Functional Specification

Project name: Sham Shazam
Student 1: Georgina Scanlon ID Number: 19392373
Student 2: Rhea Varkey ID Number: 19452962

Date submitted: 17/11/2022

Table of Contents

1. Introduction

1.1 Overview

The goal of our project is to create a location based music recognition service, similar to Shazam, but using open systems. This will take audio input, identify the song playing and record this information along with the location, using GPS, to a database.

We plan on making this as a progressive web application using react which will take audio input via the device and will then use an existing API service such as Aud-D to identify the song and record this information plus the location via GPS to a database.

We also plan on including the development of neural network technologies for reducing the noise on the recordings to improve the ability of the system to recognise music which is playing in noisy environments.

Our app will allow the user to find out information about the song and have a record of the songs that have been recorded. This can be useful for a wide variety of users that enjoy music.

Our project is also targeted towards music artists and people who play music in public areas, such as shop owners, as our project helps to figure out what music is playing in public spaces and stores it in a database so that it could aid businesses in paying royalties to the appropriate artist and record label in the appropriate time frame.

1.2 Business Context

Our project is targeted towards music artists and people who play music in public areas,such as shop owners, as our project helps to figure out what music is playing in public spaces and stores it in a database so that it could aid businesses in paying royalties to the appropriate artist and record label in the appropriate time frame.

1.3 Glossary

- **GPS:** "Global Positioning System." GPS is a satellite navigation system used to determine the ground position of an object.
- **API:** “Application Programming Interface”, which is a software intermediary that allows two applications to communicate to each other.
- **Royalties:** Fees to pay the artist/ record labels to play their songs

2. General Description

2.1 Product / System Functions

The main function of our application will allow the user to obtain information about a song playing in the background such as the song name, artist name, release date and album.

This information will be stored in a database along with the location using GPS so that the user can view a history of the songs that they have listened to.

2.2 User Characteristics and Objectives

We expect that users of our application will not require any specific knowledge to understand how to use our application. We aim to make it simple and intuitive to use, even for those not very familiar with technology. That being said, users will require some very basic familiarity with the device they are using, be it a mobile phone or computer, in order to understand how to navigate to the web application using the URL.

2.3 Operational Scenarios

The main objective that a user will have when using our application is to recognise a song playing. To do this, they would first log in to the app. Once logged in, if they are not already on the main page they will have to navigate there. The main page will display a button to start recording. Once the user presses this button, the application will start to send audio to the API to be identified. We aim to send the past 20 seconds of audio every 1 minute while the device is recording. Once the user presses the button to stop recording, the application will show them what songs were identified during the period with details such as artist and album.

Another scenario that may be encountered is if the user wants to view their history of songs that were identified. To do so, they once again will need to log in. Once logged in, they will simply be able to navigate to the history page which will display the previously identified songs.

2.4 Constraints

- The API we are using may not be able to recognise every song. Some songs, especially those that aren't that popular, may not be recognised.
- If the background is too noisy it would be difficult for the audio to pick up the actual music

- It would be hard to classify if the audio contains music and the end product may not be perfect at classifying the audio as music

3. Functional Requirements

1. Identify songs:

Description: This is the main requirement of the project as it would be able to name the song and what artist/ record label it belongs to, as they are the ones that will be receiving the royalties based on if there is music playing in the audio input.

The plan to implement this function is to read the audio input from the device which is split into 20 seconds and to identify the music through any background noise present. If there's no music playing or if it is not a new song, it identifies the next twenty seconds. If there is music playing it identifies it using the external API and then stores that information in our database.

Criticality: Highly critical. Our project depends on this functionality.

Technical issues: A possible issue with implementing this function is that we will be dependent on the APIs rate limits. We will have to ensure that we are able to make enough requests to the API for our program to be functional without hitting these limits.

Another possible issue is that in areas with background noise the functionality will depend on how we implement the noise reduction. If our implementation is not effective, we may not be able to identify songs in noisy environments.

Dependencies with other requirements: The user will need to be logged in so our registration and login system must be functional.

2. Registration and Login system:

Description: A user will need to be able to register and login in order to keep track of the songs they identified. Therefore, we will have to have a registration system to allow for this.

Criticality: Highly critical.

Technical issues: N/A

Dependencies with other requirements: N/A

3. View previously identified songs:

Description: This requirement will allow users to see information about the songs they have previously identified. The information gathered when a user identifies a song will be stored in the results database. We can then output that information into the user interface where it lists all the songs that had been playing while the program was running. The user can then see how many songs have been played and by what

artist and record label so that they can calculate how much royalties they have to pay and to which artists.

Criticality: Quite critical. As our project is directed towards people keeping track of royalties owed, having a list of previously identified songs would be quite important.

Technical issues: N/A

Dependencies with other requirements: Will be dependent on the identification function working. If the project is unable to identify the songs reliably then the history will also be unreliable.

