Audionyx

Developed by:
Abhi Chalise
Agam Singh
Connor Scott
Alyssa Webb
Hunter Froemming

What is *Audionyx*?

Audionyx is a web-based audio visualizer that turns music into interactive 3D visuals using three.js and WebGL. It lets users upload and store their music, offering a personalized experience where they can manage and replay their songs within the visualizer.

What is our Vision?

*Our **vision** is to craft a personalized user experience through captivating visuals with adaptive designs that respond dynamically to music and sound.*

*Audionyx aims to target a **userbase** of people who want to enjoy music in a new and visually captivating way.*

Demo Time! :)

Rating Tools & Methods

❖ Project Management!

- We approached this project using and AGILE methodology with use of Github's Project Tracker as we were able to track where our project was going, where bugs were, and who was responsible for what (with use of Story points and Epics to allocate importance of features)
- Additionally, we had regular meetings upwards to 3 times a week to ensure everyone was on the same page regarding commits, workload, and if anyone needed help.
- MS Teams, iMsg, Pair Programming

❖ Github's Project Tracker

- Necessary for distributing work amongst the team and determining priority through story points
- It had a great feature for logging bugs and features, but our team mostly communicated these bugs the moment they happened

❖ HTML, Handlebars, Wireframe

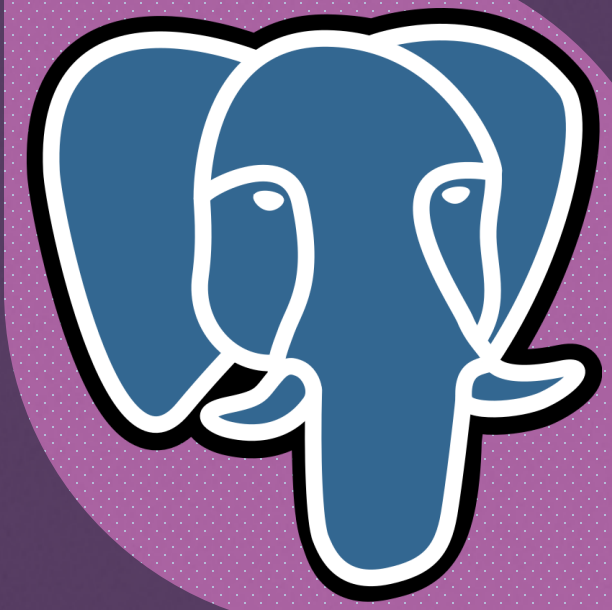
- Wireframe was useful for creating layouts for each of the pages, especially when working simultaneously on front-end UI
  - Sign-up, Sign-in, Library, and Visualizer pages
- Handlebars was necessary in the project for its reusability, as it made making a finite structure for each page completely doable.

❖ Github's VCS Repository

- Github's version control system was great for reverting changes when a push broke the project
- We especially used 'git stash' when pulling changes and it made it very easy to merge and maintain changes.
- Every member had complete access to who made commits, keeping the team organized and on same page

❖ Database (PostgreSQL)

- SQL was necessary in storing user information and integrating with authentication middleware.
- We created tables such as users, storing their usernames and passwords, as well as a library for storing the titles, genres, and urls, alongside the capability to delete this information when necessary
- Unable to store larger files, such as MP3 files (also due to storage limitations)



❖ IDE (VS Code)

- VS code was great for keeping code organized, especially for file management, formatting and readability
- Its ease of use and familiarity, especially when pushing or pulling from the repository made it easy to find errors and make changes

❖ CSS, Bootstrap

- Bootstrap's built-in styles with CSS simplified implementation of several aspects of the program (navbar, images, modals, spacing, ...)
- Not to mention very time saving
- CSS made stylizing and creating a theme for our site straightforward, especially when adding animations/gradients

❖ NodeJS

- Seamless for programming both the frontend and backend with JS
- Event-driven architecture was critical in creating a user-friendly site

❖ Frameworks (Express, Docker)

- Docker was ideal for testing and identifying missing dependencies and issues with the program
- Express was essential for simplifying routing and middleware management, limited in features
- Express.js handled endpoints and different HTTP requests such as GET, POST, PUT etc.

