Project Brief

Black = old brief, green = 0.5 additions

Document Information

Project name:	Spotify Groups
Date:	Date of start: 29/1/2021 Date of current writing: 26/2/2021
Author:	Group Project Team 11
Owner	Prof. Jason Quinlan
Document code:	
Version:	0.5

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

Project Brief

Black = old brief, green = 0.5 additions

Notes

Definition

Background:	Spotify is an integral part of life for many young students. Whether it is used while doing a computer lab, walking down the street or at a 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. • Familiarity with including and calling to outside APIs (i.e. Spotify) in a web application using HTTP calls. • Create an app that is simple, fast and easy to use. • Fulfill all of the must-haves in our MoSCoW diagram and as many should-haves and could-haves as possible, recording it as we go. • Get deliverables in on time. • Gain experience with collaborative lines of communication while adapting to the online world, through Slack, regular Teams meetings and direct messages.
Constraints and Assumptions:	• **Constraints:** • Time limit of 7/8 weeks • Manpower and knowledge of university students • Malleability of Spotify Web API • **Assumptions:** • Web app will not be beyond our abilities to make in the given time • No team members will suffer major hardware crashes over the course of development, rendering them unable to participate in the project. • Technology required will stay within the limits of React, Django and Spotify's API, and new languages will not be needed

Project Brief

Black = old brief, green = 0.5 additions

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. • Regular to-do lists from scrums and bug reports are put up and discussed on the Slack. Discussions also include where users are related to each other to allow for better documentation in the brief. • Programming team works on their separate tasks as defined below, and push/pull/merge their code with their GitHub accounts. This is noted on the Slack, which notifies the whole team when someone has merged their code.
Project Product Description:	• As of 0.5: • UI that can create a 'room' and have other users on difference machines join or leave rooms. • Once the host creates a room, they start Spotify on their laptop app or the web browser Spotify and start streaming. • All users in room will be able to synchronously hear the same music stream that the host is playing. Users can vote to skip a song, and if it reaches a certain number set by the host the song is skipped. • On creating a room, the host sets whether or not guests can pause the group stream and how many votes the song takes to skip. • Possible groupchat message function is being worked on at the moment.

Project Brief

Black = old brief, green = 0.5 additions

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.

One business benefit would be for private listening parties. A popular pledge tier for 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 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.

Spotify has also released a 'Spotify Group Sessions' function in beta. These allow up to six people to join a synchronous group, which shares a queue. Like theirs, our Spotify Groups will be available for anyone with Spotify Premium and a login code. However, there are two differences: Firstly, Spotify's app allows any participant to control the queue, while ours requires a number of votes to skip (set by host on room creation). Secondly, Spotify's app requires either a QR code scan or a Spotify link; our requires a simple four/six letter code to enter a room. It is also worth noting that Spotify preceding us does not mean it will be more successful; music label giants Universal Music Group and Sony Music released an online music store named PressPlay two years before iTunes, but it ended up forgotten.

Project Brief

Black = old brief, green = 0.5 additions

Key Stakeholders

Major Stakeholder	Notes
Jason Quinlan	Product owner, will be surveying progress and testing eventual product. Sets schedule for weekly milestones, such as regular project brief updates and inclusions (i.e. MoSCoW, Gantt chart) and provides project management resources such as UCC Entrepreneurship Seminars, educational YouTube videos and academic papers.
Programming team	Sebastian, Naina, Cathal and Allan, bringing our skill sets together to accomplish the goal of launching Spotify Groups. Will serve as a good example of a group project on GitHub for future employment, as well as an educational experience, with Sebastian and Allan improving their pre-existing knowledge of Django and React respectively, while Naina and Cathal get to come to grips with the new programming language and technologies.
Possible users	• Online gamers – would allow Spotify to be shared between people in an online gaming session, providing a shared experience over the internet in a time where people can't sit on the same couch as each other. • Families and friends – There are few art forms as intimate and personal as music, and like shared playlists, listening rooms would allow families and friends divided by oceans to experience the same album at once. • Music fans and music creators – Could be used for album launch parties, as described in business case. Fans could schedule a time to listen to an album as a group, share the link with each other, and go through it together at the same time.

Project Brief

Black = old brief, green = 0.5 additions

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	Fully functional with minimal bugs. Sound quality should be the same as playing through Spotify's Web Player, and entry into rooms should be frictionless with six-letter code.	Having a finished, functioning product is more important than certain factors i.e. high-quality graphic design for 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 Brief

Black = old brief, green = 0.5 additions

Project Objectives – MoSCoW Prioritisation

Must have:	○ Ability for users to create and join rooms with each other using a password ✓ Spotify API functioning, allowing users to log in with their accounts, play/pause it and listen to it with each other synchronously ○ Users should be able to add new songs to queue ✓ Django views and URLs linked to React JS functionality ○ Updated README describing installation and running of app ✓ The ability to choose whether guests should be able to skip songs or not ✓ Spotify API-React connection allowing a logged-in user's name to be displayed ✓ Ability for anonymous users to set own nickname on entering room ✓ List of users currently in room ✓ Ability to exit room without leaving webpage
Should have:	○ An instant messenger function for each room, with each user giving a nickname on entry ○ The ability to choose a set number of votes on room entry, have users vote on skipping a song, then have that song skipped ○ Ability for host to update room settings
Could have:	○ Web hosting on the UCC CS department servers ○ Web server coded with Heroku or Microsoft Azure ○ Emojis/simple games ○ Ability to change nickname
Won't have:	• Integration with sending images • Mobile website • Social media integration, i.e. sharing what song you're listening to on Facebook/Twitter • Ability to cast audio over to other sources (i.e. from laptop to phone with Spotify app) • Integration with non-Spotify sources of audio on the internet, i.e. Bandcamp/YouTube

Project Brief

Black = old brief, green = 0.5 additions

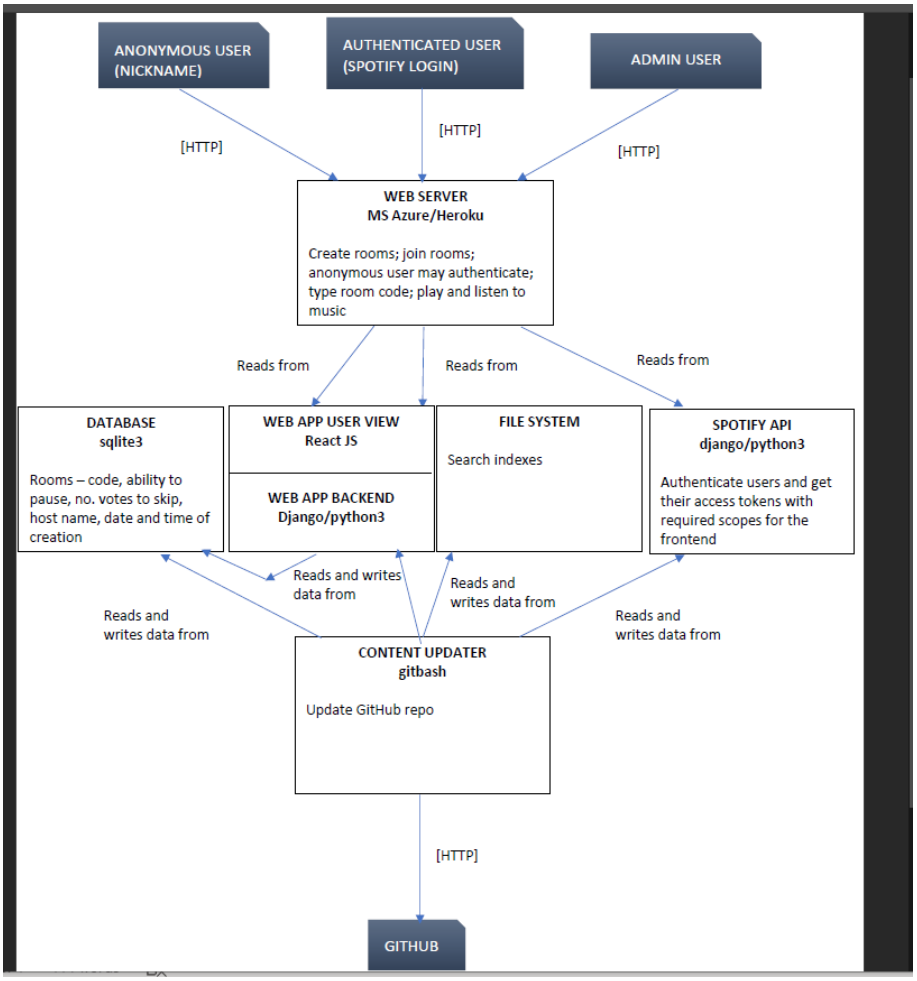
Project Management Team

Role	Reports to	Appointee
Setting up chat	Jason Quinlan	Sebastian Racki
Documentation, bug fixing	Jason Quinlan	Cathal Donovan O'Neill
Implementing visual representation of frontend code, error handling and redirecting	Jason Quinlan	Naina Nair
Implementing visual representation of frontend code, error handling and redirecting	Jason Quinlan	Allan Barry
NO ROLE	Jason Quinlan	Bernard Grabarczyk

Project Brief

Black = old brief, green = 0.5 additions

Architecture Diagram



Architecture as of halfway through semester

Previous skills of members of the project team led to us choosing to use ReactJS and Django as our frontend and backend language respectively. Likewise, SQLLite was the familiar SQL variant for the member with pre-existing Django experience, and was thus chosen. We needed Django-database interaction to hold room information, and the templates for our models. The Spotify API required using Python, as it involved dealing with Django's HTTP abilities and cookie-setting to login to Spotify and gain access to Spotify's library of songs. The app currently reads from the user's file system to load the HTML/CSS and React for the site. GitHub was set by the product owner as the content updater of choice.

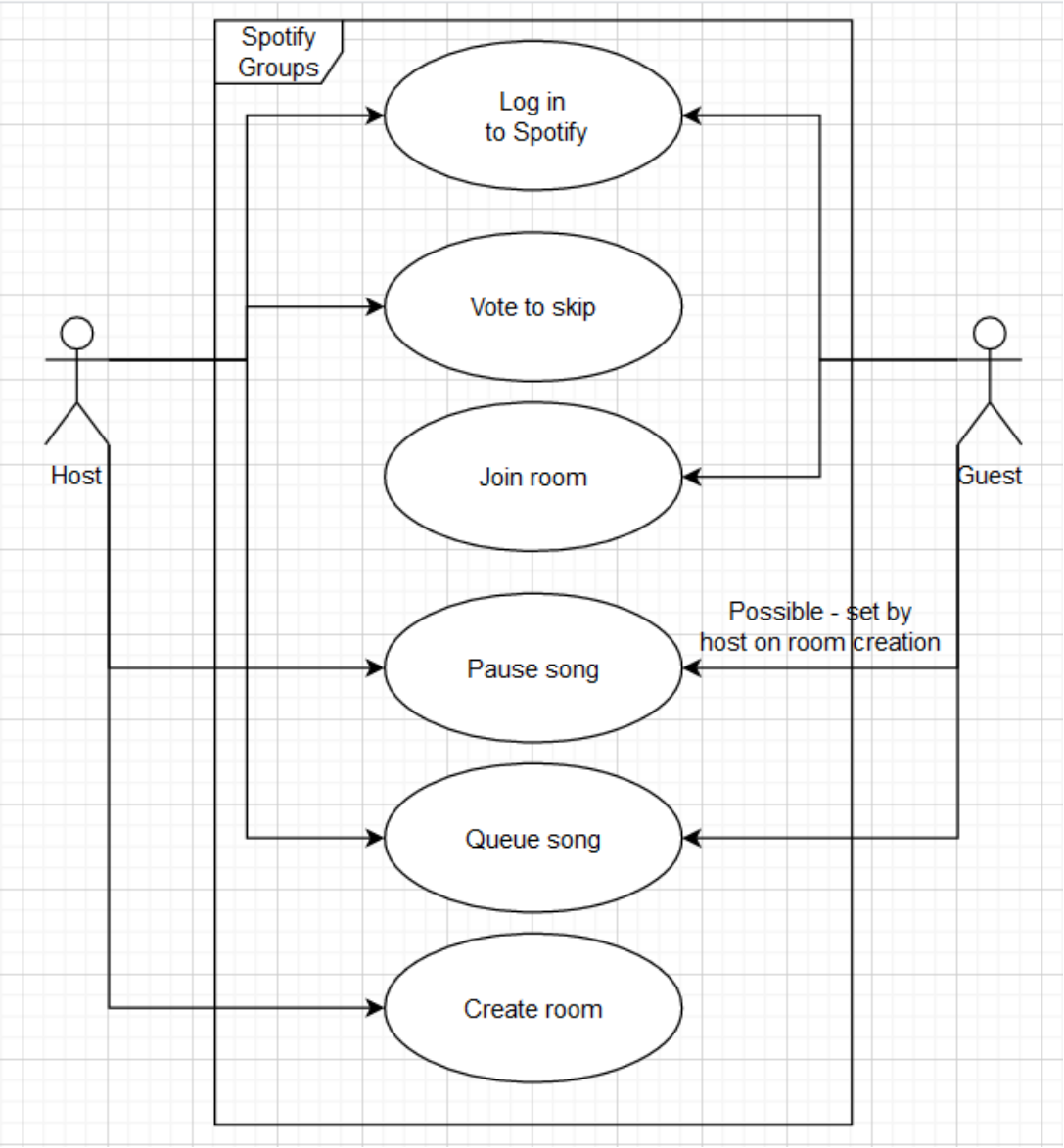
Overall, the architecture of this project grew naturally from previous experience and the requirements of the project.

(It is also worth noting that at the current time a web server has not been implemented – LAN functionality has to be implemented first.)

Project Brief

Black = old brief, green = 0.5 additions

Use Case Diagram



Use case diagram for Spotify Groups as it stands, describing what we want the app to do.

Project Brief

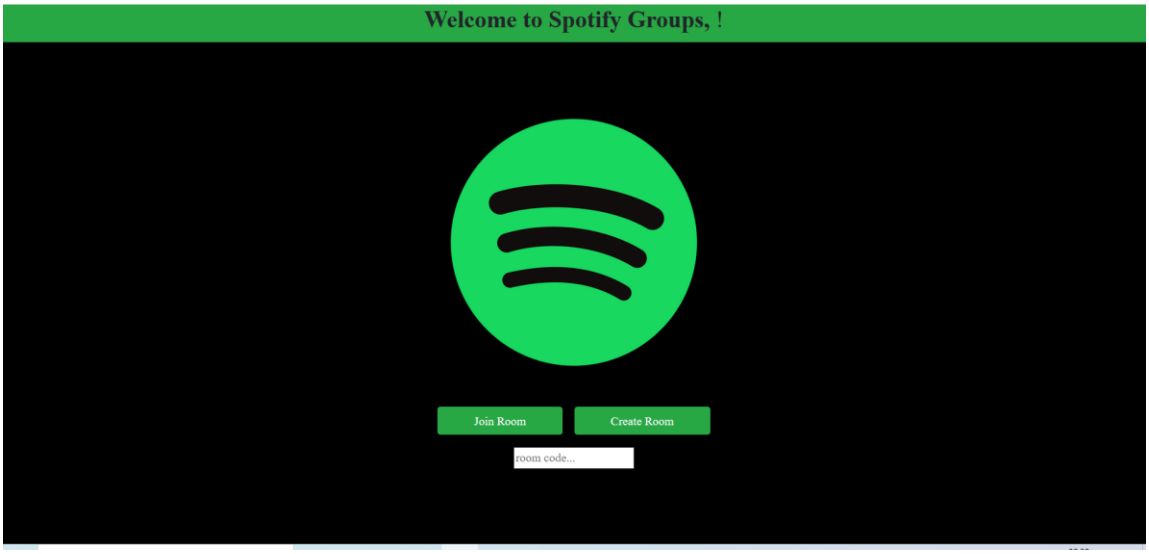
Black = old brief, green = 0.5 additions

Project updates

Week 3:

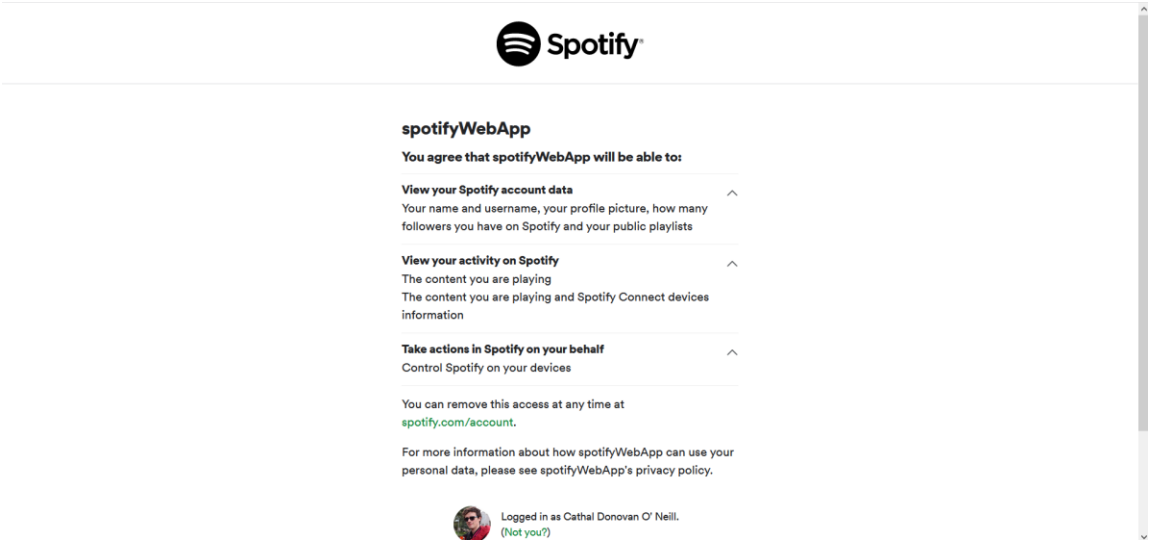
Current screens:

- Home page:



Current homepage

- Click create or join room and be prompted for Spotify logins:

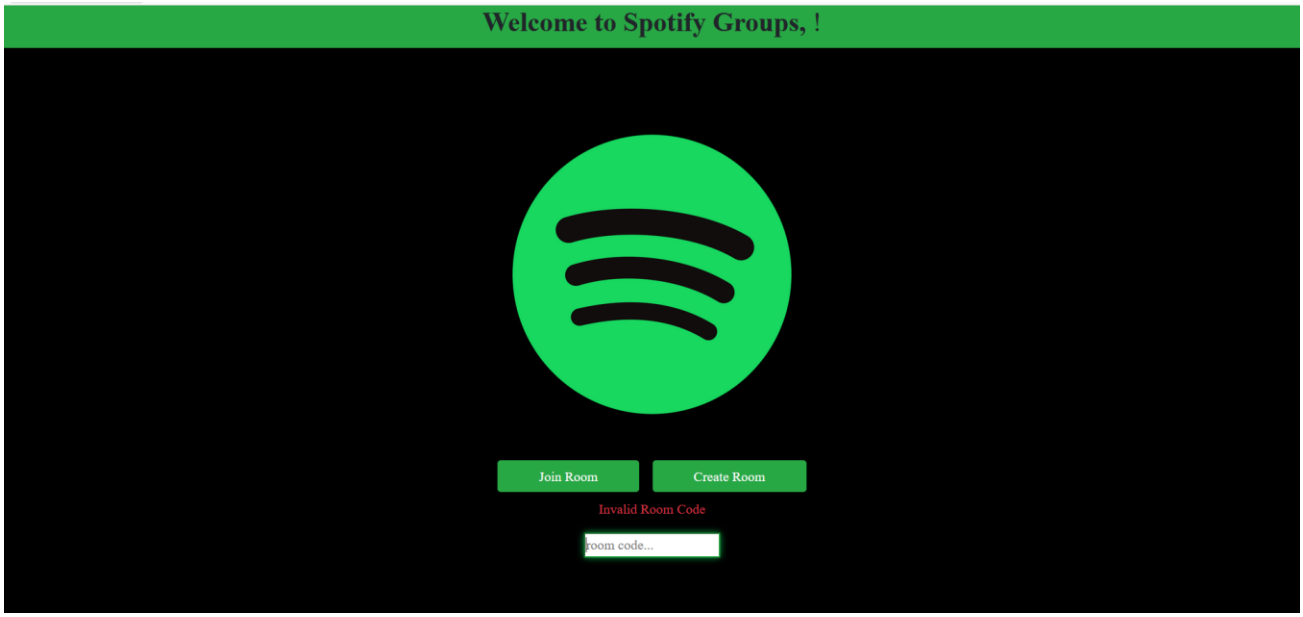


Redirected page to log into Spotify

Project Brief

Black = old brief, green = 0.5 additions

- Clicking ‘join room’ requires a valid code:



‘Invalid Room Code’ appears when bad code is input

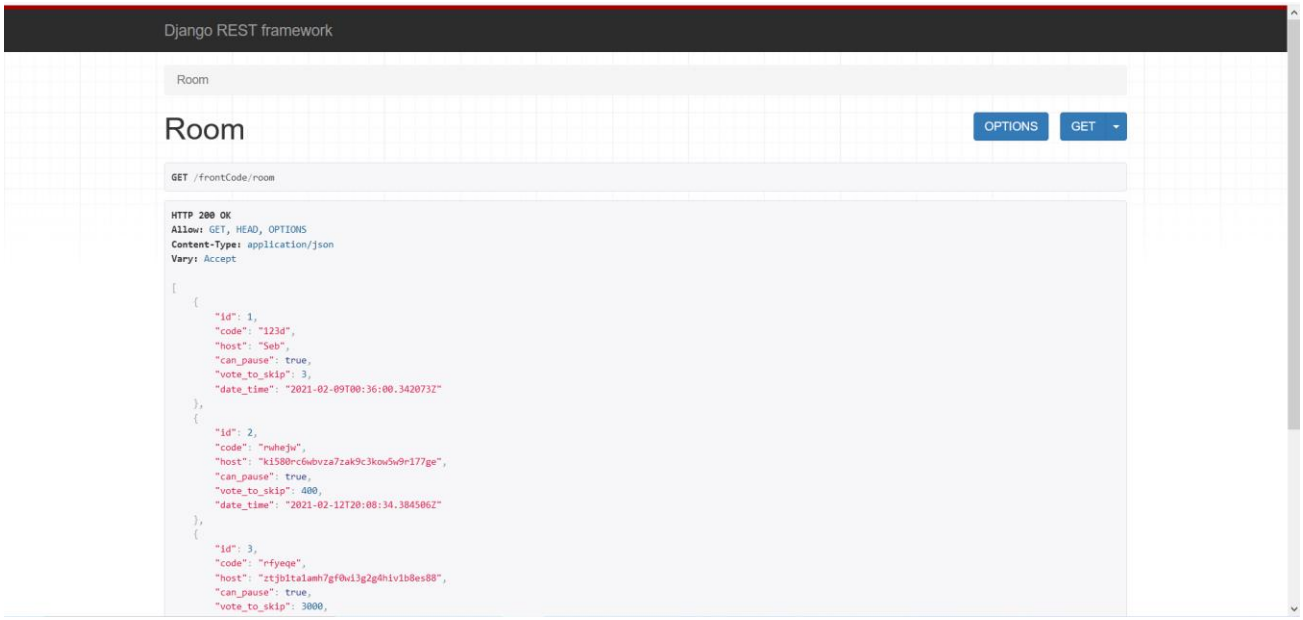
- Placeholder for room in ReactJS (currently redirected to by Join/Create Room):

Placeholder for current rooms

Project Brief

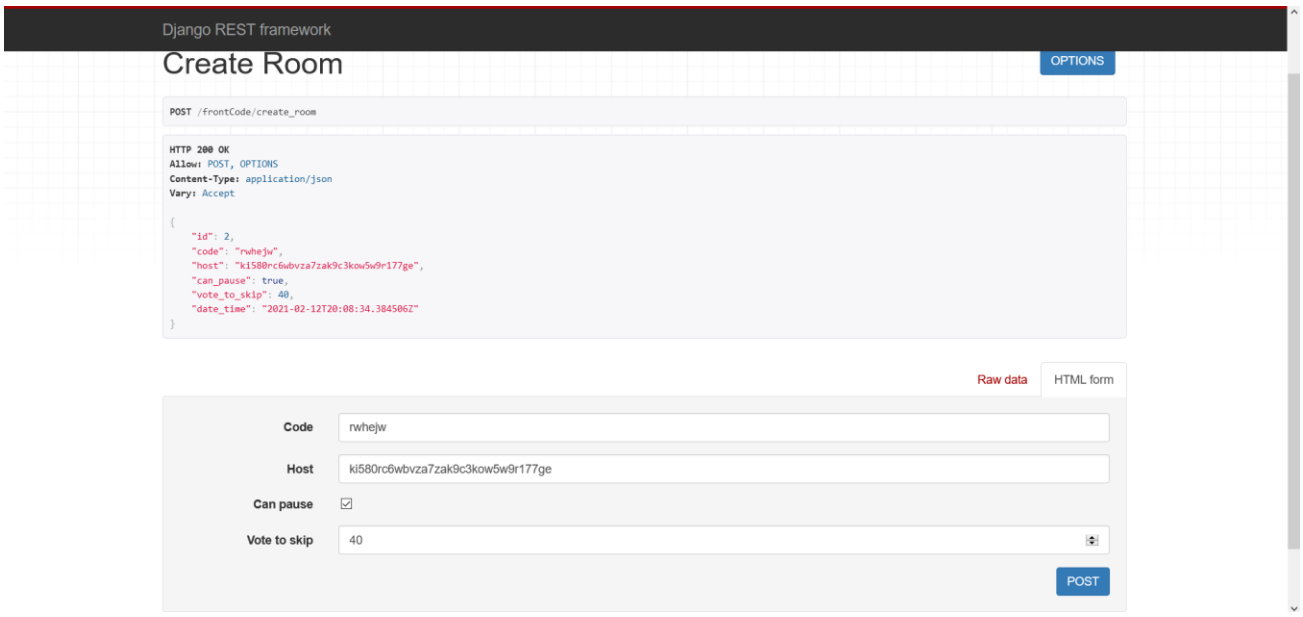
Black = old brief, green = 0.5 additions

- Current backend for collection of rooms:



The backend view of all rooms currently active in system

- Current backend for creating a room:



Current backend view of what goes on when a room is created

Project Brief

Black = old brief, green = 0.5 additions

Challenges and solutions:

- Allan:
    - Challenge: Finding a way to properly load the contents of another page when a button is clicked. Reloading a page is easy with the use of link elements, but had to make sure that buttons could call functions and perform other actions before deciding whether to reload the contents of the page. Links could not do this, so another way had to be found that did not lose out on the strengths of dynamically reloading pages.
        - Solution: Use a React Redirect function that would be called repeatedly when a page is loaded, but unless certain criteria was met would not do anything. Upon the click of a button, the function call in the page would be replaced with a <Redirect /> element changing the contents to that of another JavaScript file.
    - Challenge: Loss of CSS after the frontend was integrated with the backend.
        - Solution: Re-import the Bootstrap CSS file into the index.js file. This necessitated reinstallation of several other modules once complete.
- Naina:
    - Challenge: Unfamiliarity with Django and React
        - Solution: Doing mini-courses on them, becoming familiar enough to begin coding
    - Challenge: Redirecting to and from Spotify authentication/login page upon clicking join/create room
        - Solution: Changing some of the link tags, editing backend code, adding code to frontend, adding the redirect URI to the Spotify project's page on the Spotify Developer's website
- Cathal:
    - Challenge: Updating project brief. Received feedback from project owner and peer review requesting more detail on continuous project development along with better layout of project management team, use/sequence diagrams and information on a new competitor, Spotify's proprietary group sessions.
        - Solution: Kept track of challenges and solutions throughout the week to better describe them in brief. Analysed Spotify Group Sessions to compare to our own, put together use/sequence diagrams and implemented layout/word choice feedback from peer review.
    - Challenge: Unfamiliarity with Django and React
        - Solution: Reading resources in the Slack chat and reviewing earlier weeks allowed me to gain better knowledge of the language, along with analysing the current code. Now bug fixing Django.
    - Challenge: Fixing Create Room function. The Django backend was not displaying in browser/allowing POST requests/HTML-format input.

        - Solution: Added code to the urls.py to allow access to the create room function. Edited the post request function to fix variable-Room attributes, allow POST requests and fixed the JSON-->Django serialiser.
- Sebastian:
    - Challenge: Resolving merge conflicts in git.
        - Solution: Delete branch and reimplement separate segments of code until merge conflicts resolved
    - Challenge: Splitting up the work between the team, making sure tasks distributed equitably.
        - Solution: Assigning tasks to teammates based on people's strengths, interests and weaknesses.

