CAB432 – Assignment



MUSIC APP

Name : Yonathan Cahyadi

Student Number : 10149953

Table of Contents

[Table of Contents 2](#_Toc51114213)

[Introduction 4](#_Toc51114214)

[Mashup Purpose and Description 4](#_Toc51114215)

[Service Used 4](#_Toc51114216)

[Spotify API 5](#_Toc51114217)

[Lyric.ovh API 5](#_Toc51114218)

[Google Cloud Text Translation API 5](#_Toc51114219)

[Mashup Use Case and Service 5](#_Toc51114220)

[Get Recommended Music of The Day 5](#_Toc51114221)

[Get Music Detailed Information 5](#_Toc51114222)

[Get Music Lyric 6](#_Toc51114223)

[Get Music Lyric Translation 6](#_Toc51114224)

[Technical Breakdown 7](#_Toc51114225)

[Architecture and Data Flow 7](#_Toc51114226)

[Deployment and the use of Docker 7](#_Toc51114227)

[Test Plan 8](#_Toc51114228)

[Difficulities / Unresolved & Persistent Error 9](#_Toc51114229)

[Extension 9](#_Toc51114230)

[User Guide 10](#_Toc51114231)

[Statement on Assignment Demo 18](#_Toc51114232)

[Appendices 18](#_Toc51114233)

[Appendix A 18](#_Toc51114234)

[Dockerfile for Client 18](#_Toc51114235)

[Dockerfile for Server 19](#_Toc51114236)

[Docker-compose 20](#_Toc51114237)

[Appendix B 21](#_Toc51114238)

[01 21](#_Toc51114239)

[02 21](#_Toc51114240)

[03 22](#_Toc51114241)

[04 22](#_Toc51114242)

[05 23](#_Toc51114243)

[06 23](#_Toc51114244)

[07 24](#_Toc51114245)

[08 24](#_Toc51114246)

[09 25](#_Toc51114247)

[10 26](#_Toc51114248)

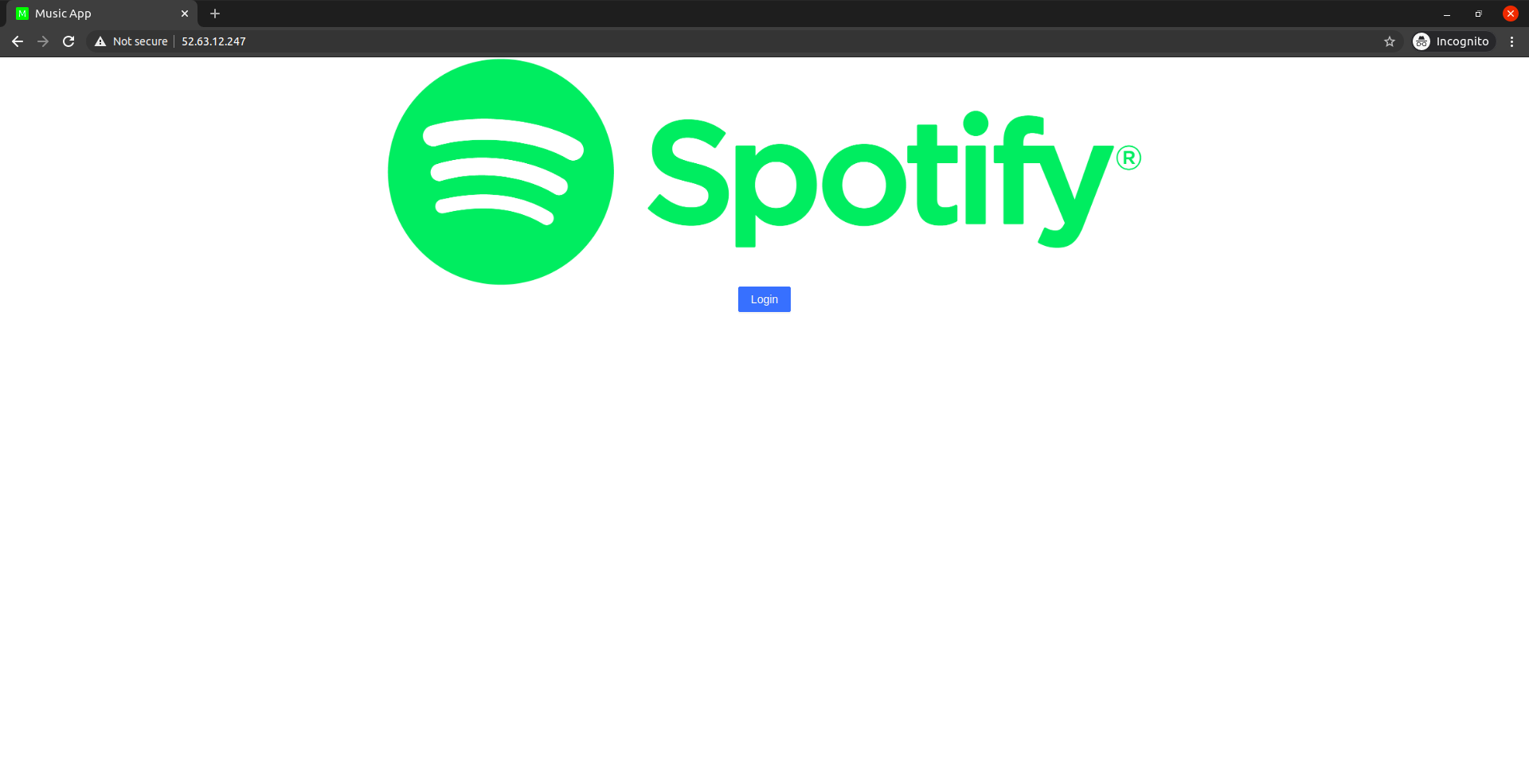
[11 27](#_Toc51114249)

