Project 2 Documentation

Note: Please copy this into your personal portfolio! You can probably set up a PDF view if you're using Google Sites, which might be easier than copying in each section.

---

1. Data

The data we used for this project is sourced from the US Geological Survey (accessible here), and we are currently using data from 2014 to 2025. Each datum represents a recorded seismic event, complete with its longitude/latitude, time, magnitude, depth, and a variety of other identifying information. In our visualization, we highlight the magnitude and depth for analysis, and use the coordinates and time to narrow down the user's area of focus.

2. Visualization Components

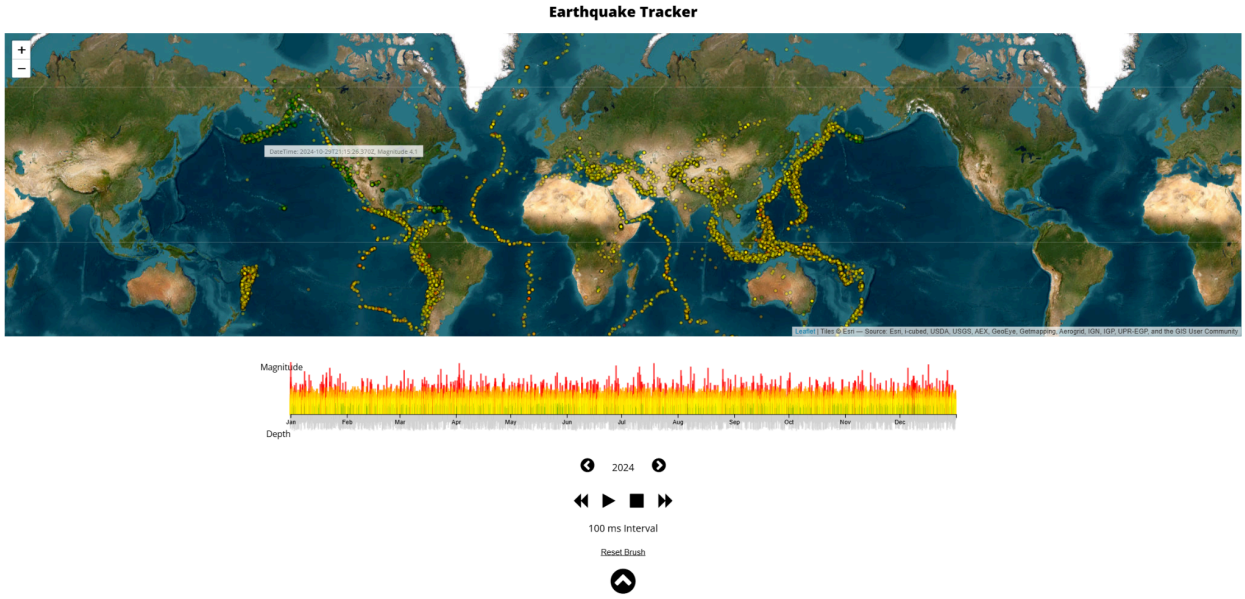

Fig 1. Default view from our completed project.

2.1. Map

Fig 2. Map component from our completed project.

The map component displays a significant amount of the data with each “dot” representing a specific seismic event. We’ll discuss the timeline next (which sits beneath the map in our GUI), but it’s important to mention that the two components are directly interrelated, and each dot in the map corresponds to a bar in the timeline bar graph. For additional information on a particular datum, the user can hover with the mouse and a tooltip will show the exact date, time, and magnitude.

This component is created using the leaflet library (accessible here). This library is used to create dynamic and interactive maps in Javascript.

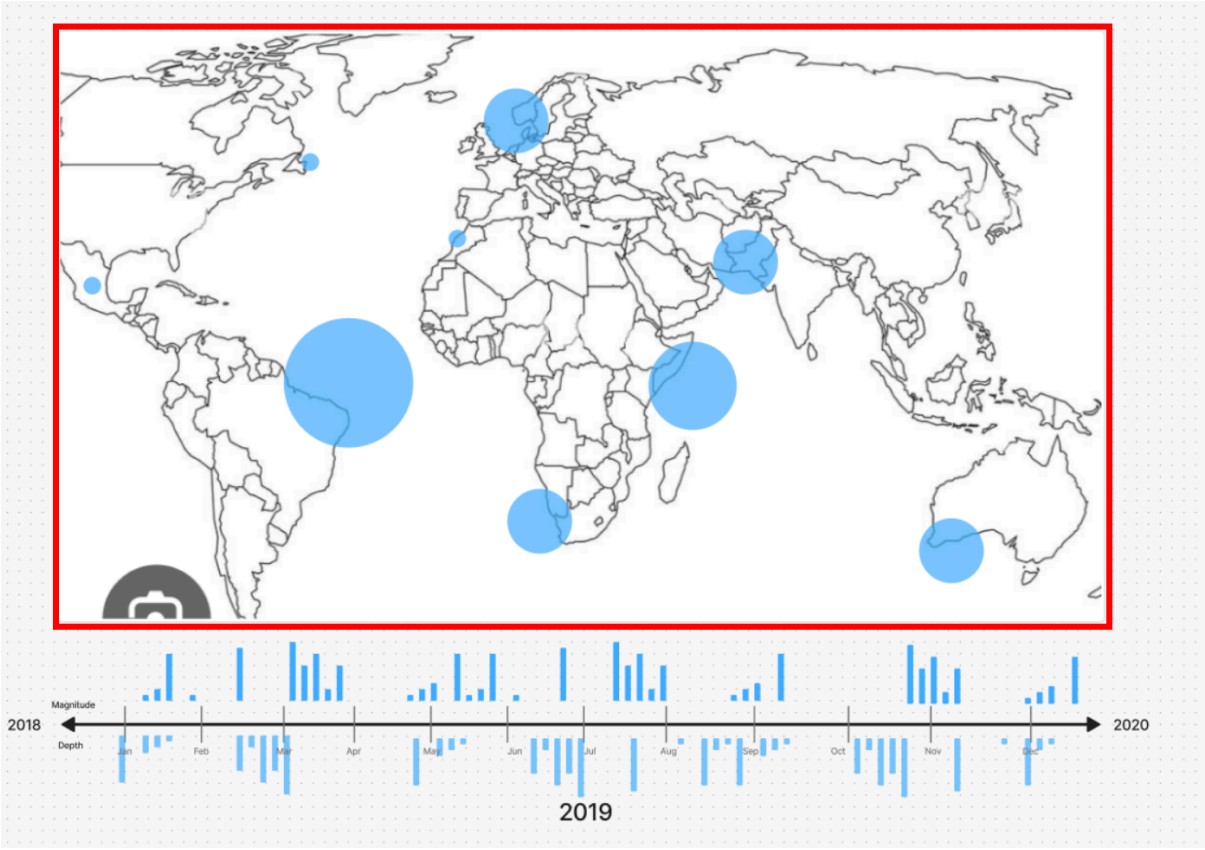

Fig 3. Initial sketch of the map and timeline (map highlighted in red).

From our initial sketch, we were fairly certain how we wanted the final product to look and behave, with a banner-like map seated above an interactive timeline that could be used to focus particular sets of data for inspection.

2.2. Timeline and Timeline Controls

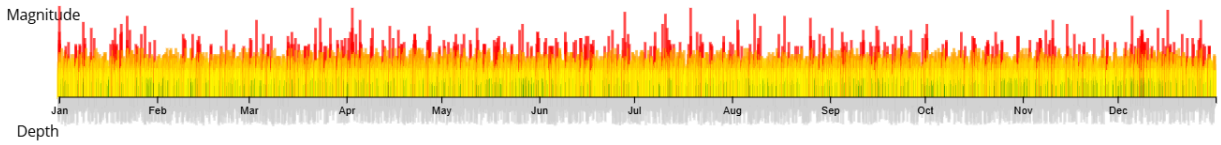

Fig 4. Timeline component from our completed project.

The timeline is a double bar chart used to display both magnitude and depth information over a particular timespan. As mentioned before, each bar in the bar graph corresponds to a point in the map component above. When combined with the timeline controls and ability to focus different timespans, this allows users to explore the data more freely, and without too much visual clutter.

For example, the user can select a certain “region” in the timeline to focus that timespan:

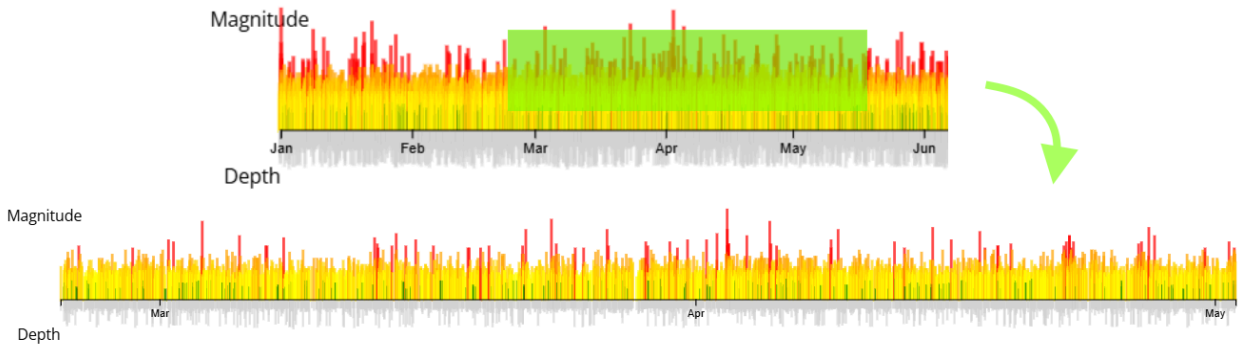

Fig 5. User selects a region from roughly March to May, and the timeline is updated accordingly.

Additionally, to support level 5 requirements, we have added a “timeline controls” panel beneath the timeline that allows for the selected timespan to be “animated,” showing each recorded datum one after another at a set speed. The controls panel can also be used to change the year that’s being shown, and due to the data we downloaded the years 2014-2025 are accessible.

Fig 6. Timeline controls component from our completed project.

Again, we determined the purpose of the timeline double bar graph early on in development, and this can be seen in our sketch (see Fig. 7). The idea to have depth shown on the bottom and magnitude on the top was our initial design, and we enhanced this with the control panel during development to add more functionality and usability.

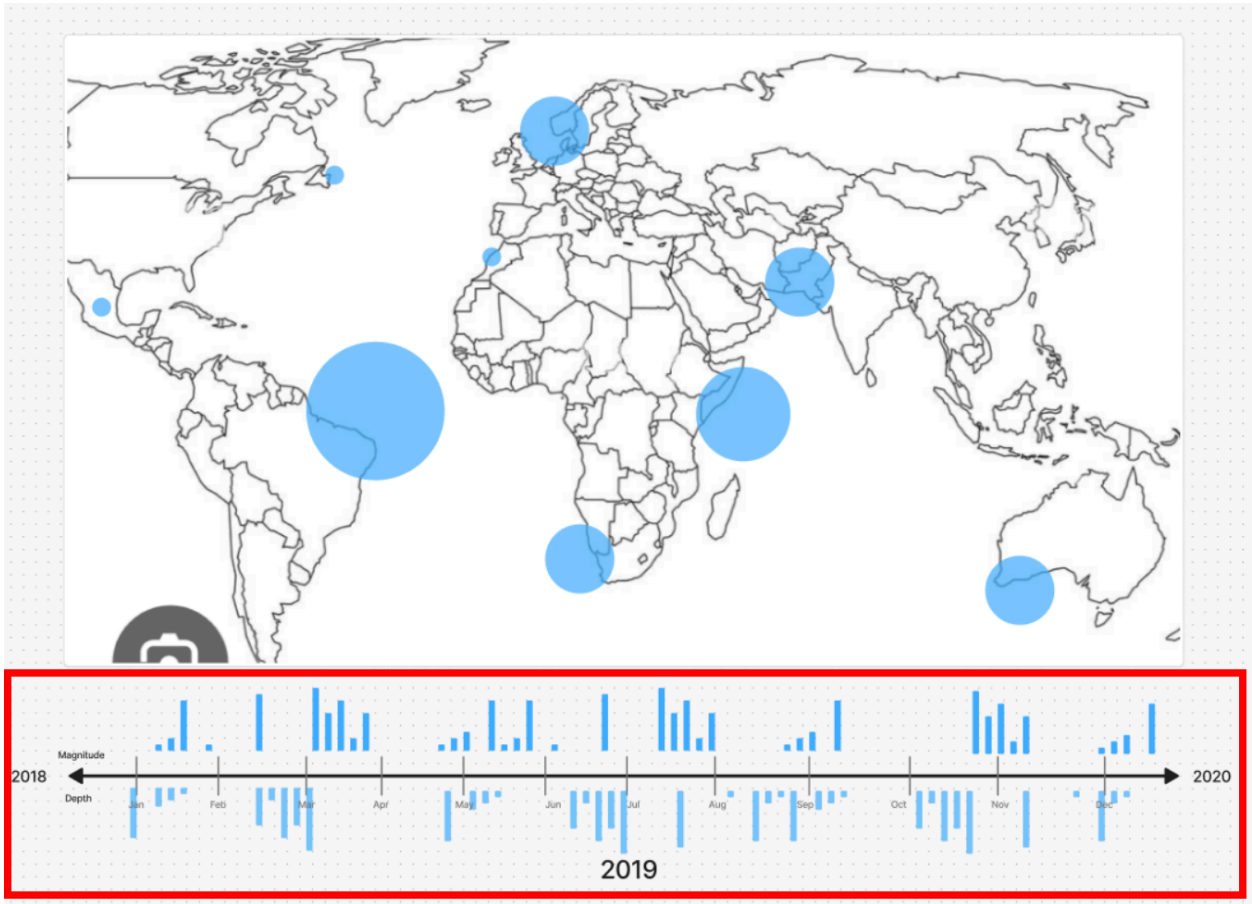

Fig 7. Initial sketch of the map and timeline (timeline highlighted in red).

2.3. Data Manager and Options

The next component enhances a lot of existing functionality and gives the user more control over what information is currently displayed on the map and timeline. We call this component the “Data Manager,” and it can be enabled by clicking a button in the bottom-center of the GUI.

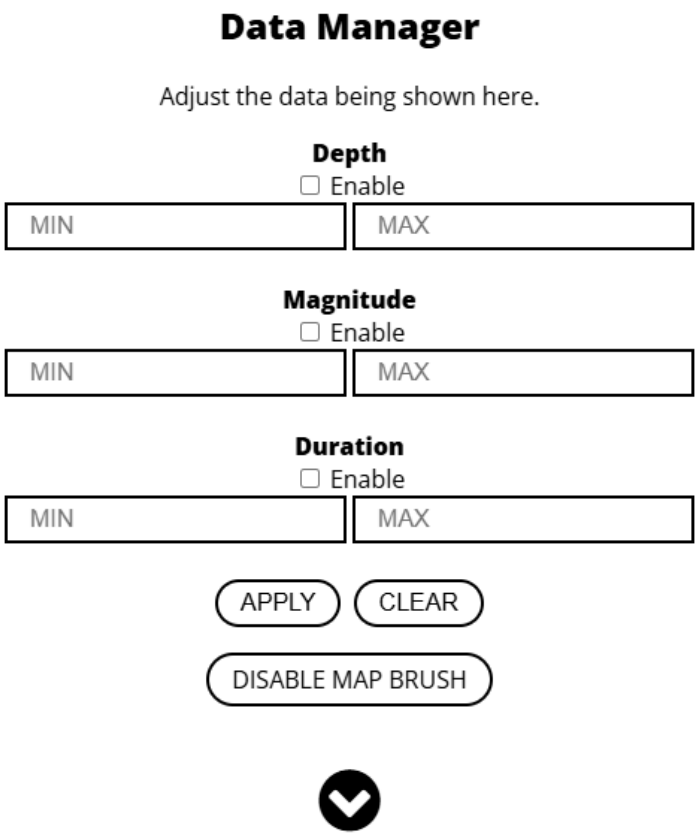

Fig 8. Data manager component from our completed project.