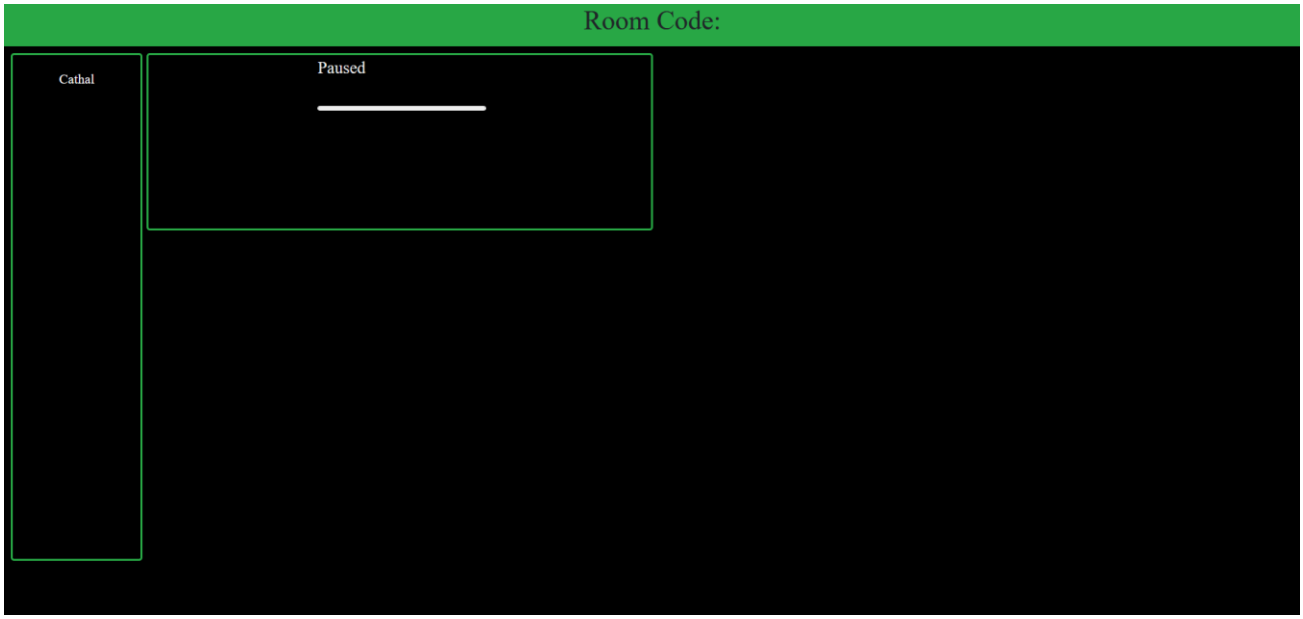
Project Brief

Black = old brief, green = 0.5 additions

Week 4:

Screens:

- Updated room page with nicknames/unfinished player:



Left-hand box is for usernames, right-hand box is for Spotify player

- Updated backend pages (No current visual display):

```
1. admin/
2. frontCode/ room
3. frontCode/ createRoom
4. frontCode/ getRoom
5. frontCode/ leaveRoom
6. frontCode/ joinRoom
7. frontCode/ user
8. frontCode/ updateRoom
```

List of current Django URLs from backend POV

Project Brief

Black = old brief, green = 0.5 additions

- Updated Spotify pages (No current display):

```
spotify/ get-auth-url
spotify/ redirect
spotify/ authenticated
spotify/ currentSong
spotify/ play
spotify/ pause
spotify/ skip
```

List of current Django Spotify API URLs

- React rendering of join/create lobbies are not yet completed, so black screen:

Incomplete join/create room page

Rest are as week 3.

Project Brief

Black = old brief, green = 0.5 additions

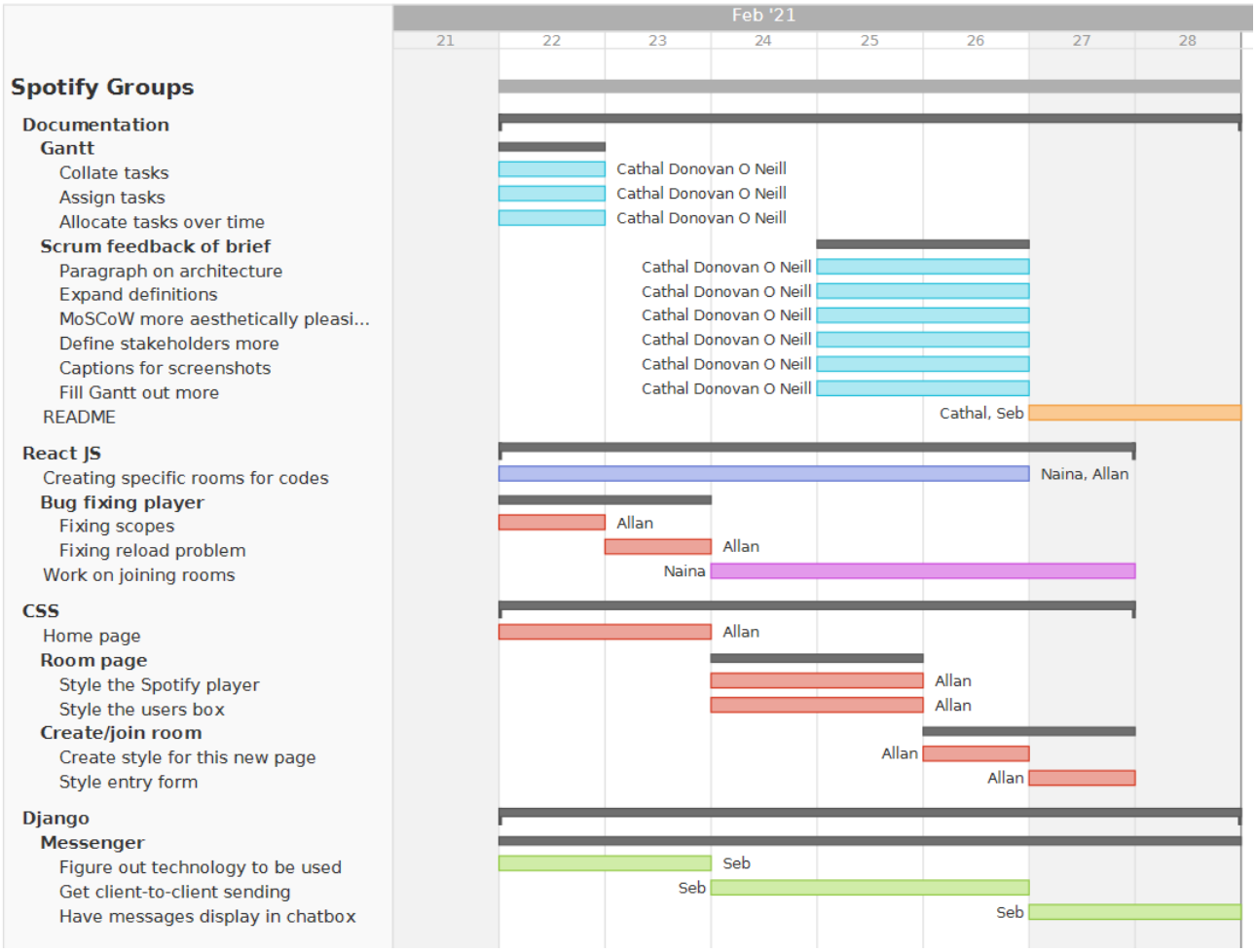
Challenges and solutions:

- Cathal
    - Challenge: Update the document with response to feedback given: Fill out the MoSCoW form more, add an architecture diagram combining all team's individual architecture diagram, update with screens, challenges and solutions, add Gantt chart.
        - Solution: Updated documents, organized meetings to discuss diagrams and combined them into one, organized meeting to discuss Gantt chart for week 5.
    - Challenge: Continued bug fixing in the Django views and URLs.
        - Solution: Fixed bugs, mostly focusing on syntax and consistent variable names. Finished looking into Rooms errors and mostly finished in Spotify API, intending to move to active development next week.
- Sebastian
    - Challenge: Finish planned Django views and URLs for updating room settings and leaving rooms.
        - Solution: Completed; now working on resolving some code conflicts and merging.
- Naina
    - Challenge: Trying to get the website to redirect to the 'join room' page when the 'join' button is pressed.
    - Challenge: Similarly, get the create button to redirect to the create room page when it is pressed before it takes the user to the actual room.
        - Solution for both: First had to code the actual 'create room' and 'join room' pages, after which had to resolve some slight inconsistencies with the frontend and backend code. Then, had to change some things in the homepage to make sure the buttons redirected to the right pages: Initially the buttons went directly to the room, so they had to be redirected to their separate pages. The 'enter code' text box was moved into the join room lobby rather than the home page.
- Allan
    - Challenge: Getting the Player working was a bigger challenge than expected. Accessing the song data was fairly simple but playback of said music could not be achieved without specific permissions/scopes used to get the access token. Up until now, for development

purposes, I had been using hardcoded access tokens obtained from Spotify's website, however one of the required scopes for the Player was not available to be set when requesting a token like this.

- Ongoing work: Now is working on obtaining the access token for the current session through working with both the backend and some complex React functionality.

Gantt Chart for week 5



Planned schedule for the week ahead

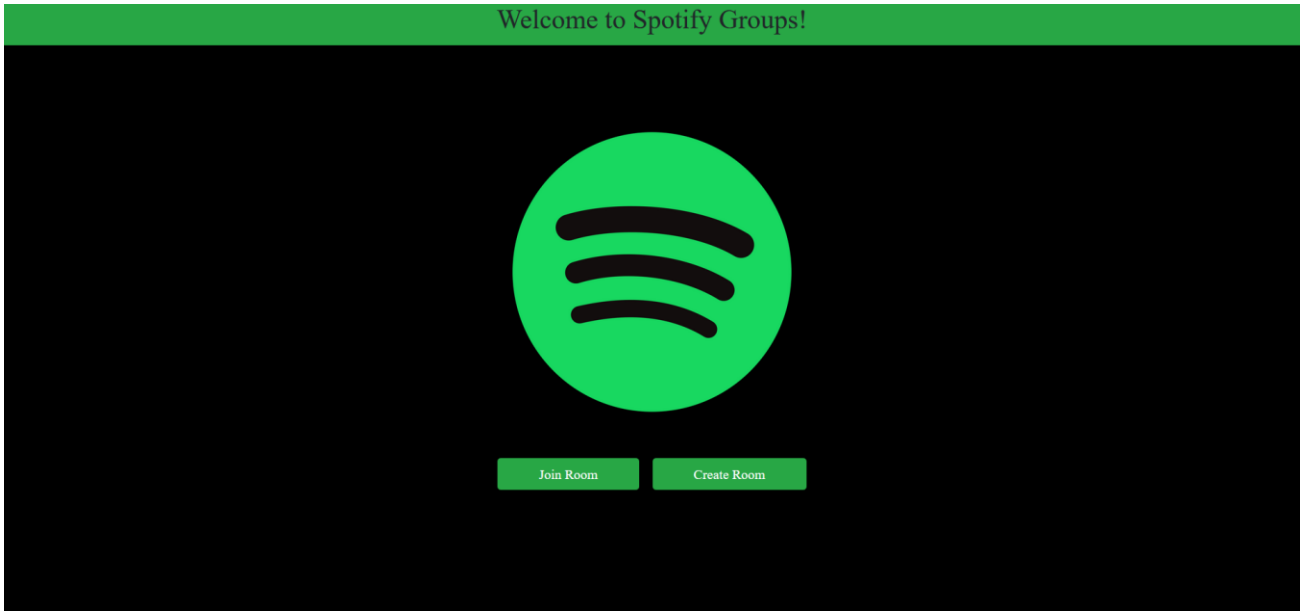
Project Brief

Black = old brief, green = 0.5 additions

Week 5:

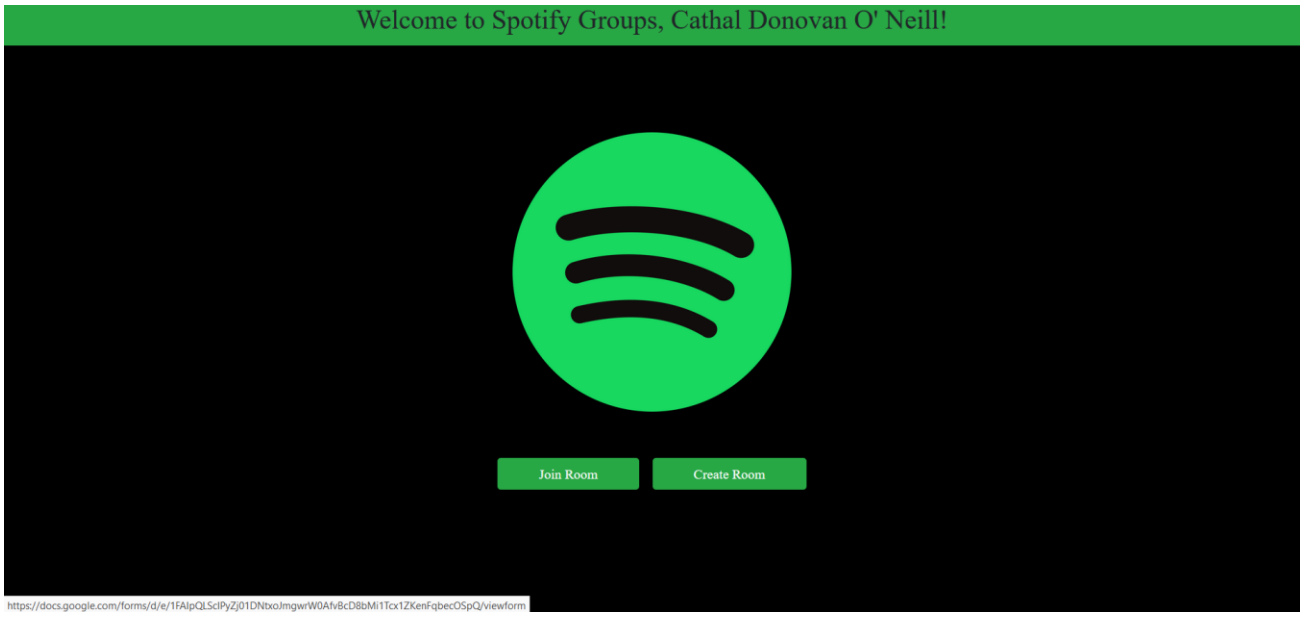
Screens:

- Home page now displays logged in user's name:
    - Before login:



New home page

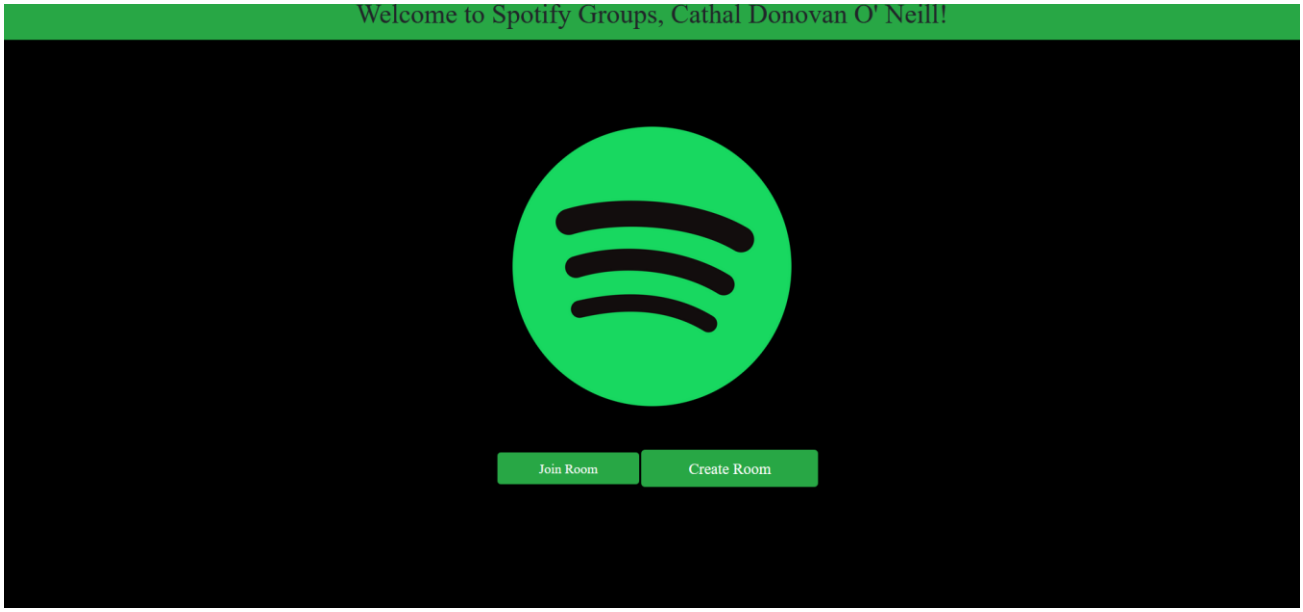
- After:



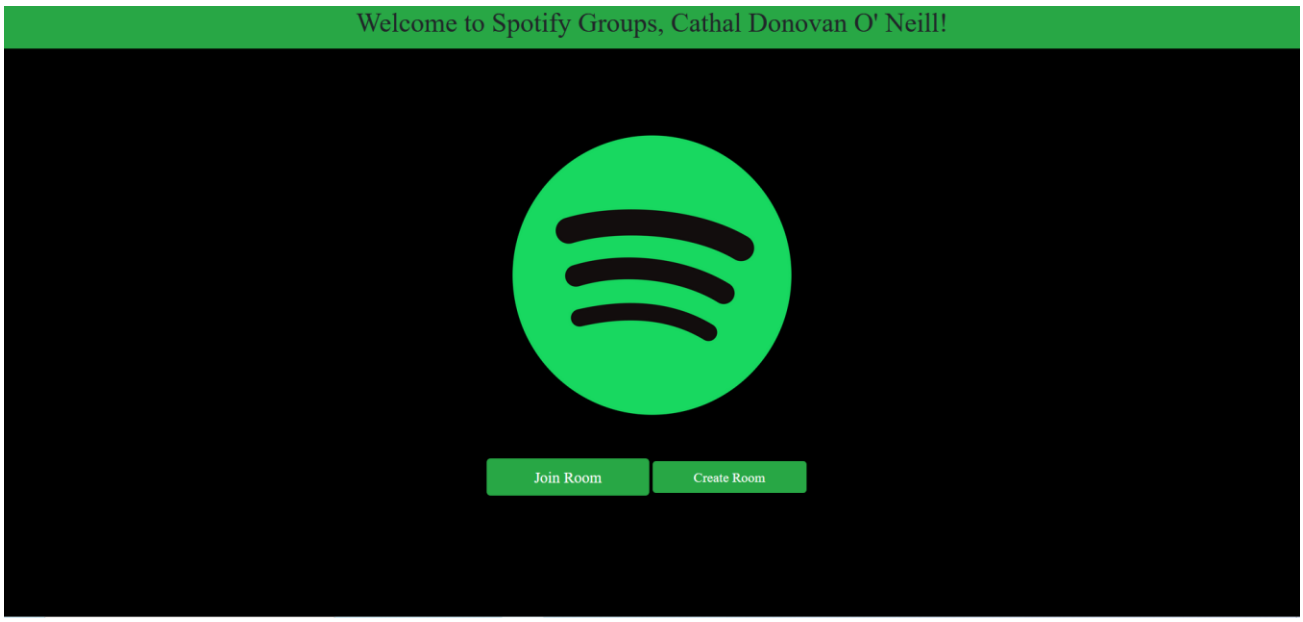
Project Brief

Black = old brief, green = 0.5 additions

- Buttons enlarge when hovered over:



'Create Room' button enlarges when hovered over

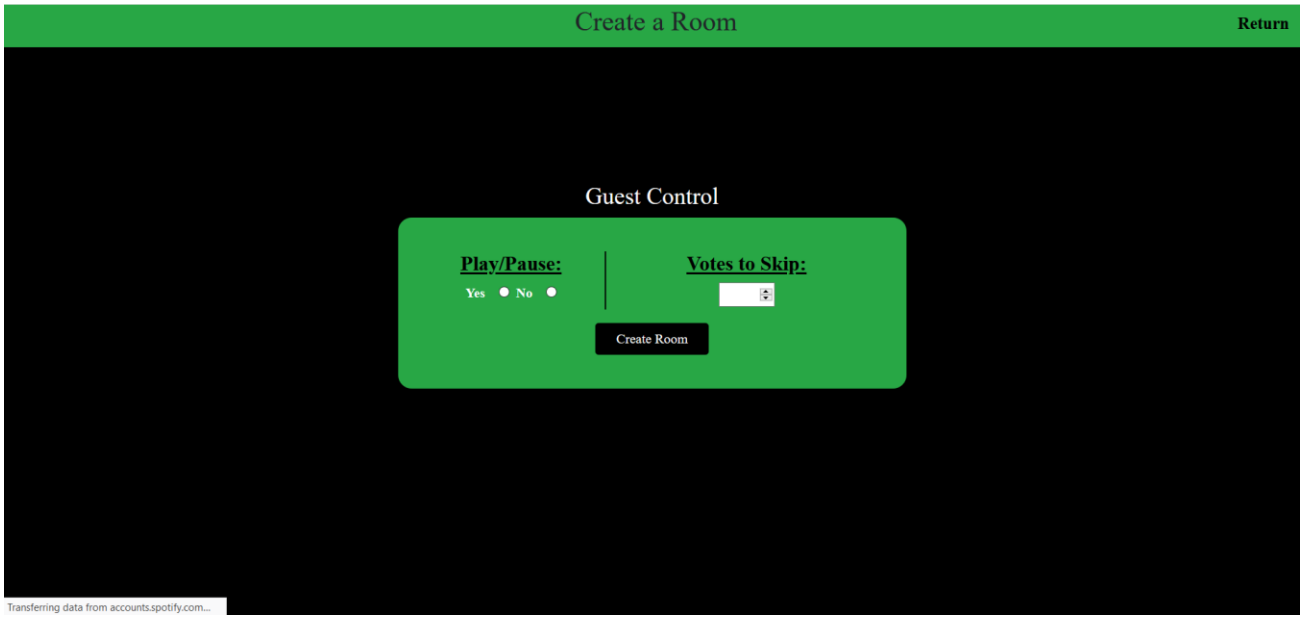


'Join room' button enlarges when hovered over

Project Brief

Black = old brief, green = 0.5 additions

- New ‘Create Room’ CSS (Return page goes back to homepage):