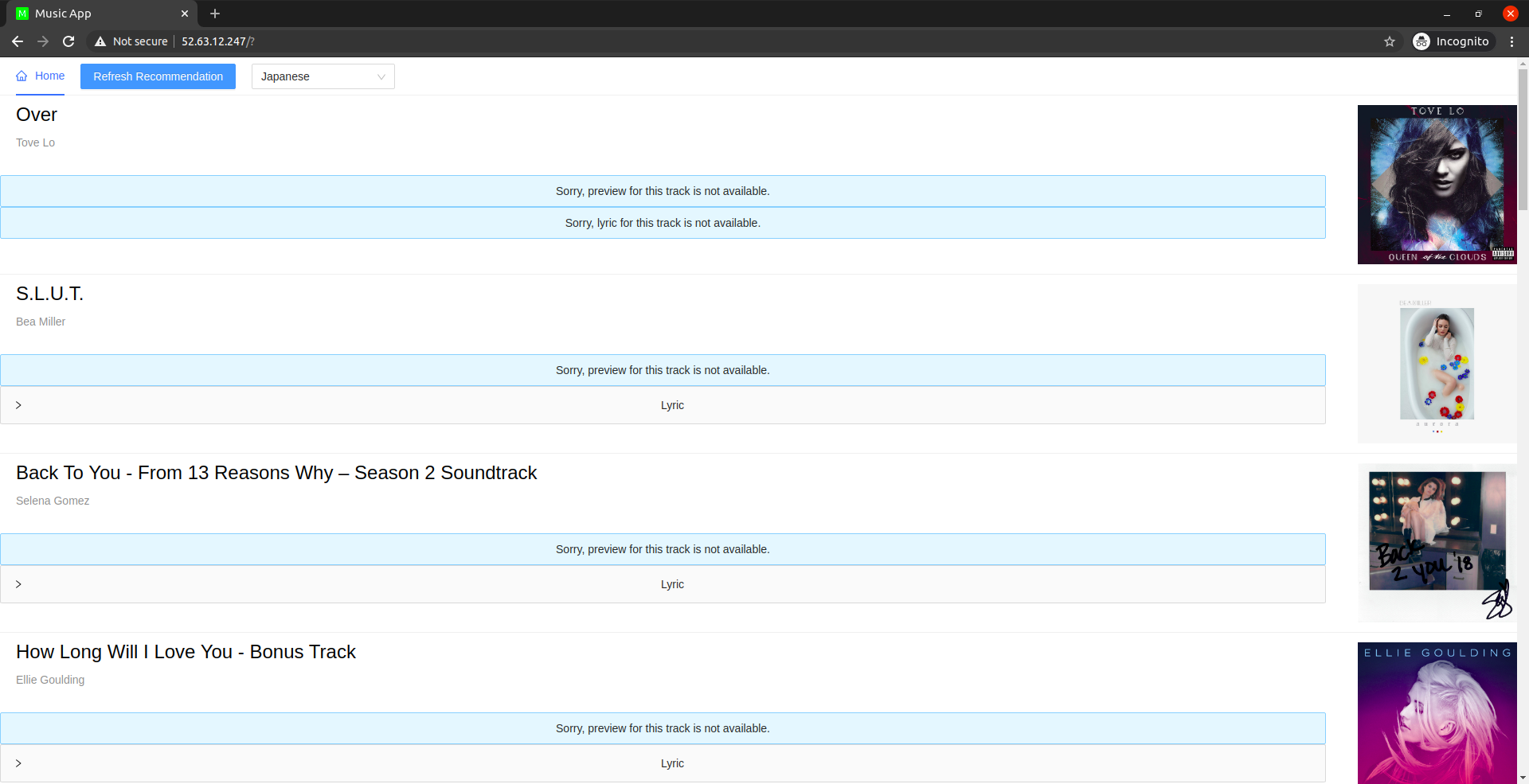
# Introduction

## Mashup Purpose and Description

The purpose of this app is to give music enthusias recommended music of the day. This app also give the detailed information regarding the recommended music, the detail the app gives including: music title, artist name, preview music and the lyric of the music in 2 different languages.

**The Login Page:**

**The Home Page:**



## Service Used

This is the list of service and API that has been used in this app.

### Spotify API

Get user Authorization and recommended music.

* Authorization Endpoint: https://accounts.spotify.com/authorize
* Recommended Music Endpoint: <https://api.spotify.com/v1/recommendations>
* Authorization Docs: [https://developer.spotify.com/documentation/general/guides/authorization-guide/#authorization-flows](https://developer.spotify.com/documentation/general/guides/authorization-guide/" \l "authorization-flows)
* Recommended Music Docs:

<https://developer.spotify.com/documentation/web-api/reference/browse/get-recommendations/>

### Lyric.ovh API

Get the lyric of the recommended music.

* Lyric Endpoint: <https://api.lyrics.ovh/v1>
* Docs: <https://lyricsovh.docs.apiary.io/>

### Google Cloud Text Translation API

Translate the music lyric into user c+hoise of language.

* Google Translate Endpoint: <https://translation.googleapis.com/language/translate/v2>
* Docs: https://codelabs.developers.google.com/codelabs/cloud-translation-intro/index.html#0

# Mashup Use Case and Service

## Get Recommended Music of The Day

|  |  |
