature, it doesn't have any pre created classes or anything and due to that every function is needed to be created from the scratch. Backend is which that processed data and of the output isn't achieved the page goes into error, so I had to tackle it very thoroughly, so the page doesn't run into any problem. The main problem

was jinja which doesn't have any debugging tools, so it was hard to find errors and solve them.

Limitations of your project

The whole project was inspired by one simple problem of not having shazam on the web where users can quickly record and search the song playing in the surrounding and find the lyrics and the song information. The projects achieve almost all the features that are mentioned and intended in the proposal of the project. But there are few features which are not fully implemented or achieved. One of the most recognizable features of shazam of listening to audio and simultaneously making fingerprints and finding the song in the database is something that I couldn't achieve. The parallel processing is something I was not familiarly with and couldn't achieve the set feature. Another limitation of the website it that there is no google signup or login which was expected on the website. The website also can't show the lyric that is synced with time. The user can download the lyrics but only in txt formant and not .lrc format.

Future works

As per the limitations mention above there are a lot of things that can be done in the future. The first thing that I wan to achieve is Google login and sign up to the user. Everyone with a smartphone got a Gmail account and letting user to login with it is quite common and easy for the user. Another simple yet quality of life feature that I want to add is letting user to download the lyric of the song in .lrc format with proper timestamps. One more feature which can be added is letting user to download their library in CSV format and letting them import the library. Displaying the source of the song where user can directly hop to and can listen to the music and add it to their playlist, I want to add a Spotify link so user can directly get redirected to it. One of the main features I want to add is the song reorganization while the song is playing instead aft the song is recorded and stored in the file server.

