A picture containing company name

Description automatically generated

Gig-Guide

Project Engineering

Year 4

Patrick Hession

Bachelor of Engineering (Honours) in Software and Electronic Engineering

Atlantic Technological University

2023/2024

< Camera 
Al recommendations for 
you... 
Danny Brown 
Olivia Rodrigo - Guts World Tour 
Profile 
Home 
Settings 

**Gig- Guide Final App Layout**

**Declaration**

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of Engineering (Honours) in Software and Electronic Engineering at Galway-Mayo Institute of Technology.

This project is my own work, except where otherwise accredited. Where the work of others has been used or incorporated during this project, this is acknowledged and referenced.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Acknowledgements**

I would like to thank the following people for helping immensely in the creation of Gig-Guide.

**Brian O’Shea:** [Project supervisor] Helped greatly with implementing ChatGPT API into my project and provided me with some example code to build off.

**Nathan Perry:** [Classmate] helped improve app U.I and gave me inspiration for initial colour scheme.

**Robert Muldoon**: [Classmate] helped bug test Gig-Guide

**Michelle Lynch:** [Lecturer] Helped with weekly check-ins on Project report and poster.

**Paul Lennon:** [Lecturer] Helped with fleshing out initial project idea.

**Niell O’Keefe:** [Lecturer] Helped with weekly check-ins.

**Table of Contents**

[1 Summary 6](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975700)

[2 Poster 7](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975701)

[3 Introduction 8](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975702)

[4 Background and Research 9](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975703)

[5 Project Architecture 10](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975704)

[6 Project Plan 11](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975705)

[7 Heading 12](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975706)

[7.1 Referencing 12](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975707)

[7.2 Notes on Content 13](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975708)

[8 Ethics 14](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975709)

[9 Conclusion 15](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975710)

[10 Appendix 16](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975711)

[11 References 17](file:///C:\Users\G00394619@atu.ie\Downloads\ProjectEngReportExampleTemplate%20(2).docx#_Toc64975712)

# Summary

The Gig-Guide project is a mobile application developed using React Native and Expo, designed to enhance the concert discovery experience for music enthusiasts. The primary goal of Gig-Guide is to leverage Spotify's API to recommend concerts based on a user's music preferences, thereby providing a personalized and engaging experience for concert goers. The app also uses Google places API to display maps of the venues.

The scope of the project encompasses the development of a cross mobile application built on react native and expo that integrates with Spotify's API, allowing users to log in and access their music data. This includes the use of OAuth for secure authentication and the retrieval of user's music interests. The application is built with many APIs which give it varying functionality. These APIS include ChatGPT for AI concert recommendations based on initial user music taste, Ticketmaster API to get up to date concert information as soon as its available and Google places A.P.I to provide venue location information to the user.

Key features:

There are so many features in Gig-guide that make it worth downloading. Let’s explore a few of these.

**OAuth 2.0 Authentication**: The core of Spotify's API integration is OAuth 2.0, a protocol that allows Gig-Guide to request access to user accounts on Spotify without sharing passwords. This process involves obtaining an access token and a refresh token. The access token is used for making API requests on behalf of the user, while the refresh token is used to obtain a new access token when the current one expires, ensuring a seamless user experience without frequent re-authentication.

**Expo's Auth Session:** Utilizing Expo's Auth Session functionality simplifies the OAuth flow. It handles the redirection to Spotify's authorization URL and the subsequent handling of the authorization code. This abstraction reduces the complexity of managing the OAuth flow, making it easier to implement secure and efficient user authentication.

Personalized Concert Recommendations Based on User's Music Preferences

Data Analysis: By leveraging Spotify's API, Gig-Guide can access a user's listening history, top artists, and playlists. This data is then analyzed to understand the user's music preferences.

**User Interface:** The recommendations are displayed in a user-friendly interface, making it easy for users to navigate through the concert options and find the ones that best match their preferences.

**Responsive Design:** The interface is designed to be responsive, ensuring that it provides an optimal viewing experience across a wide range of devices, from smartphones to tablets and desktop computers.

**Intuitive Navigation:** The app features a clear and intuitive navigation system, allowing users to easily find information about concerts, venues, and artists.

**Visual Enhancements:** The use of high-quality images, icons, and animations enhances the visual appeal of the app, making it more engaging and enjoyable to use.

**Venue Information:** The Google Places API provides detailed information about concert venues, including location, opening hours, and user reviews. This data is integrated into the app to give users a comprehensive overview of the venues.

