21. Edward Aldeen, Kian Lynde

Spotify API

- Our application / website uses the Spotify API to integrate songs, playlists, and podcasts into our website.

- Kian's Link:https://csci331.cs.montana.edu/~h94m544/finalsite/
- Edward's Link: https://csci331.cs.montana.edu/~m52t465/Final/

- https://github.com/yakamashii2/EA-KL-331-FinalProject-Spotify

- https://docs.google.com/presentation/d/19v-W-kC9vvA3fA2XpII6OgPTMIjI4lHhxe8HZqaZcXY/edit

Creative Objective

We wanted to have a website that utilized the Spotify API in a notable fashion, whilst also showing some of the workings “under the hood” along with having music play in the website. For our in class presentation we went with a profile data grabbing typescript demonstration that would grab your email, username, etc. After accidentally creating a runaway process on the server (2603 active processes) and some other issues with the API, we decided to cut down on the scale of the project and have a modest website with <iframe> embeds instead that would show songs, a playlist, and podcasts.

Tech Summary

The Spotify profile demonstration utilized an “application” through Spotify’s developer dashboard, giving us an API key to access Spotify's API. The application also had redirect links that must be set to authorize login for any account to pull the account data. It also has two important codes called the Client ID and the Client Secret. Then using those codes we would get an access token that would last an hour, giving us access to JSON data on anyone who allowed us to connect to their profile. Instead of using NextJS, it used Vite which is very similar. We used typescript, but the majority of the website’s setup was similar to a standard NextJS website.

Our current website uses <iframe> embeds to access the Spotify API to play music within our site, display a playlist, and podcasts. Instead of using the same API tied to the application we made, it uses the iframe API to access Spotify and allows us to manipulate any <iframe> elements on our page. In order to get the <iframe> embed you must navigate to the song/playlist/podcast on spotify and click share, embed and then copy the code into the relevant part of your HTML.

Member Notes

Edward Aldeen - My contributions consisted of uploading our code to our github, helping with implementation of the original Spotify profile demonstration, half of the presentation and creating the base of our newer website. The github was created as a public repository with git-bash. The Spotify profile demonstration, as stated previously, used Vite and typescript to grab the relevant information. If I could have done anything differently I would have never touched Vite or NextJS or Javascript, and I definitely would not have accidentally uploaded a runaway process onto the course server.

Kian Lynde - My contributions consisted of looking up the API and doing their initial tutorial of creating a website that uses their API to pull user data like the username and email. This included going through Spotify’s documentation to learn what we need to make the API calls along with how to actually make them and what was returned from the calls. Afterwards, I followed their tutorial using typescript to create the website. The tutorial was pretty straight forward but there was some required troubleshooting such as fixing it from looping back to the authentication page without showing any data. I also helped with the presentation and set the order of the slides to make the presentation flow smoothly and added the podcast <iframe> to the final website.

Conclusion

Overall, the Spotify API was a fun project to work on and had a mix of easy and challenging aspects to it. An example of this would be we only had a short timeframe to learn the inner workings of the API and typescript. We also learned how to use the API to get data from spotify or embed songs, playlists, and podcasts to make the site more interactive. Some of the troubleshooting we had to do to get the sites up and working properly also taught us some things. The embeds for the songs, playlist, and the podcasts in the second website worked quite well. Things that didn't work as well were trying to get the first site hosted on the course server as there were issues with Vite, assigning the port number, and redirecting properly. If the redirects weren't exact nothing would work, so it made troubleshooting difficult. In the future we would likely use the Javascript on the course server for the website as well as using NextJS instead of Vite. This is because we had more experience with NextJS from our homework assignments. The things we would keep the same would be using the embeds with iframes as they are a lot simpler to use, but limit access to the spotify API.

Works Cited

Spotify Developers. “Build with Spotify’s 100 Million Songs,.” *Home | Spotify for Developers*, developer.spotify.com/. Accessed 6 Dec. 2023.

Spotify Developers. “Display Your Spotify Profile Data in a Web App.” *Display Your Spotify Profile Data in a Web App | Spotify for Developers*, developer.spotify.com/documentation/web-api/howtos/web-app-profile. Accessed 6 Dec. 2023.

Spotify Developers. “Embeds.” *Embeds | Spotify for Developers*, developer.spotify.com/documentation/embeds. Accessed 6 Dec. 2023.

Spotify Developers. “Introducing the IFRAME API.” *Home*, developer.spotify.com/blog/2022-03-31-iframe-api. Accessed 6 Dec. 2023.

Spotify Developers. “Using the IFRAME API.” *Using the iFrame API | Spotify for Developers*, developer.spotify.com/documentation/embeds/tutorials/using-the-iframe-api. Accessed 6 Dec. 2023.

Spotify Developers. “Vite.” *Next Generation Frontend Tooling*, vitejs.dev/. Accessed 6 Dec. 2023.