New Create Room layout

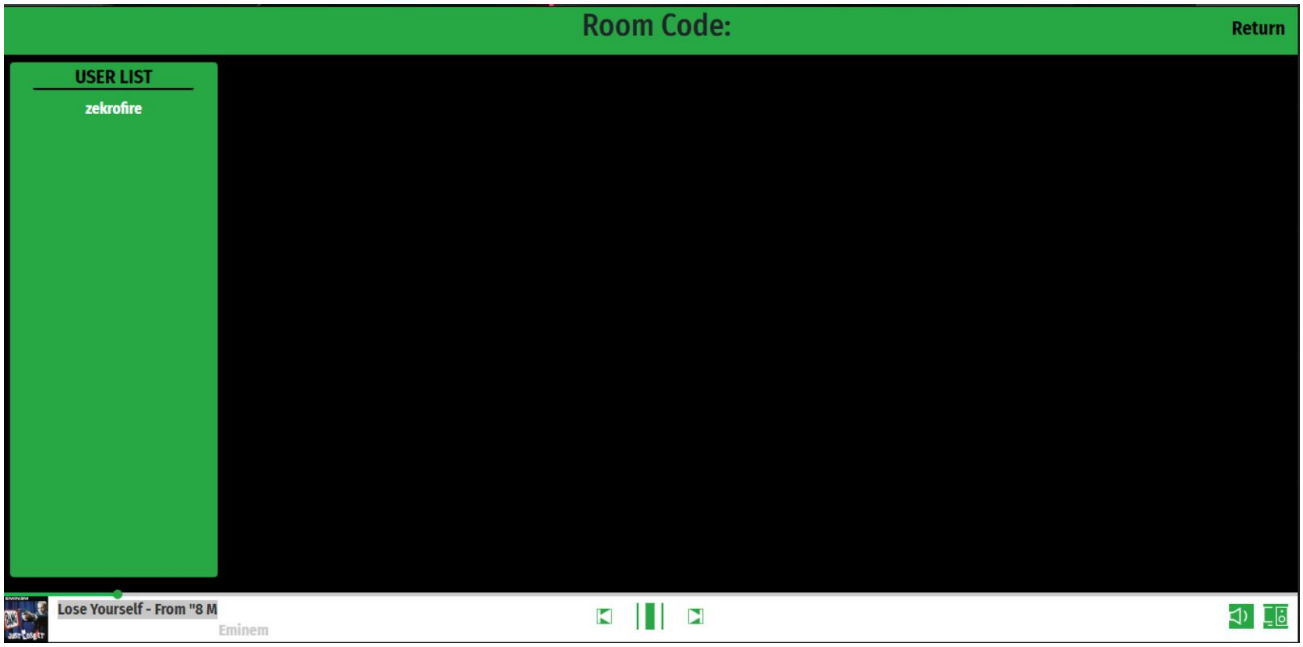
- New CSS for Room without Spotify currently running:

Room with no Spotify running at the moment

Project Brief

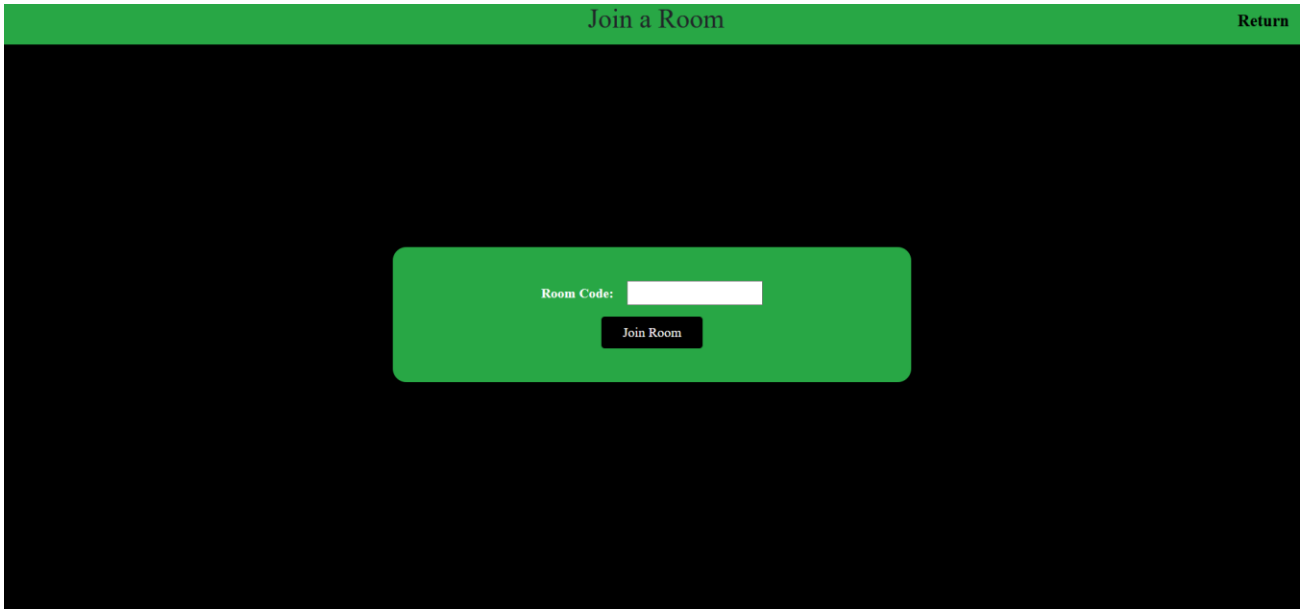
Black = old brief, green = 0.5 additions

- Spotify now runs in Room:



Room with Spotify running

- CSS for joining a room:



New Join Room page

Project Brief

Black = old brief, green = 0.5 additions

Challenges and solutions:

- Cathal:
  - Challenge: Preparing brief document for assignment.
    - Solution: Expanded tasks for updated Gantt chart, added explanatory captions to all screenshots, Modified MoSCoW for aesthetic purposes and to keep track of goals, expanded 'Stakeholders' section, expanded explanation of system architecture, expanded introductory definition by a page
  - Challenge: Bug fixes
    - Ongoing: Moved to back-burner after difficulties with merging and prioritizing the brief for the deliverable deadline. Collated list of a few bugs on paper and will move onto them over the weekend.
- Sebastian:
  - Challenge: Moved onto working on the proposed messenger function. Needed to decide on technology, and get messages sending from client to client and showing up in a chat box.
    - Solution: Using Django channels, has successfully managed to get messages sending between clients on the backend.
    - Ongoing work: Chat box on the frontend not currently outputting received messages to user. Will be working on it over weekend.
- Naina:
  - Ongoing work: Focused on fixing bugs this week, will move onto create/join rooms next week.
- Allan:
  - Challenge: Completed the ongoing work from last week.
    - Solution: Spotify player now works with playback and scopes have been fixed with web player.
  - Challenge: Modifying CSS for old and new rooms
    - Solution: CSS has been modified heavily, particularly to allow for the new player, as can be seen in the above slides.

Project Brief

Black = old brief, green = 0.5 additions