Diagrams

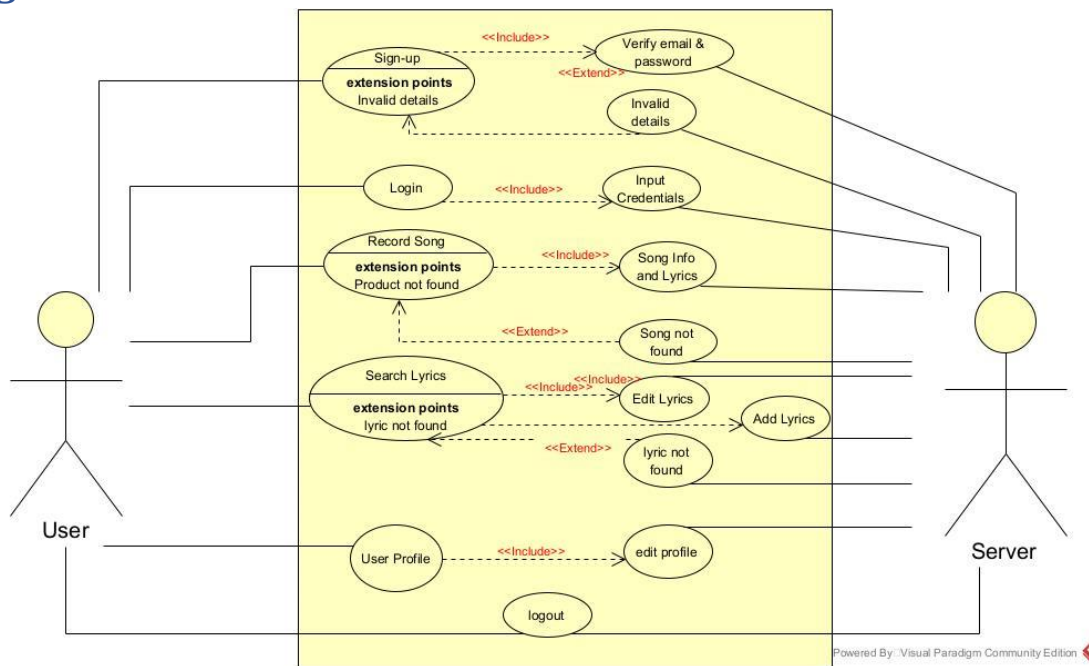


Figure 21. Use Case Diagram

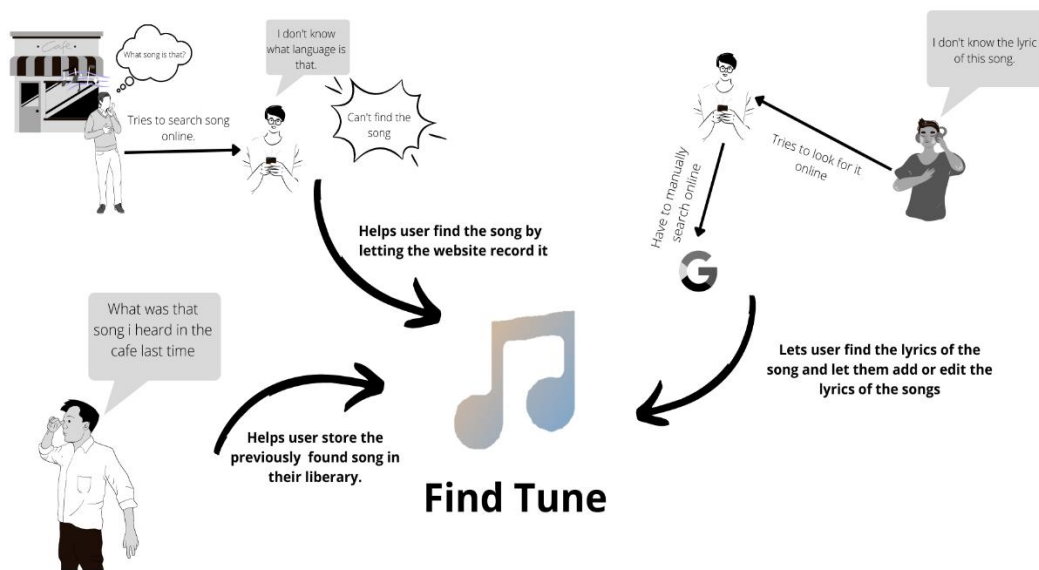


Figure 22. Rich Picture

Conclusion

Finally finishing up the whole documentation I would say that the project was a challenge for me, and it help me grow as a software developer. The project used HTML, CSS, Bootstrap and JavaScript as the frontend technologies which was quite challenge and made me learn new things about the language and the process of web development. The whole project is supported by the shazam song library where the song fingerprints are stored. The Local database contains the song information, lyrics, and the votes. Overall project does what it was set out to do and the learning experience was very precious for me for my future works. So, I would say that I have learn a lot and will use this information in my future works too.

References

Welcome to Flask — Flask Documentation (2.1.x). Flask.palletsprojects.com. (2022). Retrieved 24 July 2022, from <https://flask.palletsprojects.com/en/2.1.x/>.

What Is the Difference Between Front-End and Back-End Development?. Conceptatech.com. (2022). Retrieved 24 July 2022, from <https://www.conceptatech.com/blog/difference-front-end-back-end-development>.

What is a Database Management System (DBMS)? - Definition from Techopedia. Techopedia.com. (2022). Retrieved 24 July 2022, from <https://www.techopedia.com/definition/24361/database-management-systems-dbms>.

MySQL :: MySQL Documentation. Dev.mysql.com. (2022). Retrieved 24 July 2022, from <https://dev.mysql.com/doc/>.

Advantages and Disadvantages of using MySQL. Medium. (2022). Retrieved 24 July 2022, from <https://diliru.medium.com/advantages-and-disadvantages-of-using-mysql-36f6ffce3fa3>.

Front End Developer – What is Front End Development, Explained in Plain English. freeCodeCamp.org. (2022). Retrieved 24 July 2022, from <https://www.freecodecamp.org/news/front-end-developer-what-is-front-end-development-explained-in-plain-english/>.

Responsive Web Design - What It Is And How To Use It — Smashing Magazine. Smashing Magazine. (2022). Retrieved 24 July 2022, from <https://www.smashingmagazine.com/2011/01/guidelines-for-responsive-web-design/>.