Web Application Development

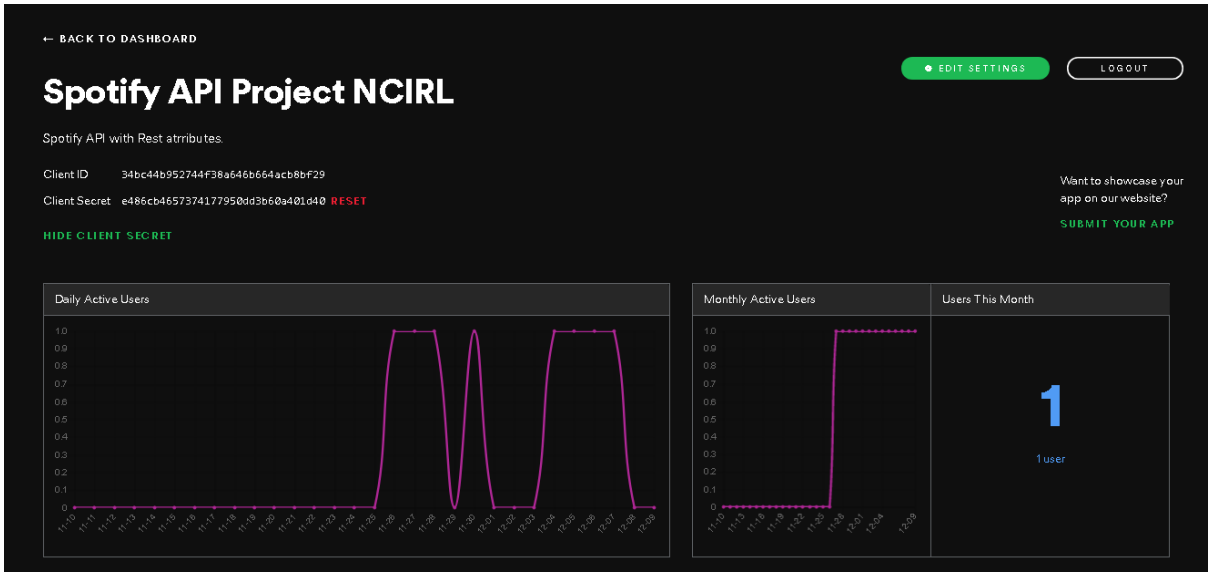
Project Codename **“Spotify API”**

Cian Larkin	**X17453136**
Carl Flynn	**X17347726**
Matthew Byrne	**X17138744**

CodeAnywhere Link

https://codeanywhere.com/s/l/ct4SiQeA8C8Pl2u1Upj6dSfFV34v2qIi63K9jn2ENcefsiinUGUMj6E5cfmhFADp

Contents

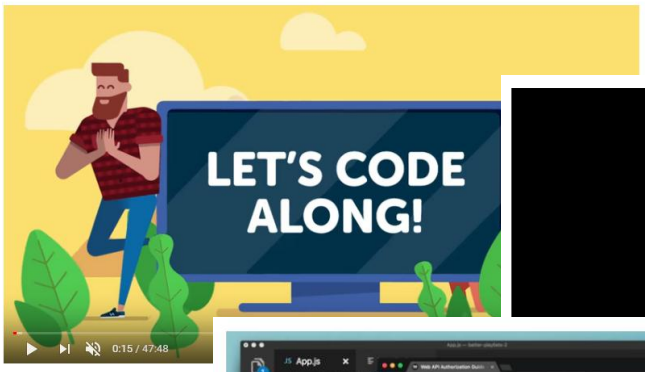


1. Introduction

For the development of our project with Web Applications we came together as a team combined, running through a various number of ideas, that we could potentially implement for our project. We came to the idea that could unlock the idea of "Spotify API" and how the online server can potentially retrieve information and data onto a local host page throwing back details of the tracks on the screen. We wanted to use a database of music with potentially XML or an online database, and have it link together to form a chain. This introduction lists the final ideas of where we went, but the overall report will go into more details on how we developed ideas at the very start.

2. Gathering Information

At first when we proposed our idea for Spotify we had a number of concepts that we wanted to do but insisted that we start out the project very bare bones, and then build on top of that information to form a concrete model. We did a number of research material online on how Spotify API is implemented on various other online and web page applications. We even came across a number of tutorials telling you how to set up your very first application. From there we formed an idea of using a potential API style database that then would link with a click of a button, information regarding tracks data, artist, year etc. We also ideas of how to play the track at the beginning of the process but ultimately decided that retrieving the basic information (artwork, artist details, now playing), would prove more effective as playing tracks will bring a massive amount of JavaScript coding even with frameworks in place. This would have increased the work load dramatically and hinder the project.



Let's Code Along - Spotify Playlis

15,836 views

Pulling data from the Spotify API - #13 React JS prototyping

3. Tools Required

Using the Spotify dashboard and going through the start-up process and understanding to call a local host was crucial on set up. We set up a basic "Hello World" transmission and then moved onto the authorization code on their website which helps you get an application up and running.

Json code is a required information to how Spotify retrieves its information online and through local host. This essentially holds the metadata for each track and artist etc.

JavaScript to run the App.js in the Rest App is essential in the operation of Spotify as it in turn links HTML and CSS together also. From there frameworks listed below, allows the program to boot.

