CS322 Music Programming  
Project Documentation

Introduction

The project began in one of the CS322 labs, where our team (Jevgenij, Fabian, Fiachra) sat down at the table and discussed all the potential ideas for a project that we are going to work on. We opened the project ideas list and went through each idea carefully, finding what interests us the most. A couple of project ideas caught our attention straight away, for example “Mobile Sound App”, “Spotify Visualizer”, “Spotify Audio Analysis” and many others. After browsing the internet and researching about our potential projects, we stumbled upon the official Spotify website made for Software Developers at [developer.spotify.com](https://developer.spotify.com/). This page instantly caught our eyes, as the front page design was extremely appealing visually, with a message “Build with Spotify’s 100 million songs”. On the page there was a button “See it in action”, which would slide the page down and show you a lot of interesting back-end code for functionality such as getting you top 5 played tracks or get 5 recommended songs based on your top 5 tracks that you’ve listened to.

That was it, our team was immediately certain about our project direction. We aimed to develop a project utilizing the Spotify Web API, harnessing its capabilities to perform exciting tasks such as generating playlists directly in a user's personal Spotify account, as well as managing them within our application. Since every member of our team was a regular Spotify user, we found ourselves intrinsically motivated to kickstart the development process immediately. After studying the possibilities of what Spotify API can do for us, our team spirit went up and interest skyrocketed. During the next team meeting, our team discussed our strengths and weaknesses, who wanted to do back-end or front-end work, full stack and database work, or just overall design and wireframes. We then assigned roles and tasks for everyone to begin our development journey.

After a couple of days of meetings and setting up the project, our team came up with a brilliant idea to create a Google Chrome extension that would serve as the main hub of our application. This idea was aligning perfectly with what we wanted to create, and Chrome extensions are easy to install and convenient to use. So the way to access all the features of the application, was simply installing the extension and clicking on the icon when it’s added to the browser. It was possible to use the extension to open new browser tabs, therefore our team could work on different pages and then simply add them to the extension, for example, through buttons inside the extension UI.

It was time to allocate the workload equally between everybody in the team. Fiachra expressed interest in front-end work, so we assigned him to start off with some wireframe sketches and then use these sketches to create a React.js page that had a bunch buttons and boxes in it. Fabian took the initiative and assigned himself Database work, where he would connect our application with “Firebase”, and store various data like user information or song and playlist names. Jevgenij thought that the idea of having an extension at all times in Chrome browser was fascinating, so he took the job of creating the extension, its UI, and back-end functionality.

Our team has set up a GitHub repository for the project, wrote a “README.txt” file which initially contained a bunch of ideas for the project that we previously brainstormed. After everyone accepted the invitation to the GitHub repository, every member of the team created a separate branch and called it something along the lines of “name/beginning”, so we could all work simultaneously on different parts of the project, and then merge them together over time. We could now track each others progress, and see how we are getting on in the process. Not everyone in the team was familiar with GitHub, so we had a couple of sessions together where we go through the basics of Git, and perform actions like “git commit” or “git push”. These sessions were extremely valuable, as it is almost impossible to build a successful application without a proper version control system.

After some weeks have passed, our team started to feel like a group of developers, and we started to get new thoughts and ideas for our project, for example, what would be the target audience of our application. We had a couple of discussions and came to a conclusion that our application will be made for avid Spotify users who seek a more personalized and enhanced music experience. This includes music enthusiasts who regularly curate playlists and are always on the lookout for new recommendations and tracks that align with their tastes. Additionally, the project caters to tech-savvy individuals interested in the practical application of APIs and those who enjoy integrating their digital experiences. By offering features like creation and generation of playlists and providing refined music suggestions, our project was going to appeal to a broad spectrum of Spotify's user base, ranging from casual listeners to hardcore music fans.

At this point of development, we still did not have a clear objective of the project or the scope. What should our application do for our end users and why will it be useful. We started to discuss various features of Spotify, which features were good, which ones were bad, which ones were missing and so on. We realized that Spotify is missing a feature. Currently, Spotify has no way of crafting the playlists the way a user would like to. You are not able to move the songs or podcasts around the UI, when a Spotify playlist is created, all of the tracks in the playlist have static positions, and you can’t move them. This was a problem because sometimes a user wants to order 5 of their favourite songs in a specific order, and listen to them in a row. This was a great opportunity for our team to create a solution, and we came up with an idea to create a separate React.js page where a user uploads their playlist, and dynamically drags the tracks from the playlist into a new playlist, in the order that is perfect for them. The user would then save this new playlist that they created and save it inside their Spotify account. This was our main objective, but of course, an equally important objective was of course to learn as much as possible and improve our skills in software development, agile principles, communication and team work.

Methodology

Before our team wrote even a single line of code, we all agreed that we should work with React.js as it is one of the most popular libraries in the world and a lot of employers looking for graduates with React skills. React was also an ideal candidate to create a web page where you can easily drag and drop content from one container to another. Everybody in our team wanted to improve our coding ability with React.js, however, in the end not everybody got to work with it. For the Google Chrome extension, React.js was not ideal because of how the Chrome extension communicates with the back-end code and Spotify API, so Jevgenij stuck to vanilla JavaScript, HTML and CSS. Since Fabian started to work on Firebase database, he also didn’t get the chance to work with React. Fiachra was the one who got the opportunity to create a dynamic React page with an appealing design and fluid functionality.

Our team used Balsamiq in the beginning of the project to create some low fidelity wireframes, sketches and just to design the overall visuals for the extension, react page and other potential UI elements of the project. Our team was already familiar with Balsamiq as we all successfully completed the CS280 User Experience & User Interface module, so we were confident in our ability to create high quality sketches and designs.

1. **Project Background and Inspiration:**
   * Discuss what inspired your team to choose this project. Was it a shared passion for music, an interest in exploring streaming APIs, or the challenge of integrating with a well-known service like Spotify?
   * Mention any specific gaps or needs your team identified in the current market or personal experience that your project aims to address.
2. **Project Objectives:**
   * Clearly define the goals of your project. What did you set out to achieve with the Spotify Web API?
   * Outline the main features you planned to implement, such as playlist creation, track recommendations, or user authentication.
3. **Target Audience:**
   * Describe who the project is intended for. Is it for general Spotify users, music enthusiasts, or perhaps developers looking for a reference in Spotify API integration?
   * Explain how your project benefits this audience.
4. **Technology Overview:**
   * Provide a brief overview of the technologies used in the project, such as JavaScript, specific libraries, or frameworks.
   * Mention why these technologies were chosen and how they contribute to the project's objectives.
5. **Collaboration and Team Dynamics:**
   * Introduce your team and the roles or expertise each member brought to the project.
   * Highlight the collaborative tools used, like GitHub, and how they facilitated the project's development.
6. **Project Scope and Limitations:**
   * Outline the scope of the project, detailing what is included and perhaps what was considered but not included in the final version.
   * Acknowledge any limitations or constraints faced during development, such as API rate limits or scope of access.
7. **Significance and Potential Impact:**
   * Discuss the significance of your project in the broader context of web applications and API integrations.
   * Speculate on the potential impact or future applications of your project.