| --- | --- |
| As an | Music Enthusiast |
| I want | To get some good music recommendation |
| So that | I can have a good music to listen |

## Get Music Detailed Information

|  |  |
| --- | --- |
| As an | Music Enthusiast |
| I want | To get Music detailed information |
| So that | I can know more of the music |

## Get Music Lyric

|  |  |
| --- | --- |
| As an | Music Enthusiast |
| I want | To get Music lyric |
| So that | I can sing along with the music |

## Get Music Lyric Translation

|  |  |
| --- | --- |
| As an | Music Enthusiast |
| I want | To get Music lyric translation of my languages |
| So that | I know the meaning of the song |

# Technical Breakdown

The framework that I used for the server is Express.js and for the client, I use React. As for the API in this project I use 3 of them, which is Spotify, Lyric.ovh and Google Cloud Text Translation API. So for this project, I use Spotify to get the recommended song, after I receive the result from the Spotify I use those data to get the song lyric using the Lyric.ovh API and all of this data processing are been done in the server. After my server receives the lyric, my server will construct the data to be sent to the client. Those data include the song name, artists, song preview URL, and lyric. This data is sent to the client in JSON format. For the Client, it will handle the Spotify authorization so we can get access token required to get the recommended music. The client also handles the translation for the lyric using Google Cloud Text Translation API.

