|  |
| --- |
| CI601- House Party Interim Report |

|  |
| --- |
| Alexander wood |

Contents:

[Introduction: 3](#_Toc182493651)

[Motivation: 3](#_Toc182493652)

[Research and literature review: 3](#_Toc182493653)

[Spotify API terms of service: 3](#_Toc182493654)

[Spotify Web API documentation: 3](#_Toc182493655)

[Limitations: 3](#_Toc182493656)

[Brighton domains: 3](#_Toc182493657)

[Real world testing: 4](#_Toc182493658)

[Requirements: 4](#_Toc182493659)

[Must have: 4](#_Toc182493660)

[Ability to log into Spotify: 4](#_Toc182493661)

[Ability to join sessions and add songs: 4](#_Toc182493662)

[Ability to create and host a session: 4](#_Toc182493663)

[Should have: 5](#_Toc182493664)

[Ability for the host to filter out explicit songs: 5](#_Toc182493665)

[Ability for the host to set a duration for the party to expire after: 5](#_Toc182493666)

[The ability to generate and join a session using a QR code: 5](#_Toc182493667)

[Could have: 5](#_Toc182493668)

[Ability for the hosts to disable duplicate songs: 5](#_Toc182493669)

[Ability for a guest to hear a preview of the song they are going to add: 5](#_Toc182493670)

[Ability for the hosts to make it so the queue automatically when the session ends: 5](#_Toc182493671)

[Ability for the hosts to extend the duration of the session: 5](#_Toc182493672)

[Ability for the guests to see items being added to the queue in real time: 5](#_Toc182493673)

[Guests to see what songs are currently playing: 5](#_Toc182493674)

[Won’t have: 6](#_Toc182493675)

[Limit how many songs guests can add: 6](#_Toc182493676)

[Voting system to push songs up the queue: 6](#_Toc182493677)

[More music platforms: 6](#_Toc182493678)

[Chosen approach: 6](#_Toc182493679)

[GitHub: 6](#_Toc182493680)

[Project management methodology and tools: 6](#_Toc182493681)

[Plan: 7](#_Toc182493682)

[Design process: 7](#_Toc182493683)

[Programming and testing: 7](#_Toc182493684)

[Further testing: 7](#_Toc182493685)

[Risks: 8](#_Toc182493686)

[Spotify API Reliance: 8](#_Toc182493687)

[Deliverables: 8](#_Toc182493688)

[Logo: 8](#_Toc182493689)

[Website and server-side: 8](#_Toc182493690)

[References: 8](#_Toc182493691)

# Introduction:

In this project I aim to create a website that will allow someone to link their Spotify account and then invite people to a session allowing them to add songs to the queue while limiting their control over the music without the need to download an app or create an account to join the session.

# Motivation:

I aim to do this as there are multiple music streaming services and each service has their own way of allowing people to share and control each other’s music, but these platforms do not support each other meaning at a party there is no way to invite someone who uses a different platform, so this gave me the idea to create a website to allow people to invite people to add songs to their queue while also removing the need for a platform specific account or app.

# Research and 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.

# 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 it is a suitable replacement for the WebSocket.

## 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.

# Requirements:

A screenshot of a computer

Description automatically generatedWhen it came to figuring out the requirements for my website, I decided to use the MSCW method to help me prioritize the features and functionality that I will be adding to my project. This method is used to categorise the importance of each feature of the project into four different categories must have, should have, could have and won’t have.

## Must have:

### Ability to log into Spotify:

Allow the host to sign into their Spotify account so that it can be used in the session creation process.

### Ability to join sessions and add songs:

Allow users to join the session created by the host and add songs to the hosts queue through the website.

### Ability to create and host a session:

Allow the hosts once signed into their Spotify account to create and link their Spotify account to a session which will then be used by the guests to add songs to the queue.

## Should have:

### Ability for the host to filter out explicit songs:

Make it so the host can enable and disable a setting which either allows for guests to search and add explicit songs to the queue or hides the songs and does not allow them to add them depending on what the hosts decides.

### Ability for the host to set a duration for the party to expire after:

Add a setting when creating the session which allows the host to specify how long the session will last for and after the designated amount of time the session ends itself so people can’t keep using the session to add songs to the queue indefinitely.

### The ability to generate and join a session using a QR code:

Make it so a QR code can be generated and used as the way for guests to join the session.

## Could have:

### Ability for the hosts to disable duplicate songs:

Make it so the hosts can toggle on and off a setting which makes it so when a guest adds a song a check is made that makes sure the song being added is not already in the queue and if it is does not add the song to the queue.

### Ability for a guest to hear a preview of the song they are going to add:

Add a feature that makes it so the guest can search the name of the song select the song and then to check to make sure it’s the right song the guests can then listen to a short preview of the song.

