



Software Design

February 14, 2020

Version 2.0

Project Sponsors

Doctor Geoffrey Roest

Doctor Kevin Gurney

Team Faculty Mentor

Scooter Nowak

Team Members

Kiley Jacobs (Team Leader)

Tung Nguyen

Zihang Shen

Yisheng Wang

Table of Contents

1. Introduction	3
2. Implementation Overview	4
3. Architectural Overview	5
3.1 Data sources	6
3.2 Tilesets	7
3.3 Mapbox styles	7
3.4 Backend Code	7
3.5 Map GUI	7
3.6 Client's Website	7
3.7 Emissions Ranking	8
3.8 Data Download	8
4. Module and Interface Description	8
4.1 Data Sources	8
4.2 Tilesets	8
4.3 Mapbox styles	9
4.4 Backend Code	10
4.5 Map GUI	11
4.6 Client's Website	12
4.7 Emissions Ranking	14
4.8 Download page	15
5. Implementation Plan	15
6. Conclusion	16

1. Introduction

The Earth is warming at an alarming rate, and the growth of CO₂ emissions is a major reason for this change. Many countries have signed agreements to reduce CO₂ emissions, and many cities in the United States, like Los Angeles, are taking the lead in reducing CO₂ emissions. However, actions like diplomatic agreements are not enough. Without an easy to use way for everyday people to see environmental changes, people cannot observe and understand the changes of regional CO₂ emissions. Also, people don't know whether their actions really mitigate global climate change or not. So, our project is to create a Web Application that can give people a very clear visual map that shows how much CO₂ emissions are increasing, as well as being able to see the CO₂ emissions level for their own hometown.

The project sponsors are Geoffrey Roest, Postdoctoral researcher in SICCS (School of Informatics, Computing and Cyber Systems) at NAU (Northern Arizona University), and Kevin Gurney, Professor in SICCS at NAU. Professor Kevin Gurney has recent research in topics in carbon cycle science, climate science, and climate science policy, and is working on a project involving simulation and quantification of U.S. CO₂ emissions, the linkages between terrestrial carbon exchange and climate variability, and the impacts of deforestation on climate. He also has worked extensively on climate policy and has been involved for over 25 years with the United Nations Climate Change Framework Convention and the Kyoto Protocol. For this project they have collected large amounts of data about CO₂ emissions in the United States, and have been able to create a map over the country that displays this data. The problems that they are facing right now are dissemination and accessibility. Since the data is only available in technical formats, you need specific software to open and analyze them; the general public doesn't have sufficient knowledge or skills to use most of this software. So, our primary goal is to build an easily accessible interface that allows everyday people to understand and explore CO₂ emission findings. Our solution will be a Web application that can show the

CO2 emissions in an interactive graphical interface. Users can access our website and see a map of America displaying relative CO2 emissions. This map will be dynamic, allowing users to change things like color and transparency, as well as view information about specific areas on the map. In this document, we will talk more about our plan to develop this and our needed functions.

2. Implementation Overview

Our primary goal is to build an easily accessible interface that allows everyday people to understand and explore CO2 emission findings. The solution our team for our sponsor is an interactive map that can show the CO2 emissions in an interactive graphical interface. Users can access our map and see a map displaying relative CO2 emissions.

We have selected several tools that help facilitate our design. Since we are building a map application, we have chosen to use the Mapbox framework. Mapbox is free, easily integratable, and a JavaScript-based framework. Mapbox allows tileset rendering, and it can handle datasets far above interactive maps that need to load files like geoJSONs directly. Mapbox GL JS, which is mapbox's tool for backend development, also avoids repeated looping, a common issue in interactive maps, by using rendered layers and Mapbox GL JS functions to access specific features.

Tilesets are lightweight collections of vector data that are optimized for rendering and are not editable, but can be styled in the Mapbox style editor. We create a vector tileset by uploading geospatial data that contains attributes, the resulting vector tileset will inherit the attributes contained in the raw data source.

After we create the map style with our client's data, we will use JavaScript to create an interactive map that behaves in different ways in response to the user's actions.

JavaScript integrates easily within Mapbox to provide a dynamic application.

To implement this project, we are using Mapbox GL JS, Tileset, CSS, and JavaScript. These tools provide the robust functionality and intuitive integration needed for development. With these tools in mind, we begin to discuss our architecture as a whole.

3. Architectural Overview

An important part of a software system is the software development environment, as this has influence on the design, build, and testing. Especially for complex systems, it is important that this has been set up correctly to maintain and guarantee productivity and quality. The architecture of this product is to have users access the webpage on the client's already created website that displays the interactive map. The map is built using Mapbox and allows users to select different layers of the map, along with a number of other features. The user can also see the emission ranking and download data on the webpage.