## Architecture and Data Flow

This diagram show the relation between my server, my client and the used API for this project. This diagram also show how the data flow.

A close up of a map

Description automatically generated

## Deployment and the use of Docker

I use Docker for making the images for both my client and server. My Dockerfile for server contains the instruction to make an image using node:erbium as the base images and copy the server code into the working directory of the container, also do ‘npm install’ to install all the needed dependency inside the container and expose the port 443. While my Dockerfile for the Client contains the instruction to make an image using node:erbium as the base images and copy the client code into the working directory of the container, also do ‘npm install’ to install all the needed dependency inside the container, it also specifies the environment variable inside the container for which is needed for the client, the Dockerfile also expose port 3000. I also use docker-compose for building the images and pushing the created images to the Docker hub, unfortunately, I cannot use docker-compose to pull the images from the repository. My docker-compose file contains information needed for the docker to build the images and push the images to the docker hub. Such information includes the Dockerfile directory for both client and server, the tag of the images and port that needed to be exposed for both client and server. For more information there would be an image of docker-compose file and Dokerfile of both the client and server in the appendix A.

For the Deployment of my app I started by making a virtual machine instance in AWS. After I make my vitual machine I can connect to it using session manager. Then inside the virtual machine I can started to pull my images from the Docker repository. After I finished pulling my server and client images I can see my pulled images by using docker images and to start running our pulled images we can do it by using these command:

Server:

docker run -d -p443:443 <Server Image ID>

Client :

docker run -d -i -p80:3000 \

-e REACT\_APP\_URL=<this machine public IP with port at 80> \

-e REACT\_APP\_SERVER\_URL=<this machine public IP with port 443> <Client Image ID>

After we run our imager, we can the running container by using docker ps . Before I can see my website I need to go to Spotify API Dashboard to Whitelist my AWS virtual machine public IP. After we White list the virtual machine Public IP. We can go to the website by go to “http://<virtual machine public IP>:80”

## Test Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Expected Outcome** | **Result** | **Screenshot/s**  **(Appendix B)** |
| Login button clicked | Redirect to Spotify Authorization/Login Page | PASS | 01 |
| Authorization Success | Go to the Home page of the app, user get a loading page, while the client waiting for the data form the server | PASS | 02 |
| User arrive at Home Page | User get some music recommendation with,  Song title, artist name, song preview, lyric in english and Bahasa Indonesia (default). User also see some option in the header of the page, such options are refresh the recommendation, and languages option for translated lyric by default it set to Bahasa Indonesia. | PASS | 03 |
| Click on Lyric Dropdown | Show both original lyric and translate lyric | PASS | 04 |
| Handle no song preview | The user get informed that the song did not have any song preview | PASS | 05 |
| Handle no song lyric | The user get informed that the song did not have any lyric. | PASS | 06 |
| Handle Lyric Translation | The user can see the translated lyric beside the original lyric | PASS | 07 |
| Click on song title | Open a spotify page regarding the music | PASS | 08 |
| Change the languages for the translated lyric | The user will see that lyric is being processes it will show the user a loading animation. After it finish processing it will change the old translated lyric with the new translated lyric (using user choice of language). | PASS | 09 |
| Authorization Failed | The user get redirected back to the Login Page | PASS | 10 |
| Error in handling user access token | The user get Error Page with button to go back to the Login page | PASS | 11 |

## Difficulities / Unresolved & Persistent Error

Some difficulties that I encouter when making this project is, dealing with Spotify Authorization Endpoint. At first I make user authorization form the server side, but after some understanding of how spotify API work I change the authorization process to the client side. Therefore when my client query my server I only need to pass the access token. When in before I need to pass around the access token back and forth between my client and server. Another difficulties that I encounter isu sing Google Cloud Text Translation API, their API documentation is confusing because of the variation of service they offer, but I manage to solve it. There is also some slight problem with my client side, because I use some library for the design of my app the library is pretty out dated. Therefore it give some warning when used, but it was not really a problem.

