Match Melody - Design Documentation

Emilia Skoins, Yizhou Zhang, Tianxin Wang, Xiangyu Sha, Rebecca Wright, Cameron McCurdy

Contents

1 Summary of Proposal

1.1 Background, Aims and Objectives

This design documentation will introduce and showcase what we have created for the software engineering team project. First, we will document a brief overview of the objectives and motivations for the project:

- To create a music recommendation website.
- To allow users to share their listening habits and music recommendations on the built-in social media page.
- Support Spotify's API and play music directly from the website.
- Create a recommendation algorithm to give each user distinct recommendations personal to them.

Furthermore, presented below are the detailed versions of the objectives which were initially presented in the Requirement Analysis document. We will look through these objectives one by one to identify how they will be achieved in the final project design:

Extracting music data from existing open-source databases, such as Spotify, and retrieving it to collect data on users' music preferences: Using the Spotify API, we can access the users' Spotify music data after they have logged in to our website and given their permission for access. This means we can use their personal music data to give specific recommendations based on their previous listening habits.

Implementing music playback functionality on the website and enabling users to interact with the currently playing song: The user will be able to play the songs directly within the 'Match Melody' website using the music player, which will be supported by the Spotify API. They can also add currently playing songs to their liked songs playlist without leaving the webpage.

Implement music recommendation functionality using an algorithm created by a combination of content-based filtering and collaborative filtering: Instead of having to collate all different types of music, store them and use the data points to make recommendations, we can use the Spotify Data Catalogue included in the Spotify API to create the music recommendation algorithm using content-based filtering.

Implementation of personalised music recommendations based on users' music preferences and historical interaction data: When the user logs into the 'Match Melody' website, using their Spotify account, they must accept certain permissions to allow us access to their Spotify music data. Using their previous listening history, we can match them to new songs based on certain track attributes such as danceability, valence or energy.

Perform security and reliability tests on the website: Various tests will be carried out on the website during the testing phase of the project. Security testing will be slightly more difficult considering we must first file with Spotify, before the website can go live and be used by the general public and we don't think we will receive this verification in time. However, we do have other methods available to provide security such as providing each user with details of what data we access, ensuring no data is stored locally, so there is no chance for data corruption, and all user data is protected by Spotify from using their API.

Evaluate the accuracy of the music recommendation system and determine its performance through testing and evaluation: Each team member will test the website music player to see how accurately the algorithm recommends songs based on our listening habits. The website will also be shared amongst friends and family to get more insight into whether the algorithm is recommending songs that are compatible with each individual user’s taste.

1.2 Changes to Specification

As we were progressing through the software project, we decided to make a few changes to the original plan. At first, we planned to include a social media page on our website, where a user could share their listening habits and meet new friends. However, with further research we discovered how much work would be involved in developing this, which was a lot more than we expected. We decided that the music recommendation algorithm and player would take priority, and the social media aspect of the website would only be implemented if all other aspects are completed first and we have sufficient time left to build it.

Also there has been a change with how we will fetch and store user information. In the original plan, we were going to implement this using a MySQL database but we were having some technical issues with using the program as a team. Through research we learned that as an alternative to MySQL, we could utilise the Spotify API to obtain the user information needed instead. We decided to switch to this plan of action instead, as the Spotify API was already going to be used within the website and it allowed for more efficient collaboration.

2 Design

2.1 Data

2.1.1 Data Types

The types of data that will be used in our system can be divided into the following kinds:

1）**Spotify API Access Related Data**

This is the developer’s own data information from ‘Spotify for Developers’, which is used for authentication and obtaining authorization and response from the Spotify API. To request the access token, data such as URL fragment identifier, auth endpoint (the URI to verify the request), client ID (the ID of developer), redirect URI (the URI of our website) and scopes (the authorization grant types) are needed. After a request to Spotify, JSON data including the access token will be returned. Below is an example of the response message [1]:

```
{
"access_token":"BQDBKJ5eo5jxbtpWjVOj7ryS84khybFpP_lTqzV7uV-T_m0cTfwvdn5BnBSKP
xKgEb11",
  "token_type": "Bearer",
  "expires_in": 3600
}
```

The token will be needed while using the functions provided by the Spotify API.

2）**Web Playback SDK Related Data**

The playback state of the website music player is required to initialise it and monitor the track’s playthrough situation. This data can be obtained by requesting the ‘Get Playback State’ from the Spotify API. Below is an example of the response [2]:

```
{
  "device": {
    "id": "string",
    "is_active": false,
    "is_private_session": false,
    "is_restricted": false,
    "name": "Kitchen speaker",
    "type": "computer",
    "volume_percent": 59
  },
  "repeat_state": "string",
  "shuffle_state": false,
  "context": {
    "type": "string",
    "href": "string",
    "external_urls": {
      "spotify": "string"
    },
    "uri": "string"
  },
//Due to limited space, the rest of the response is not showed
}
```

3）**Track Related Data**

Track related data is required in various functions of the website such as obtaining users top tracks, obtaining recommended tracks for playing and saving specified tracks. Once the top tracks are obtained, their certain characteristics will be extracted to be calculated for the recommendation. The list of top saved tracks can be acquired by requesting for the ‘Get User’s Top Items’ from the Spotify API. The following is an example of the response [3]:

```
{
  "href": "https://api.spotify.com/v1/me/shows?offset=0&limit=20",
  "limit": 20,
  "next": "https://api.spotify.com/v1/me/shows?offset=1&limit=1",
  "offset": 0,
  "previous": "https://api.spotify.com/v1/me/shows?offset=1&limit=1",
  "total": 4,
  "items": [
      {
      "external_urls": {
        "spotify": "string"
      },
      "followers": {
        "href": "string",
        "total": 0
      },
      "genres": ["Prog rock", "Grunge"],
      "href": "string",
      "id": "string",
      "images": [
        {
          "url":
"https://i.scdn.co/image/ab67616d00001e02ff9ca10b55ce82ae553c8228",
          "height": 300,
          "width": 300
        }
    //Due to limited space, the rest of the response is not showed
}
```

For the response, we extract the URI of the tracks to proceed to the next step. Saving specified tracks needs the token and the track ID. Below is an example of track ID [4]:
`11dFghVXANMlKmJXsNCbNl`

To play the recommended tracks on our website's music player, data such as track seeds, track URI and track image URI are required. This data enables the music player to play the track for the user and display the cover of the album. The track information can be acquired by requesting for the ‘Get Recommendations’ from the Spotify API. The following is an example of the response [5]:

```
{
  "seeds": [
    {
      "afterFilteringSize": 0,
      "afterRelinkingSize": 0,
      "href": "string",
      "id": "string",
      "initialPoolSize": 0,
      "type": "string"
    }
  ],
  "tracks": [
    {
      "album": {
        "album_type": "compilation",
        "total_tracks": 9,
        "available_markets": ["CA", "BR", "IT"],
        "external_urls": {
          "spotify": "string"
        },
        "href": "string",
        "id": "2up3OPMp9Tb4dAKM2erWXQ",
        "images": [
          {
            "url":
"https://i.scdn.co/image/ab67616d00001e02ff9ca10b55ce82ae553c8228",
            "height": 300,
            "width": 300
          }
        ],
//Due to limited space, the rest of the response is not showed
}
```

4）**Track Characteristics Related Data**

The data required to recommend tracks that are similar to the user's top tracks can be obtained by requesting the 'Get Track's Audio Features' from the Spotify API. JSON data will be returned as a response, consisting of several features and their values which are already calculated by Spotify. The following is an example of the response [6]:

```
[
  {
    "acousticness": 0.00242,
    "analysis_url":
"https://api.spotify.com/v1/audio-analysis/2takcwOaAZWiXQijPHIx7B",
    "danceability": 0.585,
    "duration_ms": 237040,
    "energy": 0.842,
    "id": "2takcwOaAZWiXQijPHIx7B",
    "instrumentalness": 0.00686,
    "key": 9,
    "liveness": 0.0866,
    "loudness": -5.883,
    "mode": 0,
    "speechiness": 0.0556,
    "tempo": 118.211,
    "time_signature": 4,
    "track_href": "https://api.spotify.com/v1/tracks/2takcwOaAZWiXQijPHIx7B",
    "type": "audio_features",
    "uri": "spotify:track:2takcwOaAZWiXQijPHIx7B",
    "valence": 0.428
  }
]
```

The recommendation algorithm of our website is mainly based on the characteristics of 'danceability', 'energy', and 'valence', which mainly determine the style of a song.

2.1.2 Data Flow

The following diagrams show the flow of data which occurs through the music player page of the ‘Match Melody’ website. The user will be required to log in with their Spotify account before accessing this page**.**

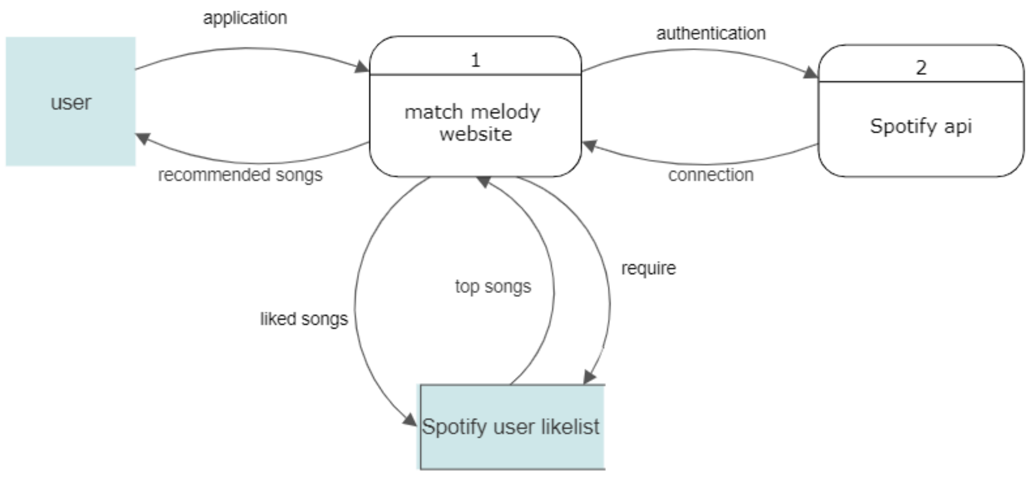


Figure 2.1.1 - Data Flow diagram of Spotify Authorization and Recommendation Level 0

In level 0 users send their application, such as requesting recommended songs or saving specified tracks, to the website. The website then requests the Spotify API to authenticate the user. As a response, the Spotify API builds a connection to the web playback SDK. The website requests the Spotify user’s top tracks which are received and returned. After processing, it returns the recommended songs to the user. To save a track, the website delivers the specified track information to the liked songs list of the user and stores it there.

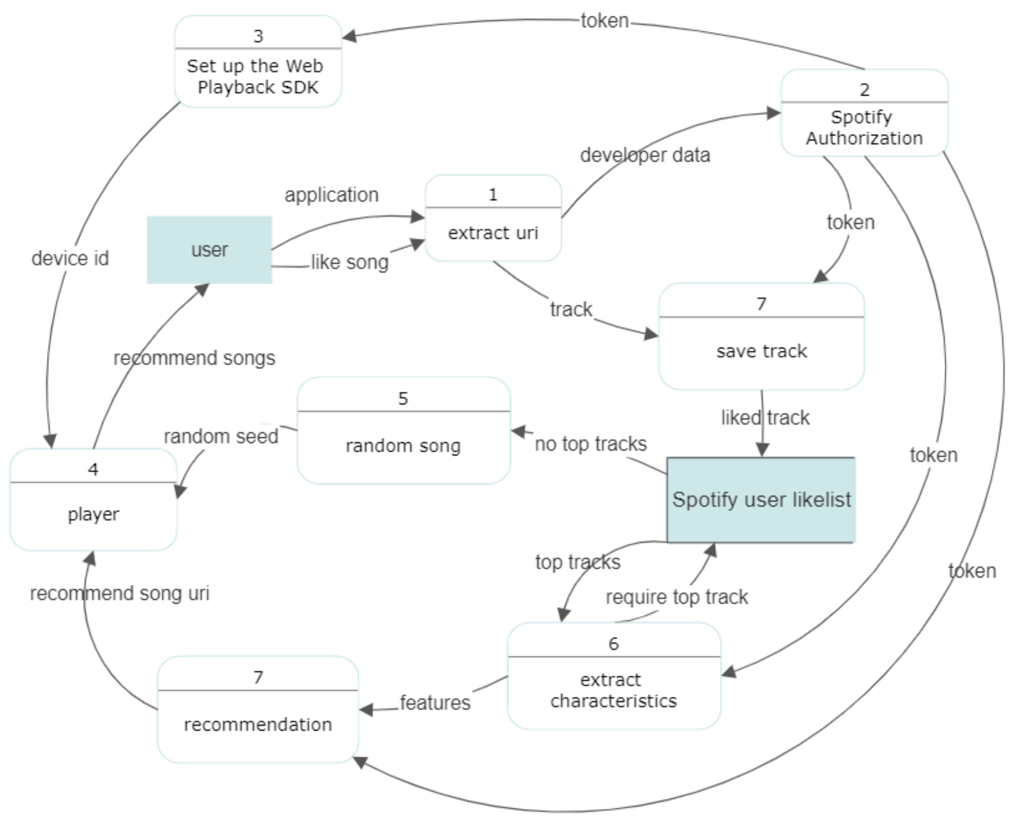


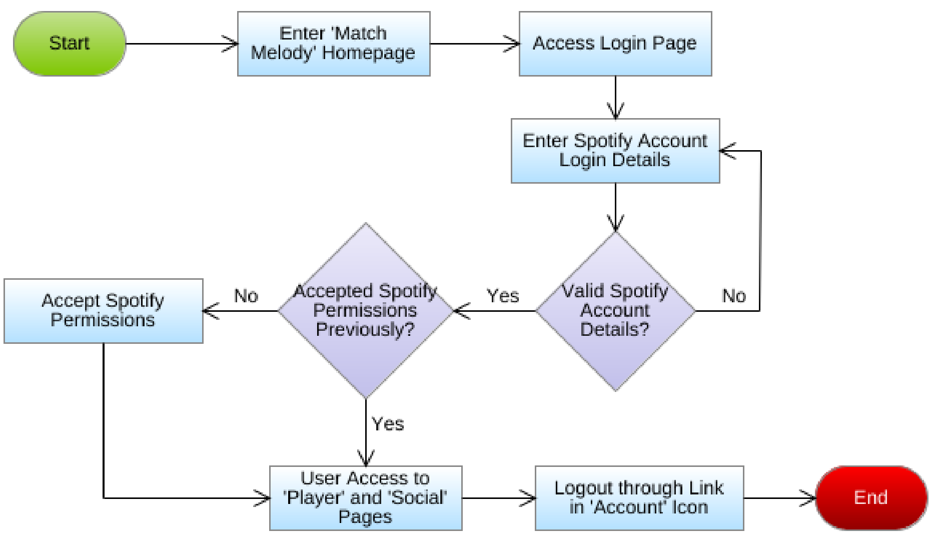
Figure 2.1.2 - Data Flow Diagram of Spotify Authorization and Recommendation Level 1

In level 1 users send their application, such as requesting recommended songs or saving specified tracks, and the URI of the saved track and the URI of our website is extracted. The URI of the website and other developer data, such as client ID, will be sent to Spotify to be authenticated. After authentication, the token is generated and can be delivered to the functions that need it: setting up the web playback SDK, saving a track, extracting track features and recommendations. After the web playback SDK is set up, it obtains the device ID of the user. Once the music player is in use, the device ID and the seeds or URIs of the tracks are needed to play the songs. To recommend songs, Top Tracks are requested and returned to extract the track characteristics. The track characteristics are used to obtain the recommended tracks.

2.2 Processes

2.2.1 Website Processes

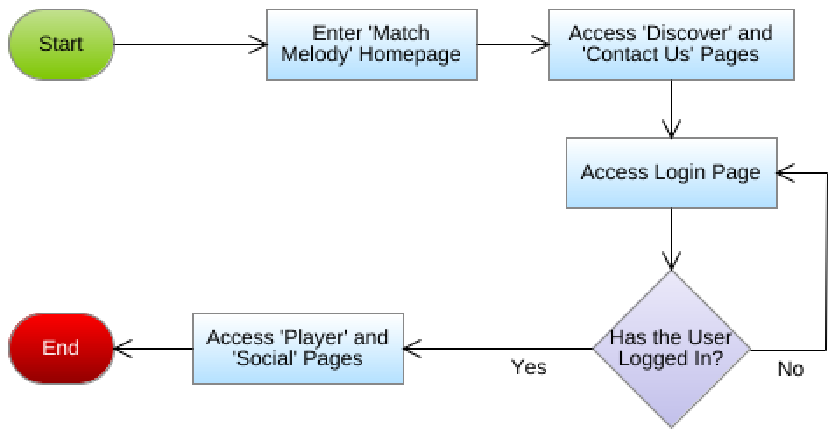
Each part of the system must work together in harmony for it to function well. The processes to be used within our website are described in the following flowcharts:



Start
Enter 'Match Melody' Homepage
Access Login Page
Enter Spotify Account Login Details
Valid Spotify Account Details?
No
Yes
Accepted Spotify Permissions Previously?
No
Accept Spotify Permissions
Yes
User Access to 'Player' and 'Social' Pages
Logout through Link in 'Account' Icon
End

Figure 2.2.1 - Flowchart Diagram of the Website Login and Logout Process

The above flowchart shows how a user will login to the ‘Match Melody’ website in order to access the main features, the music player and social media aspect, and also how to log out which is through a link provided in the ‘Account’ icon. It is a simple process, with not many steps, which provides ease of access for all users.



Start
Enter 'Match Melody' Homepage
Access 'Discover' and 'Contact Us' Pages
Access Login Page
Has the User Logged In?
No
Yes
Access 'Player' and 'Social' Pages
End

Figure 2.2.2 - Flowchart Diagram of the Website Page Accessibility Process

The above flowchart diagram shows all the website pages accessible to the user and how they obtain access to them. The Homepage, ‘Discover’ and ‘Contact Us’ pages are accessible to everyone even without logging in. We chose to do this so that a person can look through the website information and ask any questions, to see if the website would be suitable for them, before having to sign in and accept data permissions. The ‘Player’ and ‘Social’ pages can only be accessed after login, as they require the user's acceptance of Spotify permissions to access their music data for the page features to function.

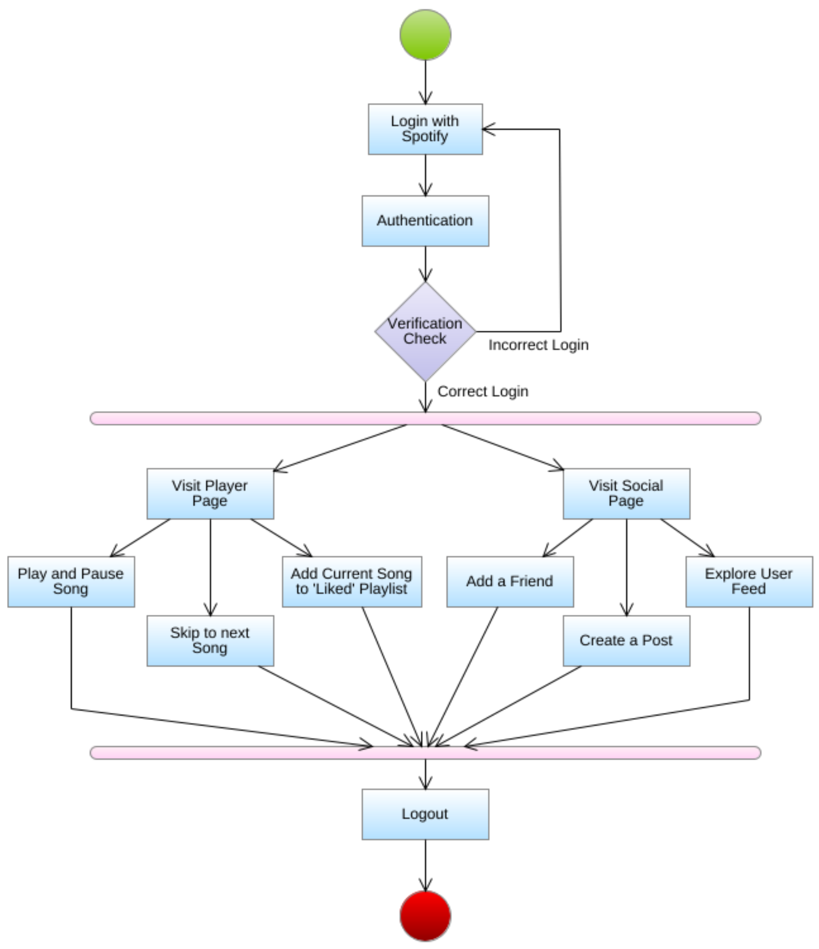


Figure 2.2.3 - Activity Diagram for During and After the Login Process

The above activity diagram shows the flow of activity through the 'Match Melody' website during and after the login process, describing all the functionality that is available to the user. This diagram helps us to visualise and describe the system so we can ensure the correct flow of actions, which would be simple and easy to follow for any user.

2.2.2 Recommendation Algorithm

A particular algorithm will be used in one of the main features of our website, the music player. This is the music recommendation algorithm. The problem which is to be solved by the creation of this algorithm is to provide a service which curates a list of songs to be played through the music player which is specifically unique to each user's music tastes.

The approach to solve this problem is to utilise the Spotify Web API in the creation of the algorithm [7]. Spotify is approaching 500 million users and has a catalogue of over 100 million songs [8] which can provide us with a good range of data to use in analysis. Using the Spotify Web API, users can sign into the 'Match Melody' website using their Spotify credentials and, with their acceptance, we can obtain their own unique music data. With this we can gather the users' top tracks from their listening history and make further song recommendations, based on certain track attributes. The OAuth 2.0 authorization framework will be utilised to authenticate all API requests and provide access tokens used to gain access to specific Spotify user data and resources.

To program the algorithm we will use JavaScript and JQuery. JavaScript is a popular language and it is also recommended to be used by Spotify itself. The Spotify Web API website has tutorials and references with samples coded in JavaScript. JQuery will be used to set up API calls to the Spotify Web API, which returns metadata such as artists and songs directly from the Spotify Data Catalogue.

The algorithm will be analysed using mathematical and experimental analysis. The performance is measured by how many recommended songs can be generated within a certain timespan and how closely they are related to the user's previous listening history. The experiments used to test our algorithm involve analysing the results of the recommended songs and comparing them to users' music data based on genre, artist and album. We will evaluate this by giving our test users a survey to fill out with questions discussing the accuracy of the recommended songs.

2.3 Interface

The project will place an emphasis on the user experience of the website, stripping down the process of music discovery to its basic elements and then integrating it into our website. Each page and element has been placed in a way to aid the navigation and flow of the website. We intend to divide the main intention of the user experience into two parts. Before login, the focus is on providing information in a non-overwhelming way, and encouraging the user to sign up. Once the user has logged in with Spotify, the objective is to deliver content to the user in a streamlined and visually appealing manner.

