|  |
| --- |
| CI601 – House Party |

|  |
| --- |
| Alexander wood - 22827042 |

Contents

[Introduction 2](#_Toc192847187)

[literature review: 2](#_Toc192847188)

[Spotify API terms of service: 2](#_Toc192847189)

[Spotify Web API documentation: 2](#_Toc192847190)

[Apple, Soundcloud and YouTube documentation: 2](#_Toc192847191)

[Research 2](#_Toc192847192)

[Tools and software being used 3](#_Toc192847193)

[Visual Studio Code (VS Code): 3](#_Toc192847194)

[GitHub: 3](#_Toc192847195)

[GitHub Projects: 3](#_Toc192847196)

[Brighton Domains: 3](#_Toc192847197)

[Planning 4](#_Toc192847198)

[Development 4](#_Toc192847199)

[Limitations 4](#_Toc192847200)

[Brighton Domains: 4](#_Toc192847201)

[Real World Testing: 4](#_Toc192847202)

[Design Decisions 4](#_Toc192847203)

[Languages Used: 4](#_Toc192847204)

[Mobile First Design: 4](#_Toc192847205)

[Minified Files: 5](#_Toc192847206)

[Basic Design: 5](#_Toc192847207)

[Micro Services: 5](#_Toc192847208)

[Risks 5](#_Toc192847209)

[Spotify API Dependent: 5](#_Toc192847210)

[References 5](#_Toc192847211)

[Appendix 6](#_Toc192847212)

[Appendix 1: 6](#_Toc192847213)

# Introduction

In this project I aim to create a website that will allow someone to link their Spotify account to a party that they create and then invite people to the party allowing them to search and add songs to the Spotify accounts queue without the need for an app or a Spotify account.

# literature review:

## Spotify API terms of service:

The Spotify API terms of service, I read through the terms of service to determine if my idea for the project would break any of the terms of service which would make the project illegal and unethical to make. The terms of service were very clear and made it very clear on what was possible to be made and what was not allowed to be done while using their API this was done by them breaking down their do and don’ts giving examples of what is not allowed for somethings to make it more clear and easier to understand.

## Spotify Web API documentation:

My next point of research was the API documentation to check to make sure that the project I wanted to do was going to possible. So, I started to by checking to see if the specific functions I needed for my project were present in their API and then followed this by reading through their API to find out how the Spotify login function works and how to handle the user’s authorisation tokens. I found certain parts of the documentation very useful, clear and easy to understand providing a sandbox playground which can be used to experiment with the API requests but there were certain parts of the documentation that required me to do more research outside of the documentation to find out how they work.

## Apple, Soundcloud and YouTube documentation:

# Research

Before starting work on the project, the first step was research into the Spotify API and terms of service to make sure that the API offered the features required to build to the website and that the website would meet all the requirements of Spotify’s term of service.

I also followed this by doing research into other music streaming services APIs to see if it would be possible to add support for any other streaming platforms in the future, but the research proved that it would not be possible as they do not offer the required functions in their API’s preventing support from being added.

# Tools and software being used

## Visual Studio Code (VS Code):

For the coding and development of the project I have decided to use the programming Integrated development environment (IDE) VS Code. I chose to use this IDE as I have experience using it and it also allows for a lot of customization to your preferences allowing me to make it easier for me to use and understand.

## GitHub:

To store a backup of my code and also allows for version control this allows me to create a repository and upload all my code and files to it storing it securely and also allowing me to create interaction of the code when it comes to adding features or changing features that could cause major issues giving me the opportunity to undo the changes I have done.

By using GitHub, it allows me to control changes through branches allowing me to separate new changes from a working and functional build by having a stable branch and a dev branch once testing is done and I know all the changes pushed to the dev branch are functional I can then merge these changes into the stable branch.

## GitHub Projects:

To keep track of the project I decided to use GitHub projects I did this as it keeps both the code and the progress tracker together while also offering the same features as other project progress trackers.

## Brighton Domains:

To Host the system, I have made I decided to use Brighton domains I did this as I already have access to Brighton domains allowing me to straight away start hosting my system including a database, database events, automatic PHP files, and HTML which meets all the requirements I needed.

# Planning

# Development

# Limitations

## Brighton Domains:

There were plans to use a WebSocket made using Nodejs to handle requests between the user and Brighton domains. I decided to use this as it would improve the performance of the user’s device and put less strain on the servers. But as Nodejs is no longer present on Brighton domains I instead opted to use a polling system which has got some performance drawbacks, but it is a suitable replacement for the WebSocket as it can be used to perform the same action.

## Real World Testing:

I had the idea to perform real world testing but due to me only being able to do testing with people from within the same module doing testing in the real world would be unethical and break multiple rules that are imposed on the assignment due to this I have decided not to do this kind of testing.

# Design Decisions

## Languages Used:

When it came to the languages that I decided to use to build the system I decided to use JS, HTML, CSS and PHP I decided on these as Brighton domains supports hosting of these types of files, and I have knowledge in how to use these programming languages.

## Mobile First Design:

Due to the system mainly going to be used on mobile devices the decision was made to focus on a design which focuses on mobile devices mainly and does not have a separate view for larger screens instead it just scales to the screen size. By doing this it allowed for some time to be saved and used to focus on features that are more important.

## Minified Files:

## Basic Design:

## Micro Services:

When building the system, I decided to build it using micro services this allows parts of the website to be reused in other parts as they as they work individually this also improves maintainability this is because the code is easier to find and also mean when updating the parts it updates across all parts of the website that use the function being updated.

# Risks

## Spotify API Dependent:

A major risk that affects the system is its dependency on the Spotify API this means that if the Spotify API becomes unresponsive or unavailable the system will not be able to work. This also means that if anything about the API changes that affects functions that is used in the system the system will need to be updated to support the new changes leaving it unavailable till the update has been done.

# References

*Spotify Web API (no date) Web API | Spotify for Developers. Available at:* [*https://developer.spotify.com/documentation/web-api*](https://developer.spotify.com/documentation/web-api) *(Accessed: 20 September 2024).*

*Spotify developer terms (no date) Spotify Developer Terms | Spotify for Developers. Available at:* [*https://developer.spotify.com/terms*](https://developer.spotify.com/terms) *(Accessed: 20 September 2024).*

*API reference  |  YouTube Data API  |  google for developers (no date) Google. Available at:* [*https://developers.google.com/youtube/v3/docs/*](https://developers.google.com/youtube/v3/docs/%20) *(Accessed: 10 December 2024).*

*Apple Music Api (no date) Apple Developer Documentation. Available at:* [*https://developer.apple.com/documentation/applemusicapi*](https://developer.apple.com/documentation/applemusicapi%20) *(Accessed: 10 December 2024).*

*SoundCloud for developers (no date) API - Guide - SoundCloud Developers. Available at:* [*https://developers.soundcloud.com/docs/api/guide*](https://developers.soundcloud.com/docs/api/guide) *(Accessed: 10 December 2024).*

# Appendix

## Appendix 1:

|  |  |
| --- | --- |
| Date | Meeting notes |
| 26/09/2024 | * Spoke about project idea * Talked about the legal side of the project * Spoke about what research needs to be done into the terms of service |
| 03/10/2024 | * Updated on legal research * Updated on project progress |
| 10/10/2024 | * Progress update on designing the project * Did ethical form * Updated on project progress |
| 24/10/2024 | * Updated on project progress * Spoke about submitting the Spotify API forms to have the project recognised by them |
| 13/11/2024 | * Spoke about the interim report focusing on what I have in It already and what I could add to it * Updated on project progress |
| 18/02/2025 | * Updated on development of the project * Shown a demonstration of a prototype working * Updated on the status of Spotify Application * Discussed features that could be added or discussed in report * Discussed the report of the dissertation |