# Extension

I'm thinking of making an event timeline of the artist past and future event. I also considering implementing Google Cloud Text Analysis API, maybe using this API and the music lyric I can give the music a tag like “Sad Music”, “Happy Music”, etc.

# User Guide

Click the Login Button:

A screenshot of a cell phone

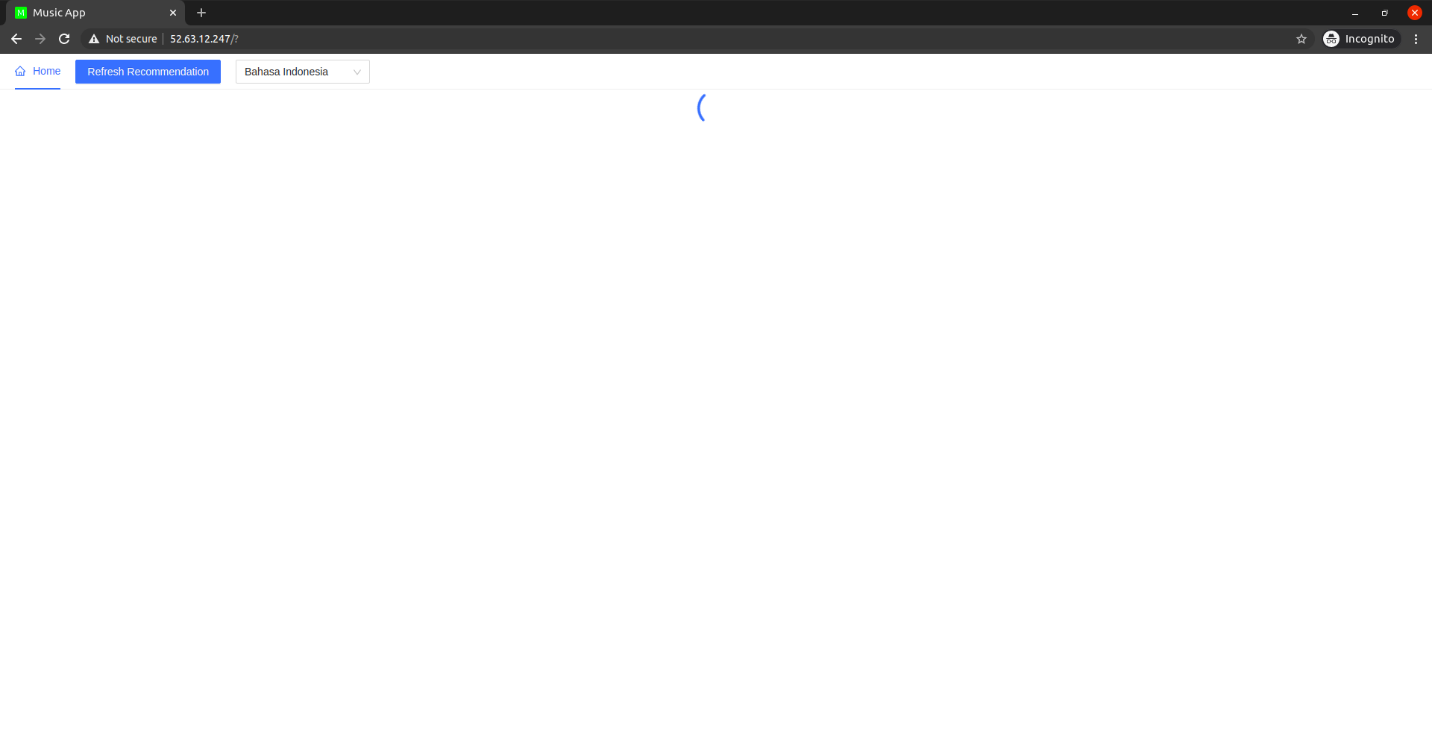
Description automatically generated

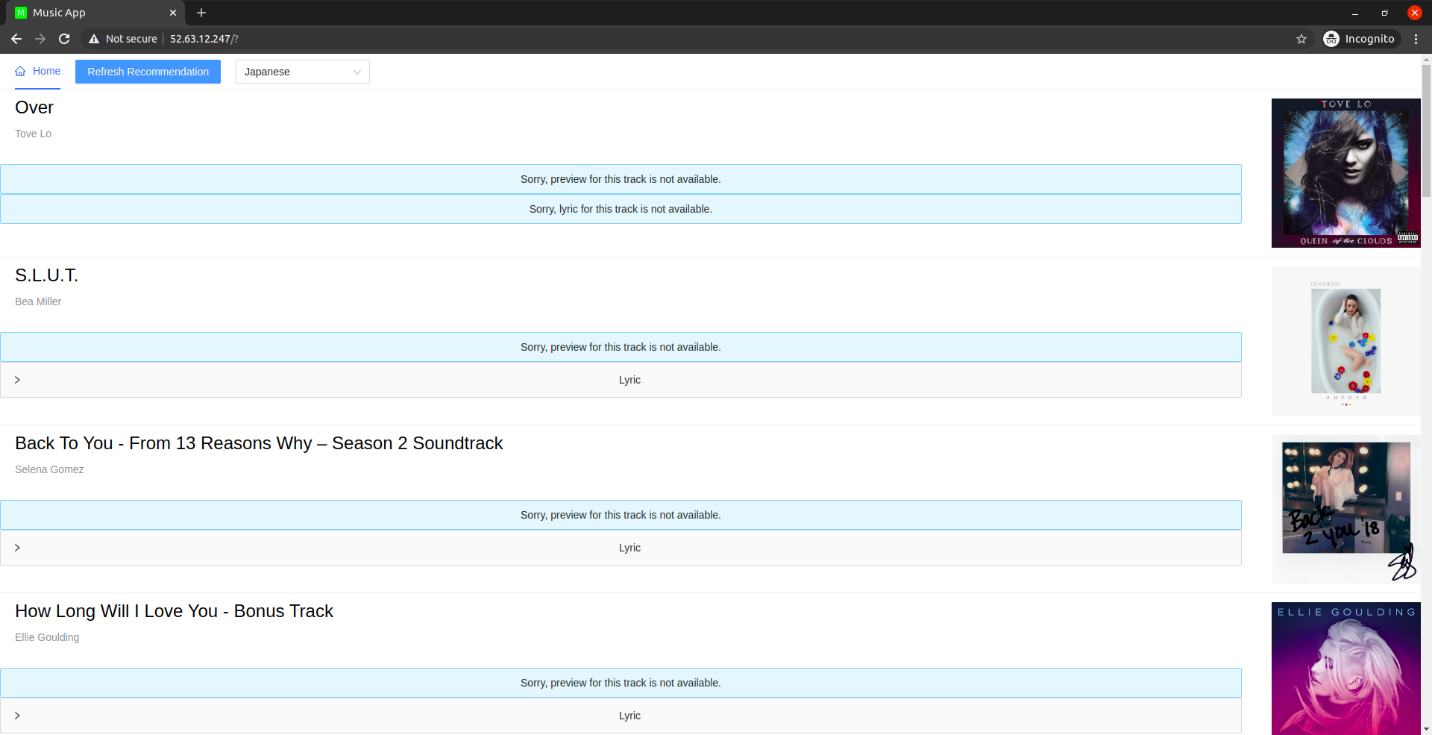
It will open a Spotify Login page, where you can log in using your spotify account

A screenshot of a cell phone

Description automatically generated

If the authorization success, it will redirect you to the Home page of the app, in there you can see some music recommended by Spotify.





You can click the Lyric and it will give you the original music lyric and the translated music lyric.

A screenshot of a computer screen

Description automatically generated

A screenshot of a social media post

Description automatically generated

You can also change the translated lyric using the button on the top of the page

A screenshot of a social media post

Description automatically generated

It will give you choice of languages

A screenshot of a social media post

Description automatically generated

After you choose your choice of language it will change the translated lyric into the languages of your choice. (for this example I choose Japanese).