❖ Javascript, three.js

- Javascript was fundamental in making the website interactable for the users, especially in using modals and event-driven programs
- Three.js allowed for 3D Graphics Rendering, specifically for our visualizer, allowing us to create, manipulate, and display 3D objects, scenes, cameras, and lighting all within the browser

❖ Render (Deployment)

- Kudos for being free to use, storage limit made us change our plan on storing files
- Not new developer friendly, it was a bit difficult to understand how to make deployment work, and their FAQ was limited
- Needed a lot of time and help to get deployment working, especially as a Mac user, having to install Brew in addition

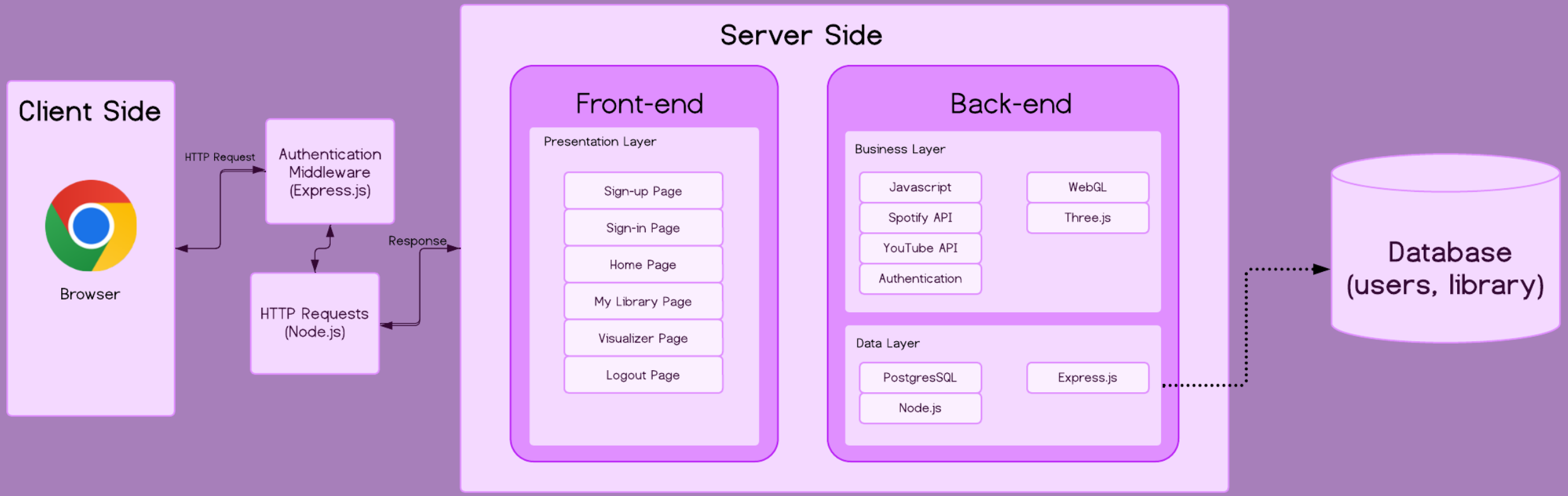
❖ External APIs

- We integrated Spotify and YouTube's APIs in order to use them to convert links obtained from their videos and songs and convert them to MP3 files for our program.

❖ Testing Tools (Mocha & Chai)

- We used Mocha and Chai as testing tools to ensure the reliability and functionality of our application.
- We preferred testing the application personally as we were testing a visual object to function, difficult to test using only code.
- The tools were useful, but our application found little use in testing our program.

Audionyx's Architecture Diagram



Challenges our Team Faced

- While working on our website we faced numerous challenges:
- Linking pages to static files such as CSS files and PNG files.
- Getting the visualizer to work with mp3 and Base64 files.
- Getting the database to work with the converters, API routes, and library.
- Render limited our team to 500mb, and our PostgresSQL database container was unable to take large files like our MP3 files.
- Yt-dlp.

Our Solution

- Our solution in approaching the limited storage was instead of storing each MP3 file for each user, we decided to instead store the Base64 encodings in the database. From there we were able to store these Base64 encodings in the database and use the converters to decode them back to MP3 to work with our visualizer.
- As for linking pages, there was a line in index.js causing file connection to not be possible.
- We reconfigured the database to work with Base64 integration and correctly matched the variables from the converter and visualizer.

Future Enhancements

1. Improve visualizer design with more customization from the user, including background images, effects, and further design choices.
2. Improve the database and incorporate the above personalization to allow for user choice.
3. Add capability to use other media links.