**Map Integration:** By integrating Google Maps, Gig-Guide allows users to view the location of concerts on a map, making it easier for them to plan their attendance and travel.

ChatGPT Recommendations to Further Expand Users' Music Taste

**Music Discovery:** ChatGPT can generate recommendations for new artists, albums, or playlists based on the user's current listening habits. This feature helps users discover new music that aligns with their tastes and expand them.

**Firebase Auth Integration**: Firebase Auth provides a secure and easy-to-use authentication system. By integrating Firebase Auth, Gig-Guide can offer users a seamless sign-up and login experience. Users can quickly sign in with no hassle and Gig-Guide will remember they have done so thanks to firebase auth persistence.

**Notifications for newly released Concerts**

One of the main features of Gig-Guide is the instant expo push notifications. These are triggered whenever a new piece of data is added to my firebase database using real triggers which can be semi-reliable. To develop Gig-Guide in the feature I would utilize Firebase Cloud functions which allow for instant reactions to changes in a database which Gig-Guide needs.

# Poster

A screenshot of a computer

Description automatically generated

# Introduction

The Gig-Guide project aims to revolutionize the concert discovery process for music enthusiasts by leveraging Spotify's API to recommend concerts based on a user's music preferences. This initiative was motivated by the desire to create a personalized and engaging experience for concertgoers, who often struggle to find concerts that align with their musical tastes. By integrating with Spotify, Gig-Guide can access a vast database of user's music interests, thereby offering a tailored concert discovery platform.

The scope of the project includes the development of a mobile application using React Native and Expo, which will integrate with Spotify's API to authenticate users and access their music data. The application will then use this data to recommend concerts that best match the user's music interests. This involves the use of OAuth for secure authentication, the retrieval of user data, and the implementation of a personalized concert recommendation system.

The report will be structured as follows: It will begin with an overview of the project, including the goal, motivation, and scope. This will be followed by a detailed description of the project's approach, including the methods and technologies used, such as React Native, Expo, and Spotify's Web API. The report will then delve into the specific features of the application, the challenges encountered during development, and the solutions implemented to overcome these challenges. Finally, the report will conclude with a summary of the project's accomplishments and the main conclusions drawn from the project.

# Background Research

**React Native**

When I first had the idea regarding making a cross platform mobile app for my final year project I had no idea where to start. But after talking to my fellow classmate Patryk Milkiewicz he pointed me towards react native for its cross platform nature, massive community support and similarity to react which we had been learning already in our full stack module in college, taught by Brian O Shea.

React Native is an open-source framework developed by Facebook for building mobile applications using JavaScript and React. It enables cross-platform development, allowing developers to create applications that run on both iOS and Android platforms with a single codebase. This is achieved by rendering the UI to the native platform's programming interface, rather than creating a web-based view, which results in applications that behave like native apps, offering better performance and user experience.

React Native is built on React, a JavaScript library for building user interfaces, but it differs in that it targets mobile application development instead of web development. React Native provides pre-built components, libraries, and reference material, making it easier for developers to build applications without starting from scratch.

One of the key advantages of React Native is its active community and ecosystem of tools. Developers can seek support and share insights through channels like the Reactiflux Discord channel or attend React Native Meetups. This community support, along with the framework's performance and code reusability, makes React Native a popular choice for mobile app development.

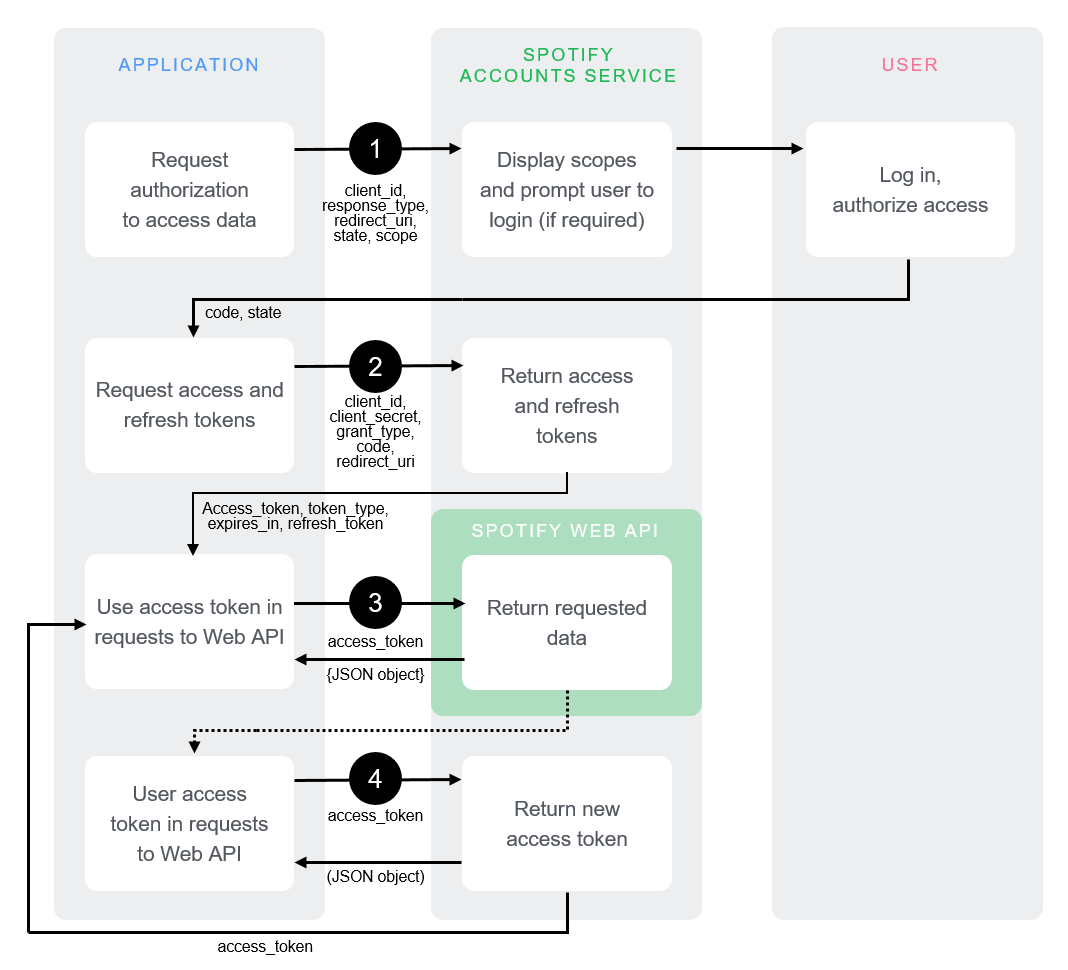
React Native's performance is comparable to native applications, as it renders applications natively. This means that even though the application is not specifically designed for a particular device, it can still use the same programming platform as a native application, leading to faster and more efficient execution.

The framework is supported by contributions from individuals and companies worldwide, including major players like Call stack, Expo, Infinite Red, Microsoft, and Software Mansion. This support ensures that React Native continues to evolve and expand its capabilities beyond Android and iOS, with projects like React Native Windows and React Native macOS exploring new platforms.

**Spotify Oauth Authentication**

A lot of my research throughout this project was revolved around the implementation of Spotify Oath and having a seamless authentication system in my project. I had a lot of questions on this topic when I first began this project such as.

* How do I authenticate users with Spotify using OAuth in a React Native Expo app?
* How do I refresh the access token to maintain user sessions without requiring them to log in again?
* What scopes do I need to request from Spotify to access a user's music taste data?
* How do I handle the authorization code flow to securely obtain user data from Spotify? I used a helpful medium article which really accelerated this progress [1]. This helped massively as it gives a solid idea of what a seamless Spotify Oath and firebase authentication could look like. This diagram which I found on the Spotify website really helped me to grasp this concept.



[2].

To authenticate a user’s Spotify within your app I learned you must utilise Expo's AUTH Session to handle the OAuth flow. This involves redirecting users to Spotify's authorization URL where they can log in and grant Gig-Guide access to their Spotify data.

I then needed to provide my Spotify app credentials (client ID and client secret) and specify the scopes Gig-Guide required. Scopes determine the level of access your app has to the user's Spotify data. For example, to access a user's music taste data, you might need scopes like user-top-read or user-read-private.

After the user logs in and grants access, Spotify will redirect them back to your app with an authorization code. This code is used in the next step to obtain an access token [3].

**Refresh the Access Token:**

Access tokens have a limited lifetime. To maintain user sessions without requiring them to log in again, I needed to use a refresh token.

When you first authenticate the user and obtain the access token, Spotify also provides a refresh token. Store this refresh token securely.

Before the access token expires, use the refresh token to request a new access token from Spotify. This process does not require the user to log in again, ensuring a seamless experience.

**Request Scopes for User's Music Taste Data:**

To access a user's music taste data, you'll need to request specific scopes during the authentication process. Spotify's API documentation provides a full list of available scopes.

For example, to read a user's top artists like I did for Gig-Guide, you might request the user-top-read scope. Adjust the scopes based on the data your app needs to access.

A screenshot of a computer program

Description automatically generated

**Handle the Authorization Code Flow:**

After obtaining the authorization code, I then needed to exchange it for an access token and a refresh token. This is done by making a POST request to Spotify's token endpoint (https://accounts.spotify.com/api/token) with your app credentials, the authorization code, and the redirect URI.

The response from this request will include the access token, refresh token, and the token's expiration time. Store these securely for future use. [3]

**UI:**  
During the initial development of Gig-Guide, the user interface (UI) significantly influenced my thought process. I dedicated considerable time to mastering the desired UI style for my project, aiming to evoke a sense of fun and excitement in the user. To achieve this, I utilized Figma to create a mock-up design for the project, ensuring that the app's purpose was effectively communicated and adhered to best practices in UI design. Here are the Figma designs I developed to give me of give me a good idea of what Gig-Guide should look like.

**Initial Figma Design Prototypes**

*A screenshot of a login form

Description automatically generated A screenshot of a cell phone

Description automatically generated A screenshot of a music album

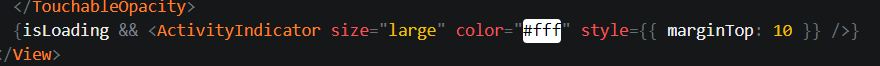
Description automatically generated A screenshot of a concert

Description automatically generated*

UI was very important to this project considering the theme. To accommodate for this I been studied various UI design techniques and principles. One of the key principles I've focused on is ensuring clarity and user control. This means making sure that the interface is straightforward and that users have the freedom to navigate and interact with the application as they see fit. [4]

I've also been exploring the importance of aesthetics in UI design during the development of Gig-Guide. This involves using colours, typography, and layout to create a visually appealing interface that is enjoyable to look at and use. By focusing on these elements, I aim to make Gig-Guide not just functional but also aesthetically pleasing.[5]

One feature I decided to implement in my code to ensure that the user had a good experience was loading buffers while waiting for the Api calls to return the concert and AI data. This was done so that users don’t become annoyed and confused that the data is not loading and instead tells them that it is on the way I implemented these in the following way.



This code calls activity indicator for the built in react-native library so that the user knows results are on their way soon.

To further improve the user experience, I decided to add a greetings message that dynamically changes based on the time of day. Using the following code.

A computer screen shot of text

Description automatically generated

**Fire Store:** Fire Store is the database I decided to use to store the concert results. I chose this as my database as it made the most sense as I was already using firebase auth for my login. Here is the configuration file for my firebase Auth. Here is Gig-Guides firebase database dashboard.

A screenshot of a computer

Description automatically generated

The user ID is gotten from the user’s firebase auth profile and allows for different storage between profiles. I decided to go with a firebase database as my project supervisor advised me to as I was already using Firebase authentication and they work seamlessly together.

**ChatGPT-API Usage**

When I first started to investigate the idea of a suggestion system for the artists, I thought I would have to use Python in my backend to suggest accurate concert suggestions to my users. However, I was soon convinced by my project supervisor Brian O’Shea to use a node express backend and utilise the ChatGPT A.I within that to achieve an A.I recommendation system. He then gave me example code which I changed to fit my own use case. Here is the A.I code I implemented in my project.

A screenshot of a computer program

Description automatically generated

The above code assigns chatGPT 4 a role as a helpful music enjoyer and then follows up with several prompts in order to receive concert recommendations. I had to tweak these several times to get the perfect recommendation from the API.

**Expo:** Using expo to build Gig-Guide was a really good experience as a developer. Expo streamlines the development process by handling app build, deployment, and submission, code-signing and certificate management, version upgrades, third-party library dependency management, and TypeScript integration. These aspects can be challenging with React Native CLI, but Expo simplifies them, making the development process more efficient and less cumbersome. When initially starting out Gig-Guide we hadn’t had our mobile app module yet as that was to come in semester 2. Starting out developing on expo using react native I was very lost initially but luckily I was recommended a video by my project supervisor Brian O’shea which greatly assisted in helping learn react native with expo. The video was by Academind and it guided me along in making the below todo list app so that I had a solid foundation in react native with expo.[7]

A screenshot of a computer

Description automatically generated

# Project Architecture

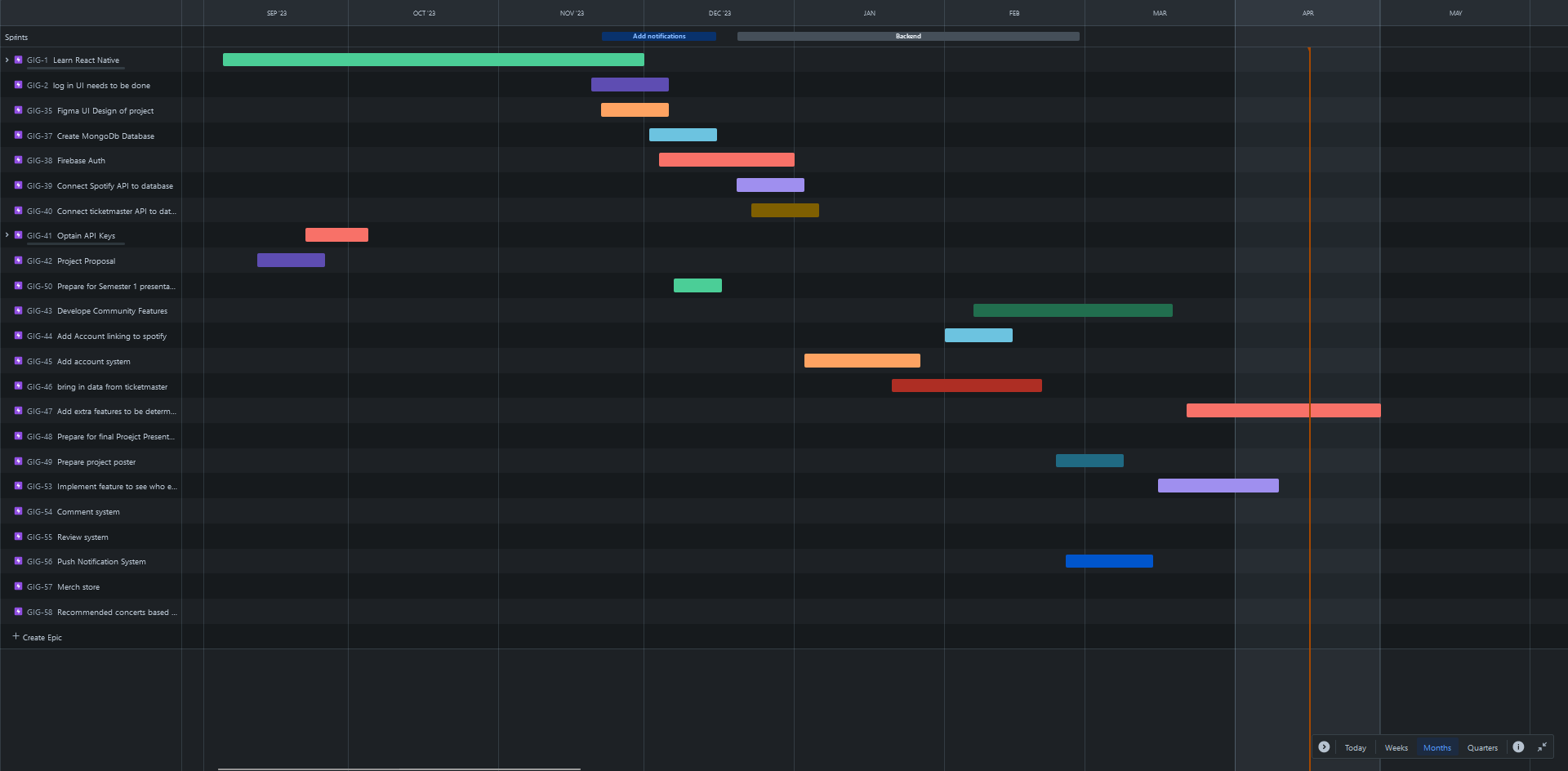
A screenshot of a cell phone

Description automatically generated

Figure 4‑1 Architecture Diagram

In my above architecture diagram, you can see that my user is initially greeted with a firebase auth login screen. In this screen the user inputs their username and password which are securely stored on my firebase auth database. After the user is securely logged in they are greeted with the home screen. Here they are then prompted with a Spotify login button. When this button is pressed the Spotify auth process has begun. Expo auth session then sends the authorization code received from Spotify to my backend. Here in my backend O auth exchanges the token for an access token. This then grants my backend code access to the users’ top artists. From here my backend uses the users’ top artists as query params for the Ticketmaster API. The Ticketmaster API then returns all kinds of details on the artists upcoming concerts such as the lineup, ticket links,

# Project Plan



Here is my original project plan that changed as time progressed. I had to change my goals as implementing Spotify auth took longer than initially planned. I heavily logged my project progress in my OneNote logs and used t hem to keep track of how progress was going myself. Throughout the project I also had weekly discussions with my class-mate Robert Muldoon to see how his project was developing. These weekly meetings really kept me accountable and made me stay on top of work. The logs for these meetings can be found in my Microsoft Teams logs. I also kept up to date with my agile Jira board which significantly helped with staying on track with my project.

A screenshot of a computer

Description automatically generated

# Ethics

An ethical consideration I had to make when designing this project is the security of the users Spotify. I must very careful during the authentication process. A lot of time was spent studying Oath 2.0 in to ensure the user could easily sync their Spotify without me having to store their details myself.

There are also always ethical concerns when dealing with ChatGPT. Fortunately, in my case I am safe from prompt injections due to the way it’s being used within in my application. It’s been run server side, and the user has no way of interacting with it. Prompt injections are when the ai can be asked to reveal sensitive information such as keys. I had to also be careful with my Spotify access token how it was stored used in my backend.

Firebase auth played a major ethical role in Gig-Guide initially as because I was so new to the JavaScript language and expo development, I was not confident in securely storing users passwords myself using a method such as B-crypt. Luckily Google firebase offers a very easy no hassle way to add login functionality to an app/website without the worry of the developer themselves having to handle all of that. I used a great video by Dan’s react Native Lab which really helped me with adding a firebase login to my code.[6] Below id=s my firebase authentication config that sets up authentication in my project.

**A screen shot of a computer screen

Description automatically generated**

# Conclusion

Gig-guide in its current form offers the users a unique way to find concerts in their area with an appealing user interface and alerts them to concerts from their favourite artists while also allowing them to discover new artist using my ChatGPT AI recommendations system. I learned a lot about making an expo app and deploying a node express backend as I went and the hurdles, I faced only further developed my learning. I would love to continue developing Gig-Guide further as I see great potential in the project. I was the further the development of this project I would add in more community centric features like following and comment sections for each Gig-Guide. I believe this could set the app apart from other concert recommendation applications and make it worth downloading. I have already planted the seeds for future development with my research into Firebase cloud functions which can be used, or a more accurate notification system based on changes that happen in the database. Finalizing the addition of this feature is the next plan I have for Gig-Guide. In conclusion I had a great time developing an app which revolved around something that I’m passionate about and it made me really invested in learning the various technologies that made my project happen such as expo and node express. This passion makes me want to continue to develop Gig-Guide and see where I can go with it next. Thank you for your time.

References

[1] Gareth Cronin , "R Using the Spotify API with Firebase to build an album-centric music manager, [Online]. Available: Using the Spotify API with Firebase to build an album-centric music manager/. [Accessed 29 04 2024].

[2] Authorization Code Flow [Online]. Available https://developer.spotify.com/documentation/web-api/tutorials/code-flow/.[Accessed 23 02 24].

[3] Imdad Codes, " Spotify API OAuth - Automate Getting User Playlists (Complete Tutorial) [Accessed] 15 04 2024.

[4] NALSengineering “10 Principles in UI Design: Enhancing User Experience through Practical Examples”,[Online] <https://medium.com/@NALSengineering/10-principles-in-ui-design-enhancing-user-experience-through-practical-examples-9d519e91b515> [Accessed] 26/4/24.

[5].Michal Malewicz “How to Design User Interfaces for Users”,[Online] <https://www.interaction-design.org/literature/topics/ui-design> [Accessed] 29/4/24.

[6] Dans React Native lab “How to Create an Expo App with React-Navie Firebase”, [Online]

<https://www.youtube.com/watch?v=mZlKwRV4MC8&t=1s> [Accessed] 28/4/24

[7] Academind “React Native Crash Course Build a Complete App”,[Online] <https://www.youtube.com/watch?v=VozPNrt-LfE&t=7245s> [Accessed] 27/4/24