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

If you want to get a new music recommendation you must use the Refresh Recommendation Button located at the top, next to the language selection dropdown box. If you just reload the page it will not change anything, because the page infomation is being stored in session storage.

A screenshot of a computer screen

Description automatically generated

You also can see song detail by clicking the song title. It will bring you to Spotify

A screenshot of a social media post

Description automatically generated

A screenshot of a computer screen

Description automatically generatedYou also can hear the song preview by clicking play on the media player.

A screenshot of a social media post

Description automatically generated

# Statement on Assignment Demo

For this assignment, I intended to make a video for the Demo.

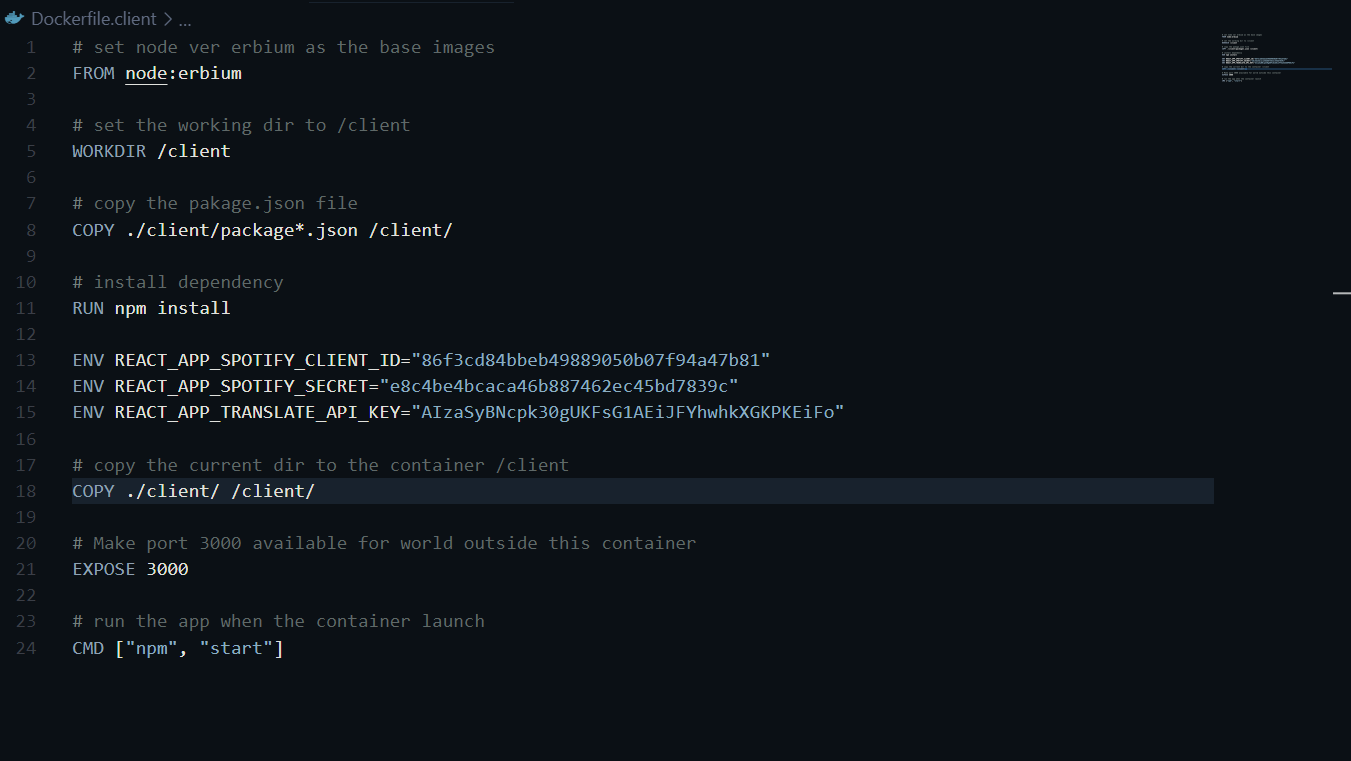
For more higher resolution: hhttps://www.youtube.com/watch?v=tsH22WObFhY

