Music Streaming App

Introduction:

This project focuses on the development of a music streaming app, a sub-topic within the entertainment domain. The primary objective is to provide users with a highly responsive and user-friendly platform for accessing a comprehensive collection of music. A standout feature of this app is its ability to store user preferences and leverage this data to deliver personalized song recommendations. Emphasizing responsiveness and user-friendliness, the app aims to ensure a positive user experience, minimizing frustration or dissatisfaction. Furthermore, the app will be available to users at no cost or follow a Freemium business model, offering additional premium features for a fee.

Features:

As the main feature of the software is to provide the functionality for which the software is built, so keeping that in mind, below is the list of features our app should have:

- Listening to Music
- Favorite list
- Music Recommendations

Architecture:

As the project (Music Streaming App) uses Next-JS and TypeScript Technology for development, to structure and organize files and components we used modular architecture and structured using the components-based approach where each components have its own logic and functions. The key directories (Folders) are the following with proper documentation on what is inside of the folder etc...

- **App**: app is the main directory for all the components, assets and utils, etc.
    - **Auth:** in this Directory all the signup and sign-in files and folders are situated.
        - **Sign-up**
        - **Sign-in**
- **Components**: in this components folder all our components (JSX, TSX, CSS, etc.) files are stored and are called whenever we want to use them.
- **Assets**: assets are the folder where we store all the images and assets we have to store for our project like logos, some kind of GIFs, etc.
- **Utils**: in this directory, we store all the helper function we use in our application (Music Streaming Application).

User interface:

The user interface of our Music Streaming Application is designed to be user-friendly and easy to use. We used TypeScript, a powerful version of JavaScript that provides extra features and tools for development, along with Next.js, a React framework that enables server-side rendering (SSR) and provides several other features for building modern web applications. With TypeScript and Next.js, we ensured an efficient development process. Our main goal was to create an attractive user interface with an appealing design, smooth transitions, and engaging animations that leave a positive impression on users before they start enjoying the music experience.

Technology stack

The technologies we used in this project are the following:

- **Visual Studio Code** (**Text Editor**): Visual Studio Code is one the very popular Text Editor for building small projects, it supports pretty much all of the languages because there are extensions for every language to support the VS code (Visual Studio Code). You just have to install the extensions with some packages (if needed) and you're good to go.
- **Next.js:** User-friendly UI powered by TypeScript & Next.js for seamless development - appealing design, smooth transitions & engaging animations.
- **Figma**: we use Figma for prototyping and designing our project (Music Streaming Applications), Figma is also a very popular designing tool.

Development process

Keeping the software development life cycle in mind, we first gather all the requirements and functionalities we could need in the project. Could we add a sign-up and login page or not (which is actually implemented at the end) then we have to design those requirements, design is something where we convert imagination to real worlds models which is basically hard to do. Then we look for APIs to use for fetching information about music like tracks, lyrics, titles, metadata, etc. but we couldn't find all out there were paid or had Freemium offers and there are some free APIs but they are not going to fit our project. That's why we came up with the idea to add some to the website which will work like a local server and we have uploaded those (Songs) to the website where we had hosted the demo. then we came across pretty much all the testing sets (unit testing. Integration testing, etc.). then we had to host the app to the [Netlify] or [GitHub Pages] and live at [Sound-vibe.app].

Security

Security is a top priority for us in our application. We ensure the protection of user credentials by implementing basic username and password authentication. Additionally, we offer the convenience and security of social media login options through Google OAuth2 and Facebook login. These authentication methods allow users to securely access their accounts using their existing Google or Facebook credentials. By leveraging these trusted authentication systems, we maintain a high level of security for user logins and protect their personal information.

Process model: