Ethan Armstrong
CART 351
Sabine Rosenberg
December 8, 2025

Project 3: The Interactive Worldwide Network

For this project, I wanted to create something that is interactive and involves a network or a group of people. Initially, when the concept of network first came to mind, my thought process was a connected digital world. I wanted to incorporate a world where everyone could input something unique about their background. I decided that since the world is very vast, I wanted to display a globe with all of the different countries where people could implement their own custom description of the country. This description would then be sent to the Mongo database and each description would then be returned to the user as an entry within the list of descriptions. Once the descriptions have been updated, the country's color on the map will then turn a brighter colour, signifying the amount of detail the country contains within its descriptions as well as representing the amount of people that contributed towards the corresponding country. Each user's description is displayed to everyone else, outlining everyone's various interpretations of a certain country. There are no limitations and all of the biases of each country could be very apparent or could be not so obvious. It's all up to the people.

In order to make this project work in the first place, I needed to find a map. I attempted various resources, however in the end I ended up using d3.js. D3.js is a Javascript data visualization library that is open-source and designed for use on the web. D3.js has many forms of data visualizations, such as maps, charts, and other forms of visual data. It uses SVGs or Canvas paths to convert geographic data into a visual map. In order to use this map, I also would need some data to work with. Thanks to GeoJSON, I was able to utilize map data from GitHub and transfer it to the corresponding map. GeoJSON is effectively a JSON file that stores country and geographic data such as borders, country shapes and their corresponding attributes like their country name. With the data that I accumulated I was able to plug it directly into the JavaScript file and post it to the Mongo database. It then returned the country's name and its corresponding description and user.

Ultimately, the whole process was very complex at first, since I had to understand for the first time how GeoJSON and d3.js worked. Understanding how paths work and having to set up the paths of the various countries so that they would match the corresponding country proved to be challenging, however I managed to get it to work successfully in the end. Since I haven't studied math in years, some of the calculations for the 3D mapping took a long time to figure out, but after trial and error I was able to graft my way through the problems in front of me. This was also true for most of the JavaScript code, which was easily the most difficult part of the project. Everything else, including the Flask instructions, the HTML and CSS were very straightforward for me to implement, since all it involved were routes sending and fetching descriptions sent in by the user.

Overall I'm fairly content with how the map turned out in the end, even though there was a very large setback. I was very satisfied with how the zoom and drag on the globe worked out since I expected to have a lot of issues with it but instead it basically worked as planned straight away. Despite this, there were unfortunately things that didn't work that I had extreme difficulty with that were unexpected. After submitting the description to Mongo, the colour wouldn't update on the map to match the description count and would remain the same colour. This proved to be an extreme downside for the whole project and didn't match my initial vision. I wasn't sure what the problem was and after a lot of attempts at debugging I eventually just decided to give in and leave it as is. As mentioned earlier I am still very proud of my attempt, and I don't think that the project was a complete failure even though the colour feature was essentially a bust. A feature to implement looking back would be determining if the message had case specific descriptions, for example if the word cold or hot was mentioned, then the colour of the country would become more red or blue the more times that word was mentioned in different descriptions. This feature however would rely entirely on the colour feature actually working as instructed, but this unfortunately never came to be. After all, the map functionality works, and that is what matters most within this context.

References:

GeoJSON Map:
https://github.com/holtzy/D3-graph-gallery/blob/master/DATA/world.geojson

d3.js Library:
https://d3js.org