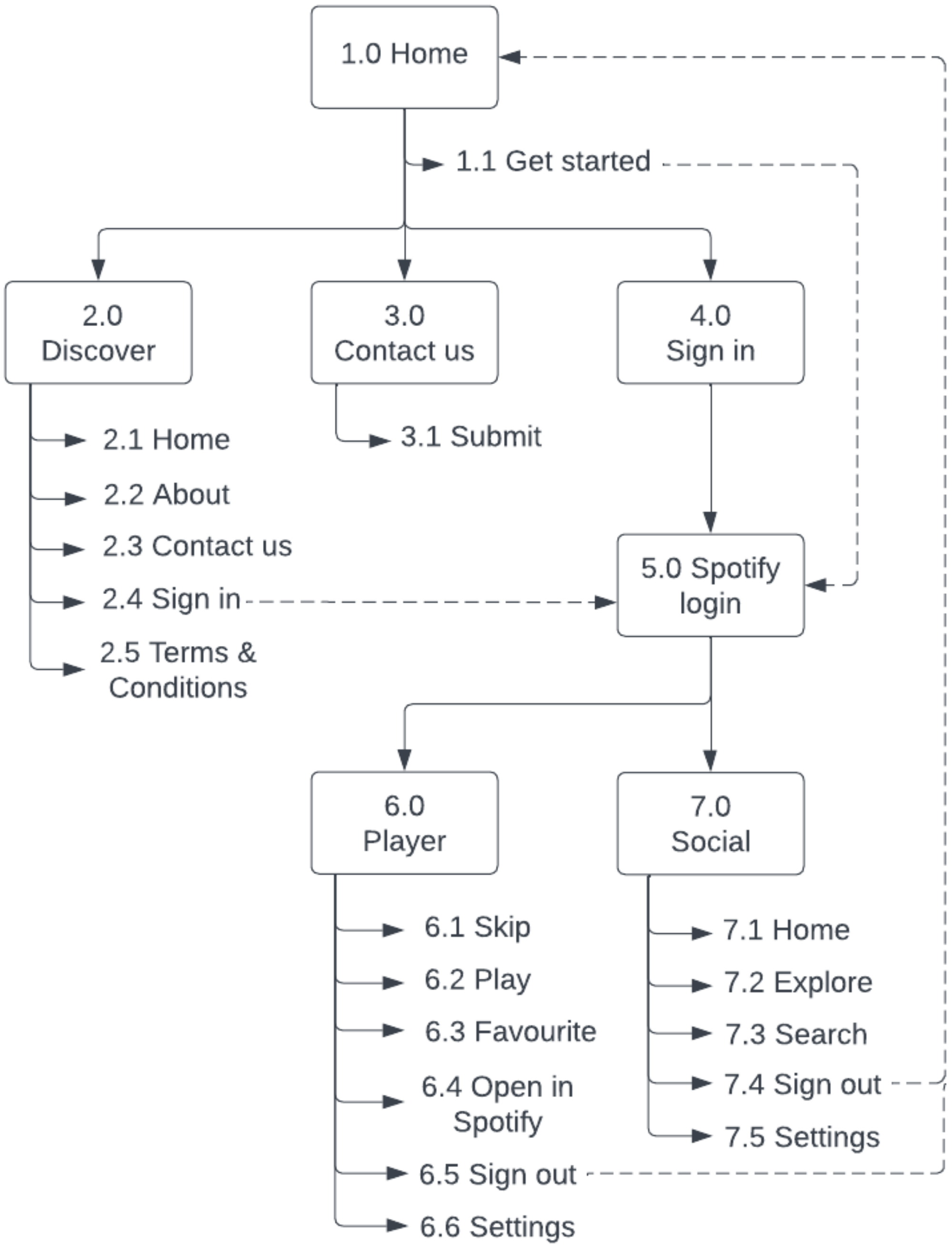
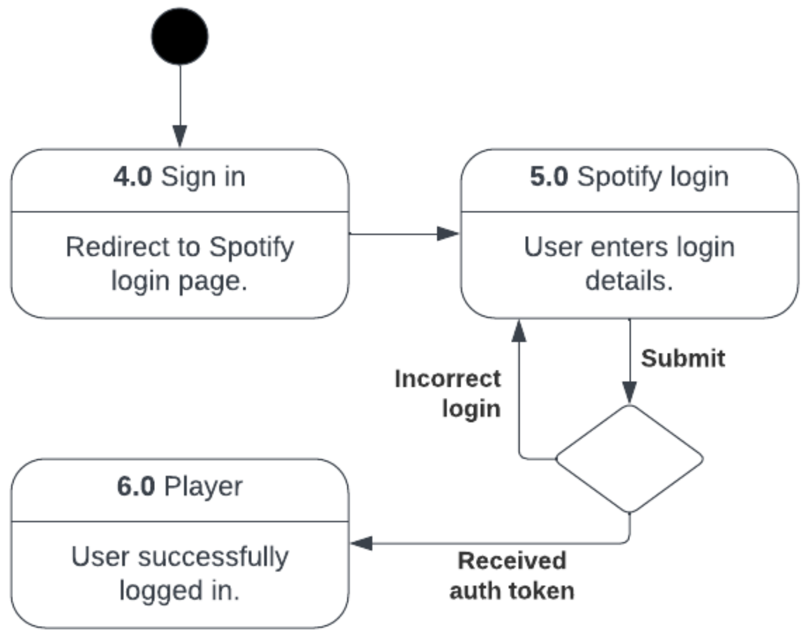


Figure 2.3.1 - Sitemap and Navigation Diagram

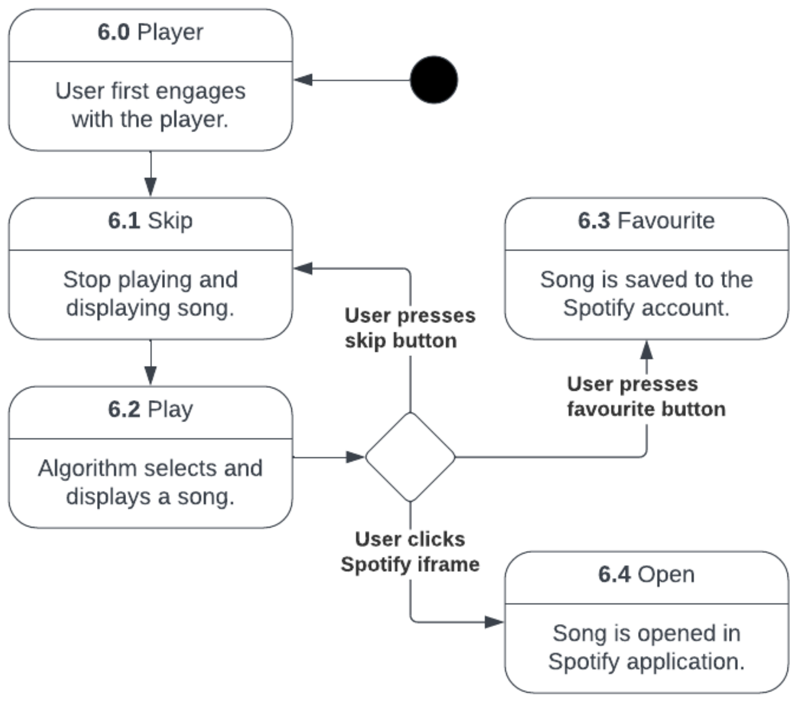
There are seven main components to the website as shown in the above diagram. The music player (6.0) and social page (7.0) can only be accessed once the Spotify login process is complete (5.0) and an authorization token has been received. Each sub-item of a page represents a clickable element on that page.



4.0 Sign in
Redirect to Spotify login page.
5.0 Spotify login
User enters login details.
Submit
Incorrect login
6.0 Player
User successfully logged in.
Received auth token

Figure 2.3.2 - Statechart Diagram of the Login Process

The above figure shows the process the user takes to reach the music player page. First, they are redirected to Spotify after clicking the sign in button. Then, once valid login information is entered, they are redirected back to the player page of the website with their Spotify account connected.



6.0 Player
User first engages with the player.
6.1 Skip
Stop playing and displaying song.
6.2 Play
Algorithm selects and displays a song.
User presses skip button
6.3 Favourite
Song is saved to the Spotify account.
User presses favourite button
User clicks Spotify iframe
6.4 Open
Song is opened in Spotify application.

Figure 2.3.3 - Statechart diagram of the Music Player Interaction

Figure 2.3.3 shows how the user operates the music player. Initially a simple slogan is displayed, showing the user that the skip button is clickable. Once they click it, a song is displayed, along with the functionality as described in the diagram. It is a simple loop of listening, favouriting and/or skipping.

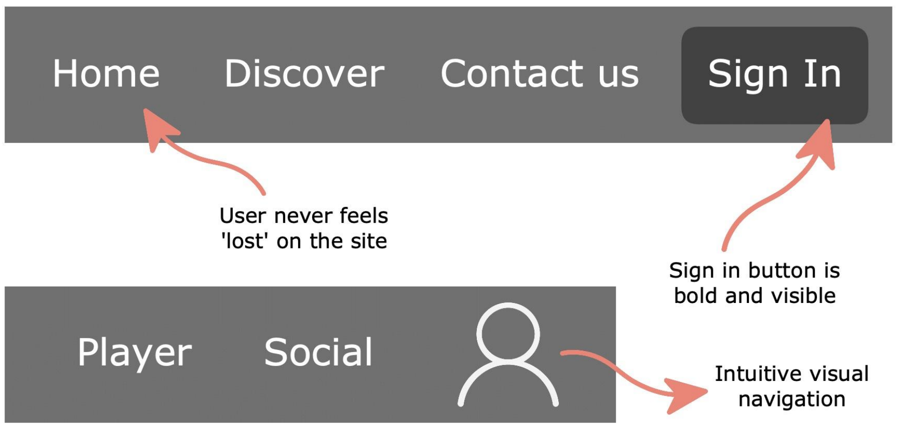

Home
Discover
Contact us
Sign In
User never feels 'lost' on the site
Sign in button is bold and visible
Player
Social
Intuitive visual navigation

Figure 2.3.4 - UI Mockup of Navigation Bar

The navigation bar required a lot of consideration as it determined how easily a user is able to find what they are looking for on the website. We opted for a straightforward design, drawing attention to the key links, such as sign in. Furthermore, in place of an ‘account’ link, an icon was placed in the navigation bar as it is more self explanatory and visually effective.

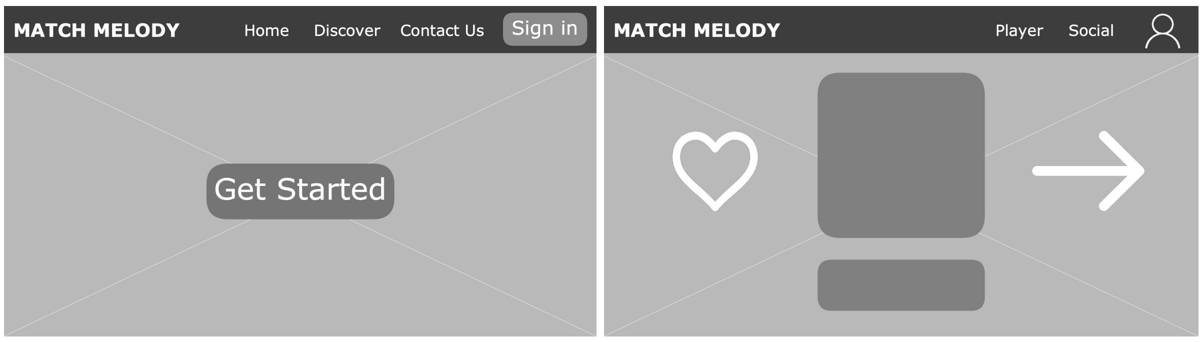


Figure 2.3.5 - Final Wireframe Mockup

The final wireframes of the pages show the structure and elements of each page. Our design is focused on ease of use, as well as being visually appealing. All buttons and links are positioned so that the user doesn’t have to look far for what they want. For example in the music player, all controls are central to the album art which is the largest element of the page.