### Ability for the hosts to make it so the queue automatically when the session ends:

Add a setting the host can enable or disable making it so when the session is ended or expires automatically the hosts song queue is cleared and set back to how it was before the session so there are no songs remaining from the session.

### Ability for the hosts to extend the duration of the session:

Make it so the hosts can increase the time they have remaining on their session in case the time they decided when creating the session was not enough so that a second session does not need to be created, and they can stay using the original one.

### Ability for the guests to see items being added to the queue in real time:

Add the ability for the guests to see the song queue and when a guests or the hosts adds a song to the queue either using Spotify or the website have it update showing the current queue.

### Guests to see what songs are currently playing:

Add the ability for the guests to see the song that is currently playing so that if they like the song, they can save it to their own playlist.

## Won’t have:

### Limit how many songs guests can add:

Add the ability for the hosts to restrict the number of songs each guest can add to the queue of the session and then after they hit the limit, they are then unable to add anymore.

### Voting system to push songs up the queue:

A voting system to be implemented that will allow for the guests to vote on songs in the queue and move them up the queue bases on how many votes the song gets.

### More music platforms:

As the website uses APIs to handle the playlists and song searching it would be possible to add support for more music platforms such as YouTube, SoundCloud and Apple Music.

# Chosen approach:

## GitHub:

To store and manage my code I will be using GitHub with two branches a stable branch and a dev branch. The stable branch will contain the code which has been tested and the dev branch will be used to contain the code which is currently being worked on and tested this is done so that I can use the stable branch as a point to revert to in case of any changes which break the code.

To protect the stable branch from changes that could be damaging, I have also decided to implement branch protection on it this means that to add code to the stable branch you must open a pull request from the dev branch it must then be reviewed before it can be pushed to the stable branch this it to monitor what’s being added to the stable branch and to prevent any untested code from being added.

## Project management methodology and tools:

The design methodology I will be using for this project is the waterfall methodology I will be using this for the design process to layout the requirements and the feature that will and won’t be in the final product of the project. Then when I comes to the development of the project, I will be combing the waterfall technique with a kanban board to help manage and know what I am working on and what’s left to be worked on.

The tools I will being using to manage the development is the projects function on GitHub which allows you to create lists of items and priorities them depending on their importance and then move them to a section to show that the task has been completed.

A screenshot of a computer

Description automatically generatedI have used this feature to create a Todo list listing out what’s currently being worked on, what’s been finished, what’s left Todo and an extra section that has features I could add if I have the time to near to the end of development. I have also created a MSWC to help identify the key parts of the project and to also identify what is not important for the project’s development.

# Plan:

## Design process:

At the beginning of the design process I started by coming up with the features the project requires, what I would like to add, what could be added and features that project won’t have in the first version. This then allows me to plan out how the coding of the project will happen

During the design process of the project, I will create basic sketches of the website to help with designing the layout for the website these sketches contain no colours and are very basic as they are only using to show the layout. This will then be followed by creating a more in-depth wireframe of the design using the basic sketch as the layout but then adding colour and making any changes to the layout if needed due to any issues seen when fleshing out the design more.

## Programming and testing:

Once the design process has finished I will then move onto building the project, as the website will be modular and each page will work separately from each other I will build the website page by page and doing testing on the page before I move onto making the next page by doing this I will be able to keep on top on the development as I will not be working across multiple pages or files making it easier. While I am working on the individual page I will be working at the same time on the server-side files for example the API adding the required functionality to the API as it is required for the page I am working on.

## Further testing:

Once I have a working version of the project I will then do Whitebox testing myself seeing if the website is able to handle the requests as expected and seeing if I can find any bugs that will causes issues down the line. I will then follow this up by doing Blackbox testing I will use this to see how people who have never seen or interacted with the website before will use and traverse the site I will then uses this information to then see if I need to adjust the layout of the website to make it easier to understand and use.

# Risks:

## Spotify API Reliance:

Since the project relies on the Spotify API to function any issues such as it being unavailable or changes to how the API works will affect my project and could causes problems with it such as it not working until the code has been updated to meet the new requirements of the API.

# Deliverables:

## A green and black sign Description automatically generatedLogo:

For my project I designed a logo which is going to be used as the favicon for the website this is important for the website as when someone shares an invite link or a link to the website the logo shows up, so I thought it was important to have on that shows the name of the project.

## Website and server-side:

At the end of this project, I aim to have a website that works and allows people to create sessions, invite people to the sessions and allows the people invited to add songs to the hosts song queue. Along with the with website I also aim to have a finalised database and server-side scripts to manage and maintain the sessions keeping them running for the specified amount of time.

# 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).*