# Document Information

|  |  |
| --- | --- |
| **Project name:** | Spotify Groups |
| **Date:** | Date of writing: 29/1/2021 |
| **Author:** | Group Project Team 11 |
| **Owner** | Prof. Jason Quinlan |
| **Document code:** |  |
| **Version:** | 0.2 (Added roles) |

Note: this document may not be valid anymore. Please check the configuration management system for the latest approved version of the document.

# Approval

|  |  |
| --- | --- |
| **Date** | **Name and Signature** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Notes

# Definition

|  |  |
| --- | --- |
| **Background:** | Spotify is an integral part of life for many young students. Whether while doing a lab, walking down College Road or at a much-missed and now-illegal party, it’s allowed music to be everywhere. Spotify Groups combines this universality with the shared nature of post-pandemic things like Netflix Party or Disney+ Watch Together. |
| **Main Goal:** | To create an app allowing people to create a shared chat room that allows for simultaneous Spotify playback (one user presses play and all users hear playlist). If time allows, an instant messenger function will be implemented to allow communication. |
| **Desired Outcomes:** | Knowledge of new programming languages – at the moment that consists of at least React JS and Django Python. Create an app that is simple, fast and easy to use. Get deliverables in on time. |
| **Constraints and Assumptions:** | Time limit of 7/8 weeks, manpower and knowledge of university students, malleability of Spotify Web SDK |
| **Interfaces:** | No similar projects attempted before, although Sebastian has developed some Django storefront projects before. |
| **Project Approach:** | As deliverables are announced our team will assign roles to complete them, looking at the success or failure of meeting previous deliverables successfully. Each member’s prior knowledge will be used, i.e. Sebastian’s experience with Django and Allan’s experience with JS. There will be weekly scrum meetings with Jason Quinlan, but also regular meetings outside of timetabled hours held over Teams. |
| **Project Product Description:** | * As of 0.1: * UI that can create a ‘room’ and have other users on difference machines join or leave rooms. Once inside room, a Spotify prompt can be searched to find a song, which is then played to all users. * Possible groupchat message function is being discussed. |

# Outline Business Case

One of the first apps that exploded during the halcyon days of the first lockdown was Netflix Party. This Chrome extension synched video playback in Netflix/Disney+/HBO Max (requires cheeky VPN). Like Zoom, it had existed beforehand but gained a spike in interest due to the current global situation.

To our knowledge, however, there has not been a widely-adopted similar application for music streaming.

A business benefit would be for private listening parties. One popular bonus from donating large sums to fundraising platforms (i.e. Kickstarter, Patreon) is access to private community events with content creators – portfolio reviews for artists, exclusive access to IM forums with the content creators. Spotify Groups would allow for a lightweight way to organize private once-off events – for example, a musician could create a private room for people who donate €20 or more, send the link to those people, join the chat and interact with their community.

In terms of business risk, there’s a popular Discord bot that can be commanded to scrape YouTube and play it in browser. Combined with Discord’s forum-like functionality, that can create a similar experience to this product.

There are a couple of key differences: Firstly, rooms are non-permanent, rather than the permanent nature of Discord servers; its transient nature could be useful for listening parties and once-off social occasions that don’t require a permanent groupchat. Secondly, Discord’s Rhythm bot is a rough scraper that can come up with the wrong version of a song if they share the same title. Spotify’s search function is more specific.

# Key Stakeholders

|  |  |
| --- | --- |
| **Major Stakeholder** | **Notes** |
| **Jason Quinlan** | Product owner, will be surveying progress and testing eventual product. |
| **Programming team** | Will serve as a good example of a group project on GitHub for future employment, as well as an educational experience. |
| **Possible users** | Might be useful if people want to use it – a Python/React program could end up more lightweight than Discord, for example. |

# Project Objectives

|  |  |  |
| --- | --- | --- |
|  | **Target** | **Tolerance** |
| **Scope** | New application in Django/React allowing for shared Spotify listening and possible direct messaging. Documentation describing code and process. | Theorised possible additions to scope include the aforementioned messages option, emoji reacts and simple games, but are dependent on time. |
| **Time** | 7/8 weeks | Not much, as placement will be starting. |
| **Cost** | Free in currency, ~20 hours per week | If project is finished early cost in time will be smaller, if it requires more time to complete cost will be greater. |
| **Quality** | Good, hopefully! | As Voltaire said, “Perfect is the enemy of the good” – having a finished, functioning product is more important than certain factors i.e. beautiful UI. |
| **Risks** | Not getting project finished in time, project sprawl with added features, project being more complex than expected. | Sprawl can be added if basic features are completed, and can be rolled back to previous versions if necessary. Complexity is not necessarily bad, as long as project team has time to adjust. |
| **Benefits** | Small and simple application at end, allowing for connected experiences New knowledge of languages, team skills and apps such as Slack | Benefits speak for themselves – not much of a tolerance margin here. |

# Project Management Team

|  |  |  |
| --- | --- | --- |
| **Role** | **Reports to** | **Appointee** |
| **Sebastian Racki (Spotify API Integration)** | N/A | Jason Quinlan |
| **Cathal Donovan O’Neill (Documentation)** | N/A | Jason Quinlan |
| **Naina Nair (Implementing visual representation of frontend code, error handling and redirecting)** | N/A | Jason Quinlan |
| **Allan Barry (Implementing visual representation of frontend code, error handling and redirecting)** | N/A | Jason Quinlan |

**- - - - PRINCE2® Templates - - - -**

A guide for this template, it’s latest version, and all other templates are available at [mplaza.pm/templates](https://mplaza.pm/templates)

Also, you may be interested in using our PRINCE2 eLearning Courses available at [mplaza.pm](https://mplaza.pm/)

Copyright 2018, Management Plaza

You can use this document for free in your projects and for your personal purposes. Redistributing this document or using it for training requires permission from Management Plaza.

This document is based on AXELOS PRINCE2® material. Reproduced under licence from AXELOS. All rights reserved.