Fig 9. Initial mockup of data manager component.

From top to bottom (in Figure 8), the data manager supports a number of important features for this project: (1) the depth, magnitude, and duration filters can be used to filter out seismic event data, (2) the apply and clear buttons allow the user to commit or undo their changes, (3) and the “enable map brush” allows the user to select regions in the map for manual inspection (similar to how the brush that can be used in the timeline component).

From this menu, the user can also modify the map settings. This currently allows for different map backgrounds to be selected:

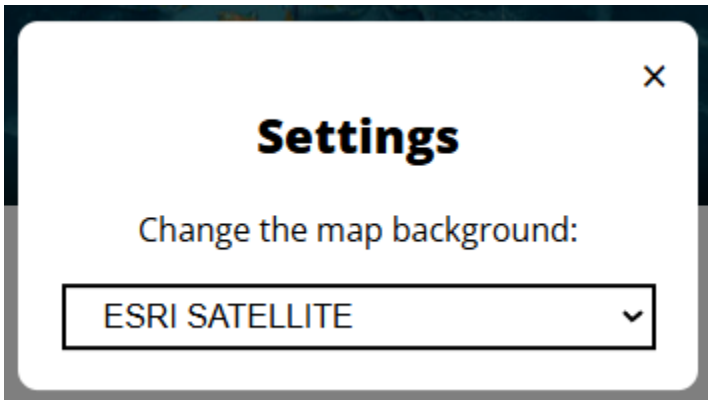

Fig 9. Map settings component from our completed project.

With the aforementioned components, our project succeeds in addressing levels 1 through 5 for this project, and gives users the ability to freely explore the USGS earthquake data with a variety of different tools, settings, and visualizations.

3. Discovery

A great visualization tool isn't "great" unless it allows you to uncover something interesting or valuable from the data it displays. During and after implementation, we made several compelling discoveries with our visualization.

Firstly, after completing the map, we were struck by how well-defined the fault lines are. In hindsight, this might be obvious, but the lines are extremely visible and clear when looking at the data from an aerial map perspective. Another interesting discovery was the extreme concentration of medium-to-high-magnitude earthquakes beneath Japan. In popular culture, this area is known for having a large number of earthquakes, but seeing it visually gives a whole different understanding to the phenomenon. After some brief research, we discovered that this was due to Japan resting atop the meeting point of *four* different tectonic plates, which can be observed in our visualization as well.

4. Technical Details & Process

Resources Used

- **USGS Earthquake Data**: As mentioned before, we used the USGS earthquake data for our project.
- **D3**: We used the D3 JS library to create in-depth visualizations, such as the timeline double bar graph.
- **Leaflet**: We used the Leaflet JS library to create our dynamic and interactive map.
- **HTML, CSS, JS**: Otherwise, our project is constructed using a default front-end web stack, including CSS for styling, HTML for structure, and JS for general scripting.

Code Structure

- Our code is structured simply, using JS classes to encapsulate certain components (for example, the map, or the timeline). We used a "css" folder for styles, a "data" folder for our datasets, an "images" folder for the icons/images used, and a "js" folder for our code and for the third-party JS libraries.

Links

- GitHub repository for our code: chendrix99/DataVisProject2
- Live-hosted link for our visualization tool: https://data-vis-project-2.vercel.app/

5. Demo

DV_Project2_Demo.mp4

6. Team Breakdown

Caleb Hendrix	● Worked on initial design sketches ● Worked on Level 1, 2, 4 goals
Jake Huseman	● Worked on initial design sketches ● Completed documentation and demo
Yale Miller	● Implemented tool tips and other visual features ● Worked on Level 6 goals
Prabhjot Singh	● Styling overhaul ● Worked on Level 5 goals
Chris Laney	● Worked on initial design sketches ● Worked on Level 1, 2, 3 goals