Bibliography

Chepkasova, V. (n.d.). Retrieved 5 September 2021, from <https://icons8.com/illustrations/author/5c07e68d82bcbc0092519bb6>

*How to Create, Read, Update & Delete Users in Asp. Net Core Identity*. (2018, November 19). *YogiHosting*. <https://www.yogihosting.com/aspnet-core-identity-create-read-update-delete-users/>

Murugan, M. (n.d.). *Custom User Management in Asp. Net Core MVC with Identity*. Retrieved 11 September 2021, from <https://codewithmukesh.com/blog/user-management-in-aspnet-core-mvc/>

Natural Earth. (n.d.). *Natural Earth.* <https://www.naturalearthdata.com>

OpenLayers (n.d.). *OpenLayers*. <https://openlayers.org>

Rick-Anderson. (n.d.). *Introduction to Identity on Asp. Net Core*. Retrieved 14 September 2021, from <https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity>

*Set Up Standard Payments*. (n.d.). Retrieved 10 September 2021, from <https://developer.paypal.com/docs/business/checkout/set-up-standard-payments/>

Ilgin:

**Paypal documentation:**

PayPal’s documentation was very extensive, clear and detailed. I had only used Google’s external API previously and did not have any experience in dealing with payments on apps that I’ve built. Paypal not only had clear instructions on how to integrate the API but also explained everything in an understandable way. They also had many different sections on how to customize it to suit your needs, which even though I didn’t have to use much due to the nature of our web app was still very interesting to read. The only thing I customized in the end was once the payment goes through, it redirects you to a new page, which changes the users role. However, this was quite easy to implement thanks to the extensive information they provide users with on their website.

**Introduction to Identity on ASP.Net Core (Microsoft Documentation):**

Even though I had previously used ASP.NET Core Identity, there weren’t many things I needed to customize or that required using their built in functions. However, with this assignment I needed to figure out how to manually update user roles and delete users in my code. My sources such as *Custom User Management in Asp. Net Core MVC with Identity and How to Create, Read, Update & Delete Users in Asp. Net Core Identity* were a great way for me to see a step by step guide on how to implement such functions but reading Microsoft’s own documentation was an even better way for me to understand the concepts of ASP.NET Core Identity and have an in depth view of what their built in functions can do and how I can apply them in my code to reach my goals. It was also very useful to read up on all the components and understand how they each work.

**Holden:**

**Natural Earth:**

Natural Earth provides geographic data that is open-source and free-to-use. In order to dynamically store and build map data, we opted to use GeoJson as the primary data type. GeoJson is an extension of the JSON format, allowing it to be used interchangeably with JSON in front-end, back-end, and database functionality. By using vector data to build visual elements rather than static images, it was much easier to manipulate and customise data created using GeoJson.

**OpenLayers:**

OpenLayers is an open-source, free-to-use JavaScript library that provides functionality to convert vector information into customisable graphic elements. This worked well for us in combination with GeoJson data to provide a custom experience that allows graphics to be produced on-the-fly within a user’s browser, reducing bandwidth and decreasing latency.