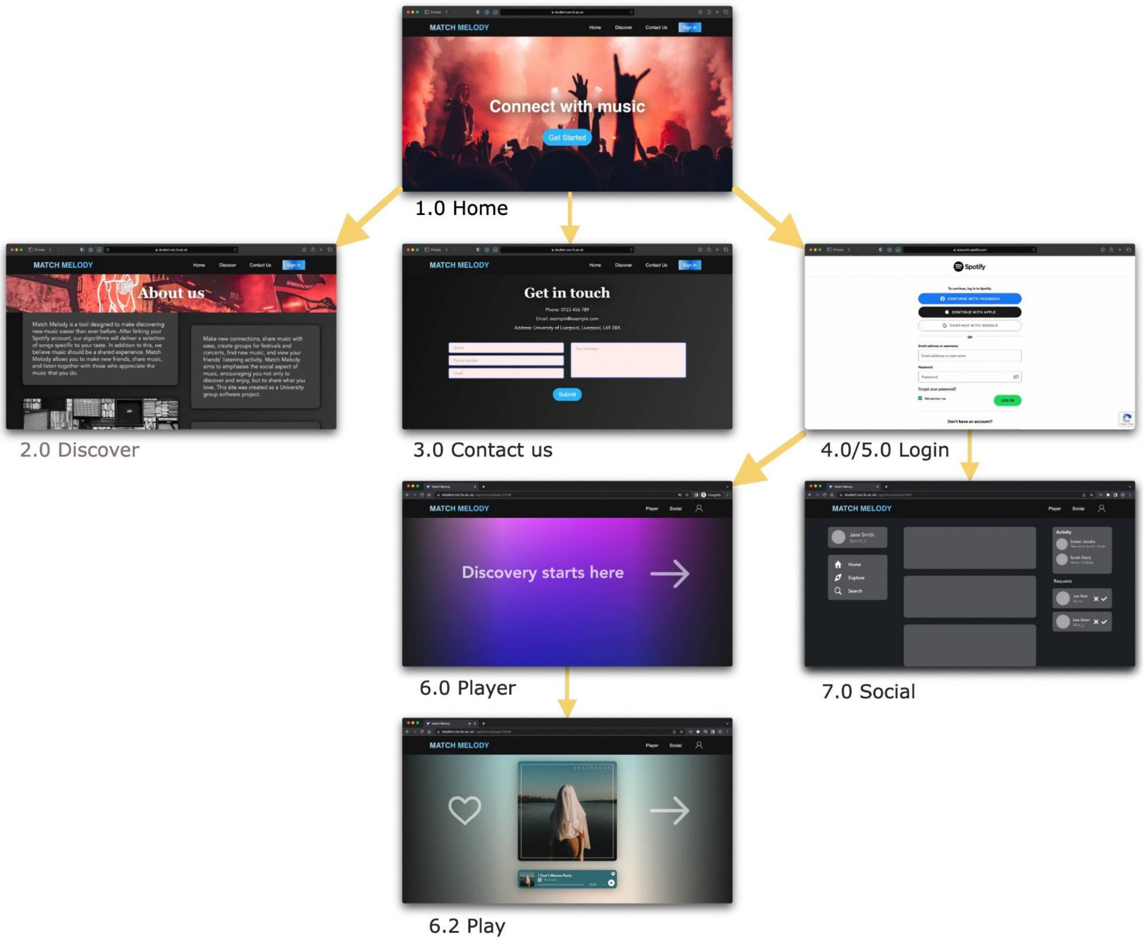


Figure 2.3.6 - Storyboard of the Site Navigation

__1.0__

Upon accessing the website front page, the user will be greeted with a simple webpage layout. A large bold image across the whole screen, a navigation bar along the top, and a large button which will link to the login page. The aim of the navigation bar is to make the user experience as simple and intuitive as possible. A high level view of the website's functionality is presented to the user on every page.

__4.0__

Before the user has signed in, there are 3 points of entry to the login page: the home page, the navigation bar, and the footer of the discover page. The aim of this design choice is to point the user in the right direction, and make the flow of the site as smooth as possible.

__6.0__

The music player page is intentionally designed to be as minimalistic as possible. The purpose of the website is to remove the unnecessary components of discovering music. A thumbnail of the album art is displayed in the centre along with the music player. On the left a favourite button, and the right a skip button. This simple layout should allow new users to interact with the player with no tutorial or introduction.

2.4 Evaluation

In order to reach our projects objectives, we developed some criteria for evaluating our website during the design stage:

- **Accuracy of the Recommendation Algorithm:** The website should effectively recommend new music to the user based on their Spotify Top Tracks.
- **Quality of the Website:** The website should operate smoothly and utilise its main features in the proper way.
- **Website Performance:** The website should respond correctly and in a timely manner to the different actions of the user.
- **User Experience:** The website should be easy to navigate, simple to use and provide clear instructions and feedback to the user.
- **User Satisfaction:** Users should be satisfied when using the website and agree that it provides accurate recommendations for their preferred music.

To achieve these criteria we have designed evaluation methods which include testing.

To determine the accuracy of the recommendation algorithm, we will conduct A/B testing. To implement this we randomly split test users into two groups, one using a website with randomly generated music and the other using the website with the recommendation algorithm. The behaviour and feedback of the two groups of test users will be observed, and the algorithm's accuracy is judged by comparing the proportion of the 50 songs generated to how many of those songs the test user saved to their liked songs list.

For the website to meet the expected quality, component testing is used to test that each component of the site is working as required.

Name of Component	Test Case	Test Steps	Expected Output
Login Button	Test of login function	1. Open the login page 2. Authorise access rights 3. Click on the "Login" button	User logs in to the website
Navigation Bar	Test of navigation bar function	1. Select the different contents of the navigation bar 2. Click on the corresponding button	The user jumps to the page corresponding to the navigation bar
Switch Songs	Test of switching songs	1. Click on the "arrow" to switch songs	Switch to the next song
Pause/Play Button	Test of the	1. Click on the	Pause/play music

	pause/play function	pause/play button	
Add Liked Songs	Test of adding songs	1. Click on "heart" button for the currently playing song	Song added to the user's Spotify list of liked songs

Table 2.4.1 - Competent Test

Test cases are run to verify the expected behaviour and record the results. Any bugs discovered should be fixed and regression testing performed to avoid additional bugs being created during code changes.