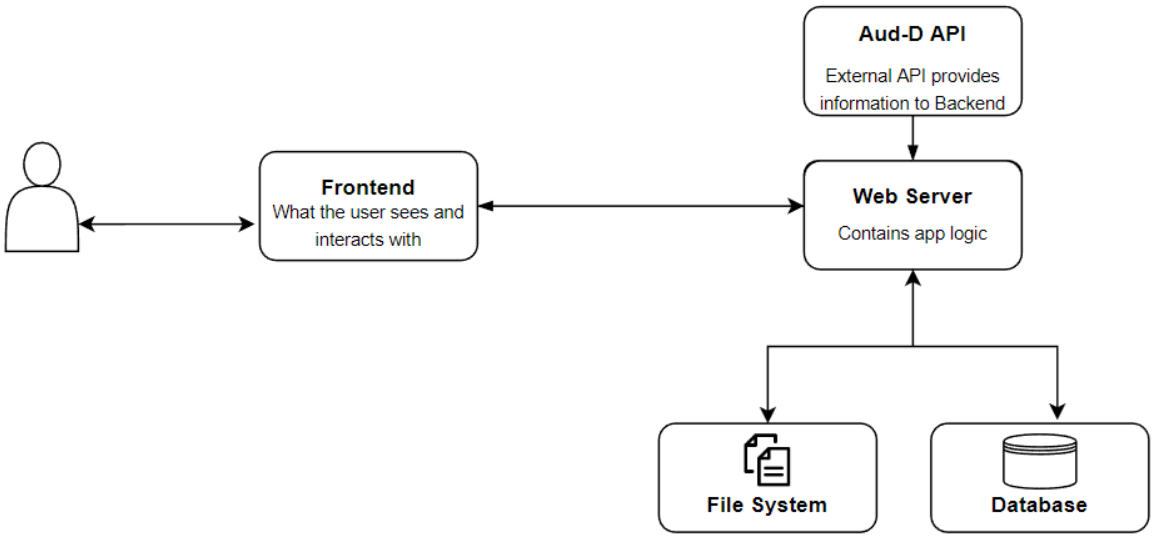
4. System Architecture

Our system will include a frontend User Interface, a backend web server and a database. It will interact with the external API Aud-D, in order to identify music.

When a user interacts with the system, they will use the User Interface through their device's browser. When they interact with the UI, a call will be sent to the backend web server. If the user wishes to identify a song, a call will need to be made to the external API which will send the song details back to the web server.

When a user identifies a song, the song information is stored along with some of the user's information such as GPS location to the database. This database also provides information to the web server about the user's previously listened to songs, so that they can view their history on the UI.

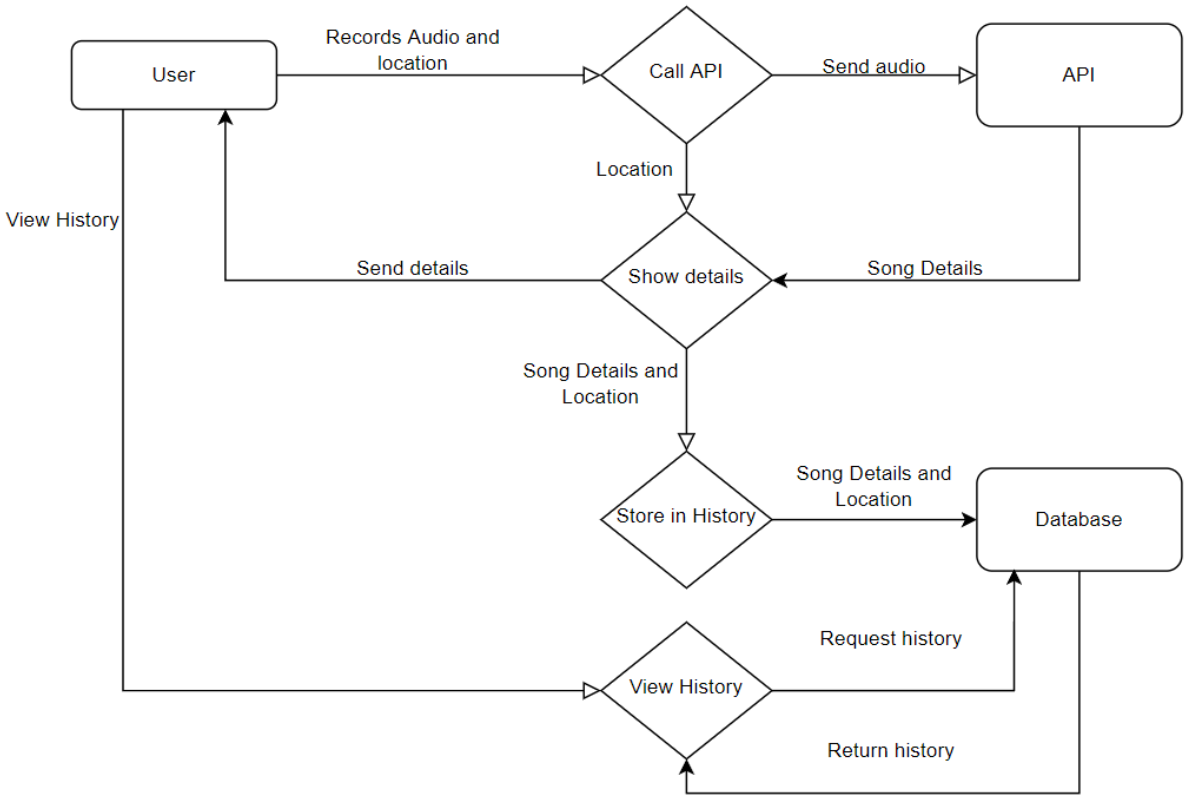
Finally, the server's file system is used to store any images or styling that is displayed on the frontend to the user.