# Appendices

## Appendix A

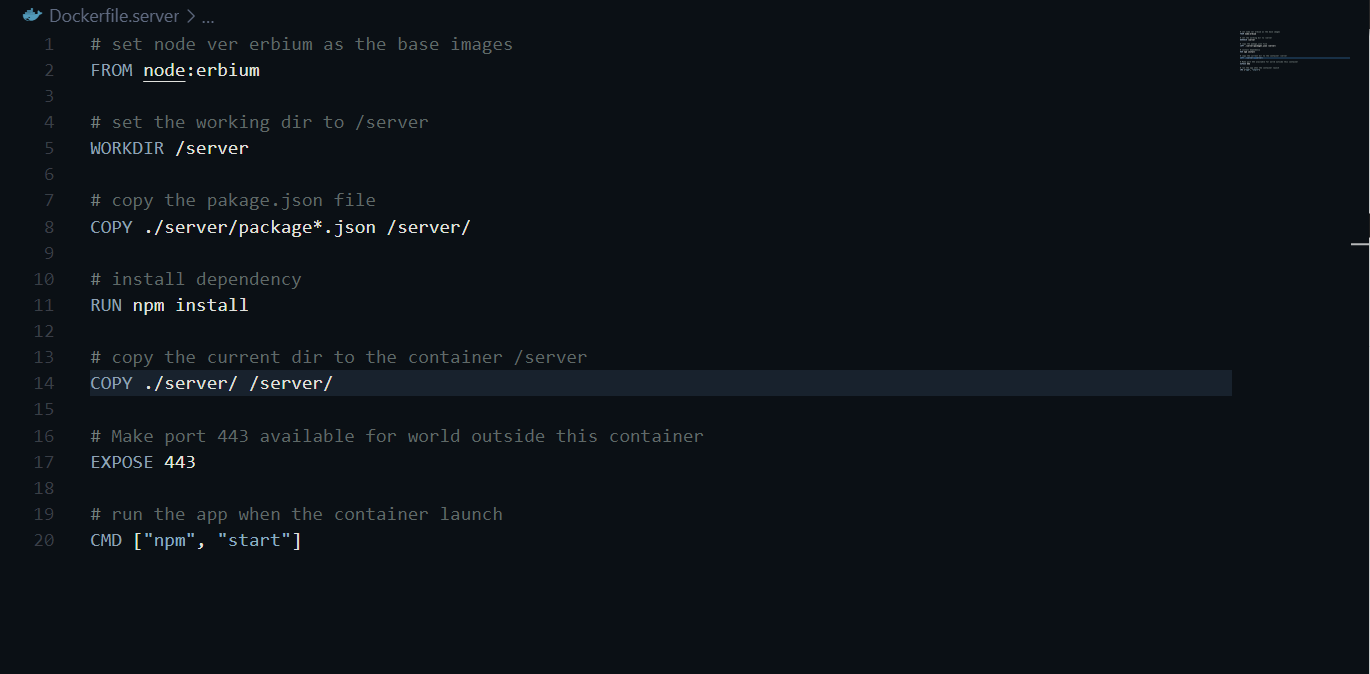
### Dockerfile for Client

Dockerfile.client



### Dockerfile for Server

Dockerfile.server



### Docker-compose

docker-compose.yaml



## Appendix B

### 01A screenshot of a cell phone Description automatically generated

### 02

A screenshot of a cell phone

Description automatically generated

### 03

A screenshot of a computer screen

Description automatically generated

### 04

A screenshot of a social media post

Description automatically generated

### 05

A screenshot of a computer screen

Description automatically generated

### 06

A screenshot of a computer screen

Description automatically generated

### 07

A screenshot of a social media post

Description automatically generated

### 08

A screenshot of a social media post

Description automatically generated

A screenshot of a computer screen

Description automatically generated

### 09

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

### 10

A screenshot of a cell phone

Description automatically generated

### 11