Integration testing is then used to test the various modules and components of the website to ensure that they work together, and interact well to achieve the overall functionality of the website. During integration testing, we simulate various user scenarios such as browsing through the different web pages, playing music and saving music to the liked songs list.

Performance testing can help us to identify any bottlenecks or performance issues, so that we can take appropriate action to improve the performance and reliability of the website. To improve the website performance, we use JMeter and PageSpeed Insights [9]. We will test the loading time, interaction time, visual stability and response time of the website based on user experience and website performance, and optimise the results.

Finally, in order to survey user satisfaction, we created a questionnaire for all test users of our website to fill out which can be found here: https://forms.gle/KgAQZ2dwDPCWGgsF9. The questionnaire focuses on user interface and experience, music recommendation, and the future potential of the website. We hope to gain valuable customer feedback from this which will help us to improve our final project even further.

3 Review Against Plan

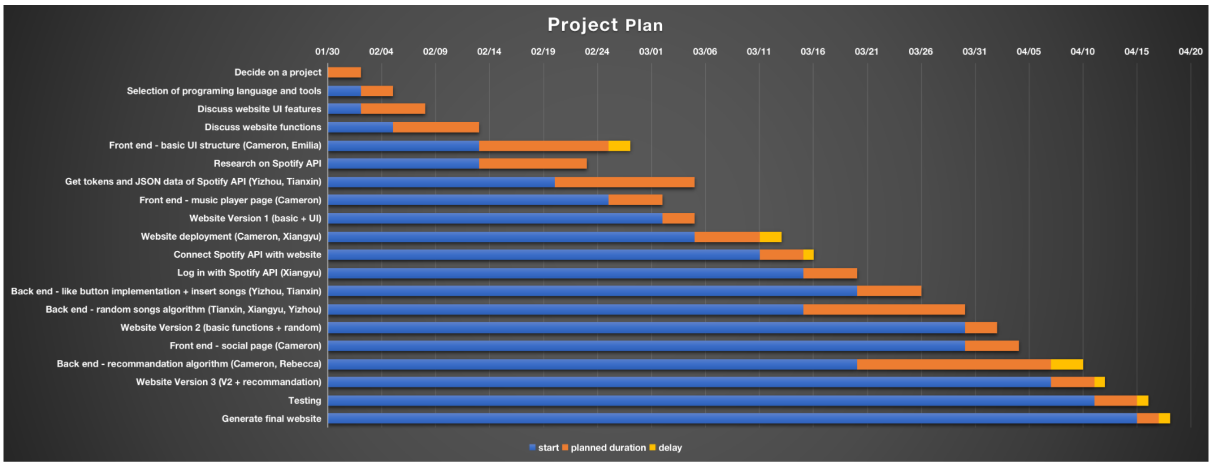


Figure 3.1 - Updated Gantt Chart

Our previous Gantt chart did not show detailed task descriptions and task allocation was not very clear. So we created an updated version of the Gantt chart with an additional element of who performs each task, as shown in Figure 3.1. The horizontal axis is a timeline of the entire project, showing visually the expected progress from the beginning to the end of the project. The vertical axis shows a broad summary of the tasks and the team members who will complete them. As you can see from the diagram, there is some sequential processing and some tasks that run in parallel. This gives us some extra time and allows our project to be completed more efficiently.

The first change from the original plan is that we tweaked some of the initially planned tasks. As discussed earlier in 1.2 Changes to Specification, we initially planned to use a MySQL database to fetch user information, but eventually decided to use Spotify user data by calling the Spotify API. As a result, we removed some of the tasks on the Gantt chart associated with building our own database and added new tasks relating to the Spotify API, such as "Get token and JSON data for Spotify API" and "Login with Spotify".

It is common for a project to deviate from its original vision during execution, especially in terms of time planning. During the project so far, we have encountered several technical difficulties that resulted in delays of the project schedule. The newly added delay bar in the updated Gantt Chart, represented in yellow, expands after the orange bar called "planned duration", indicating delays compared to the original plan. For example, we found that the first basic user interface designed showed some unexpected typographical errors when opened on different browsers and devices. Due to this, we spent more time than expected on additional debugging and testing to ensure compatibility between all browsers and devices. This is how the first delay occurred and the team's workload increased over the next few days, so subsequent tasks would not be subject to further delay.

Despite these initial setbacks, we hope to work collaboratively and effectively manage our time in the upcoming weeks, to catch up on delayed tasks and minimise the negative impact on subsequent tasks.

4 References

[1] developer.spotify.com. (n.d.). Getting started with Web API | Spotify for Developers. [online] Available at: https://developer.spotify.com/documentation/web-api/tutorials/getting-started
[2] developer.spotify.com. (n.d.). Web API Reference | Spotify for Developers. [online] Available at: https://developer.spotify.com/documentation/web-api/reference/get-information-about-the-users-current-playback
[3] developer.spotify.com. (n.d.). Web API Reference | Spotify for Developers. [online] Available at: https://developer.spotify.com/documentation/web-api/reference/get-users-top-artists-and-tracks
[4] developer.spotify.com. (n.d.). Web API Reference | Spotify for Developers. [online] Available at: https://developer.spotify.com/documentation/web-api/reference/get-track
[5] developer.spotify.com. (n.d.). Web API Reference | Spotify for Developers. [online] Available at: https://developer.spotify.com/documentation/web-api/reference/get-recommendations
[6] developer.spotify.com. (n.d.). Web API Reference | Spotify for Developers. [online] Available at: https://developer.spotify.com/documentation/web-api/reference/get-several-audio-features
[7] developer.spotify.com. (n.d.). Web API | Spotify for Developers. [online] Available at: https://developer.spotify.com/documentation/web-api.
[8] Ruby, D. (2022). Spotify Stats 2022 — (All Data & Stats Listed). [online] demandsage. Available at: https://www.demandsage.com/spotify-stats/.
[9] “Make your web pages fast on all devices,” *PageSpeed Insights*. [Online]. Available at: https://pagespeed.web.dev/.