5. High-Level Design

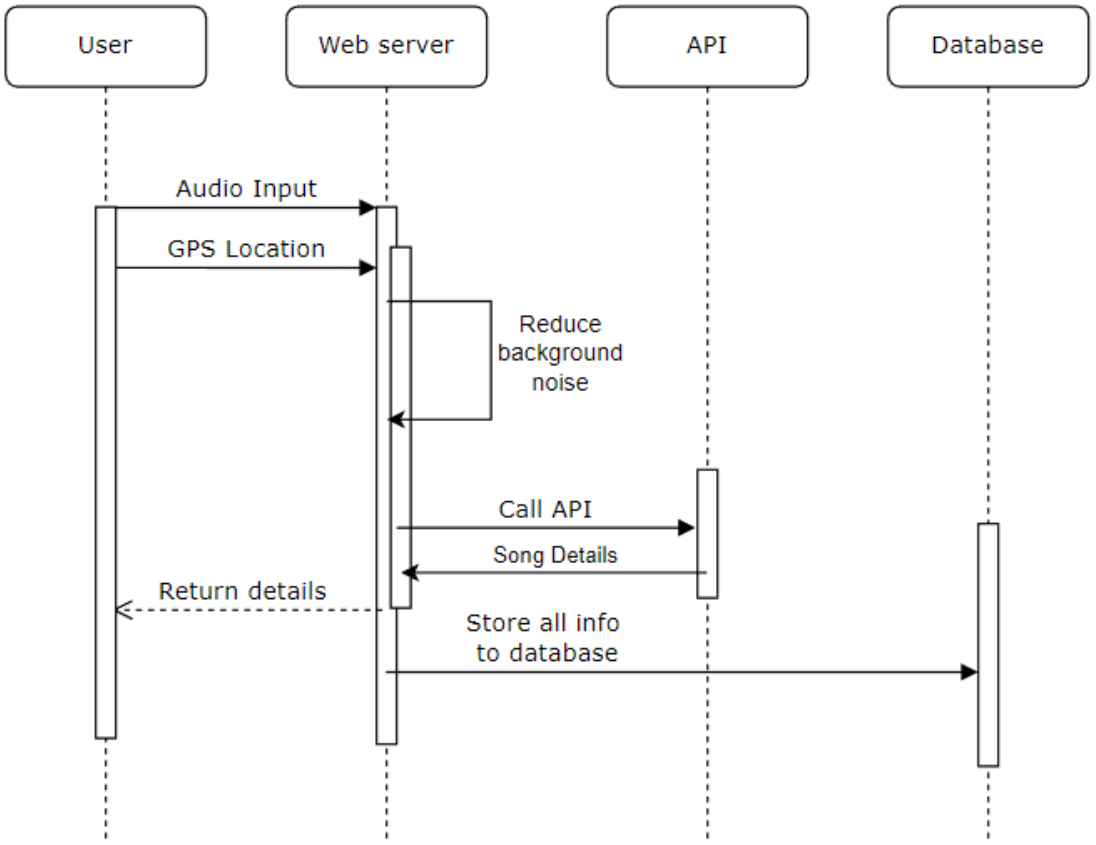
This section should set out the high-level design of the system. It should include one or more system models showing the relationship between system components and the systems and its environment. These might be object-models, DFD, etc.

Data flow diagram:



Sequence diagrams:

Identifying a song:



6. Preliminary Schedule

- Week 1: Set up the function to call to API and get song information from a given audio clip.
- Weeks 2-4: Aim to implement identification function that takes audio input from device
- Weeks 5-8: Aim to be able to get GPS location from device and store it along with song information to database.
- Weeks 6-12: UI/UX of the web interface
- Weeks 10-16: Implement polling to send calls to API at set intervals with audio clip to identify.
- Weeks 17-22: Reduce background noise on audio input.
- Weeks 23-25: Testing of the program
- Weeks 24-25: Documentation

[Link to Full GANTT Chart]