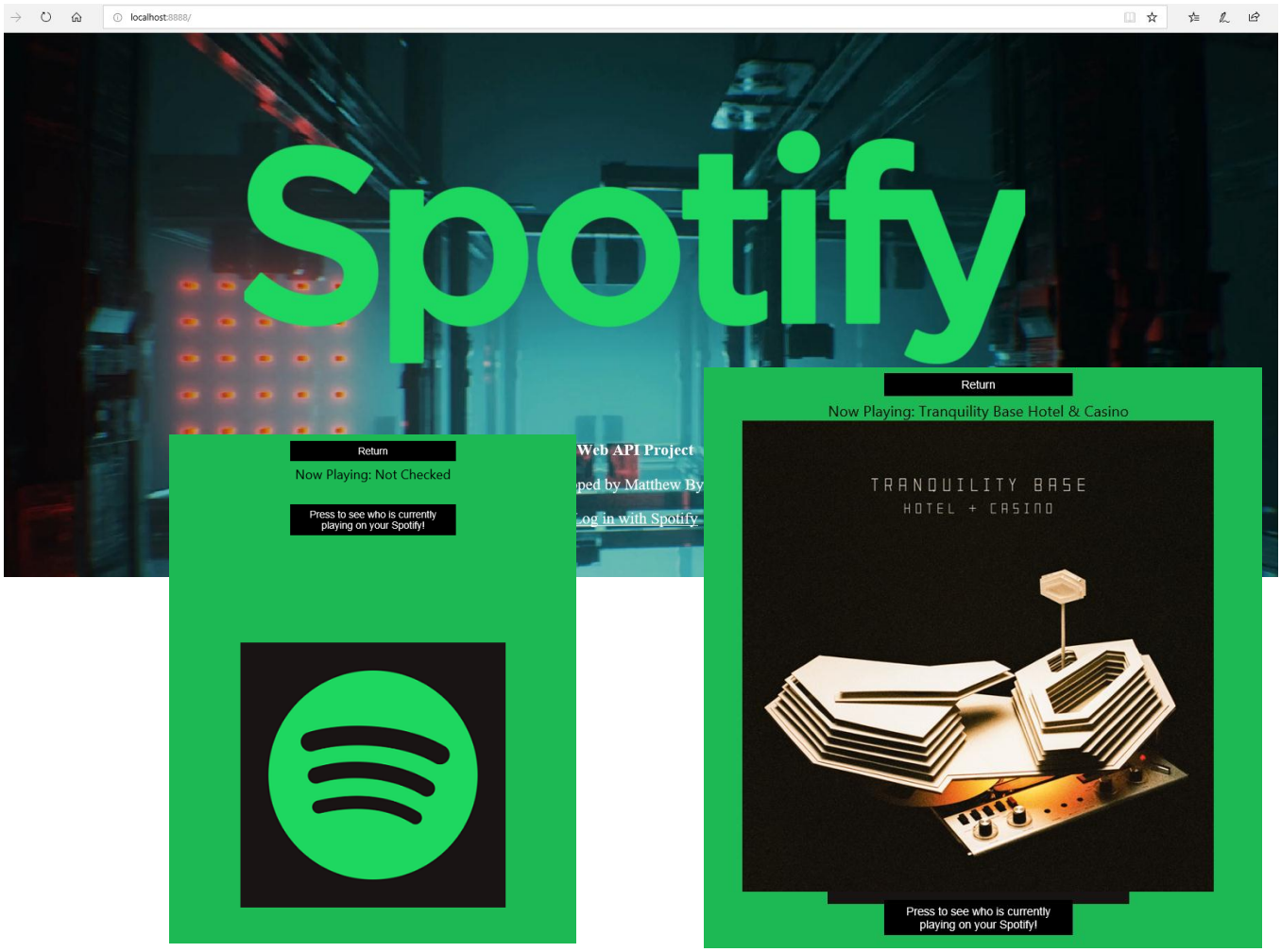
A framework for the creation of a Spotify application using **React** was used and installed on top of additional side server scripting with Node.js. The authorization code for Spotify is dedicated on one page with local host 8888 while the main Spotify artwork site is dedicated and rerouted to local host 3000.

4. Aim / Purpose

To display user details information on a local host server (3000 and 8888), that updates information in real time while you access your very own Spotify on any device. This is obtained using your own "Client ID" and "Client Secret" codes dedicated to your Spotify.

There is also a redirect URI of 8888 included. From there included in the React activated application built, there is a piece of JavaScript written that allows to set parameters and then send user information to online database entitled "spotify-web-api-js".

This in turns returns artwork data and track name details through Spotify API.



5. Conclusion

Along with presentation slides also included with this project, we will detail what were the problems and the benefits of using side servers. React was a fantastic piece of technology that allows real-time running and real ideas of what do going forward with the project, but we also knew we had a limited amount of coding skill to make every feature in Spotify API and this disappointed us to a great extent.

It was reasons for this that we wanted to display so much more but ultimately in our coding experience could not. We feel we have dominated the side server techniques really well and by bringing in two servers was a big challenge.

The artwork and information displayed based purely on JavaScript and literally fetching the information from a database is fantastic. There is no iframes or widgets involved, and the process is using the API as the project outlined. We also managed to get simple json data up on the screen as a history of Spotify and this was executed well also, as entirely the second page consists of zero html, something we never thought we could do in a project.

We hope that we have managed to bring something different to the table for API use, and that it was never about showing a simple website but taking piece by piece the applications needed to make something entirely new. We hope you have enjoyed this insight.

6. References

https://github.com/spotify/web-api-auth-examples

https://github.com/JMPerez/spotify-web-api-js

https://www.youtube.com/watch?v=9C85o8jIgUU&t=115s

https://www.youtube.com/watch?v=prayNyuN3w0&t=1071s

https://www.youtube.com/watch?v=m3YpkqhHKdk&t=2s