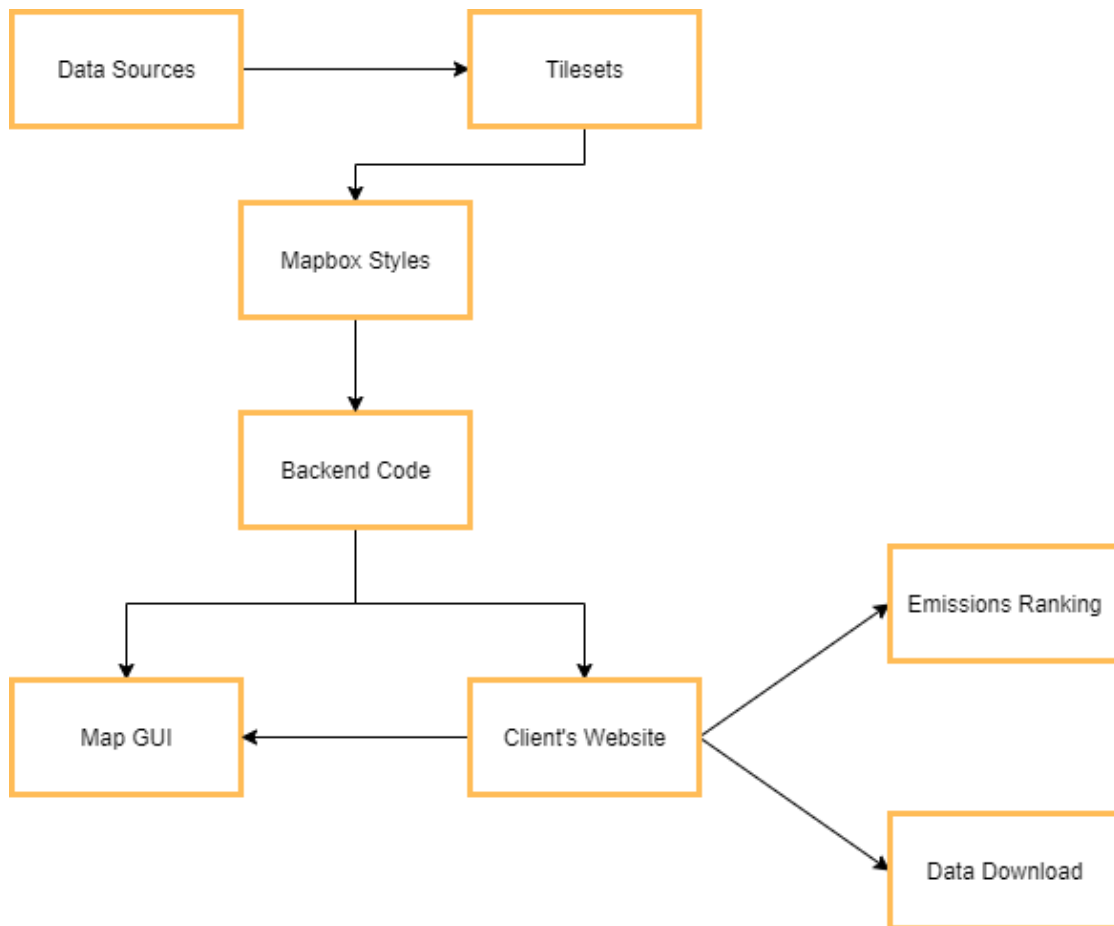


Figure 1: Architectural diagram

A general overview of the system flow is depicted in Figure 1. It shows each of the components that make up our project.

3.1 Data sources

Our data is mainly Geotiff images (raster data) client, which are data files compatible with Mapbox, allowing us to use backend code to manipulate the way the data is displayed. We can upload up to 10 gb to Mapbox. This data will be uploaded to tilesets to create image layers in Mapbox.

3.2 Tilesets

Tilesets are a collection of raster and vector data broken up into a uniform grid of square tiles. It is an optimized way to save and transport data by splitting in tiles.

Mapbox relies on tilesets to keep the maps fast and efficient. Mapbox style will take generated tiles to create a visual map.

3.3 Mapbox styles

Mapbox styles is a document that defines the visual appearance of a map, such as what data to draw, the order to draw it in, and how to style the data when drawing it. We use mapbox style to display the CO2 emission according to each city. Mapbox style has root properties that provide important descriptive information related to our map.

3.4 Backend Code

We use Mapbox GL JS, a Javascript library to render an interactive map from Mapbox styles. Javascript, HTML, and CSS are used to display emission ranking and data download as well.

3.5 Map GUI

The map GUI module is responsible for presenting the map to the user. This component provides the user access to all the functionality of our map. The map interface consists of functions written with Mapbox GL JS.

3.6 Client's Website

We will implement our web pages into our client's website. There are three web pages; the map, emissions ranking, and data download. Each page will have their own GUI, with the map being the bulk of our project as it is the most dynamic.

3.7 Emissions Ranking

This page will display the emission data by state per year. Users can change the years to see different ranking charts. The data will be displayed in a bar chart with the state totals.

3.8 Data Download

The web page has a list of data files sorted by years with a description for users to download. The data will be in the form of csv files and stored on our client's domain.

4. Module and Interface Description

4.1 Data Sources

Our data is provided by our clients. It is raster data that comes in the form Geotiff files. Geotiff is a standard image file format that includes additional spatial information imbedded in the .tif file. We use these images to display the CO2 emissions over the past 10 years.

4.2 Tilesets

Tilesets are the primary data format for Mapbox maps. Mapbox allows users to upload Geotiff data, as well as some other file formats, into Tilesets and store them in there. At the moment, the data provided by our client is only around 300MB, so the free quota provided by Mapbox is completely sufficient. We can directly use Mapbox's tilesets as our database. Figure 2 below shows how the data will upload into the Tilesets, and how this data will be converted to tessellated representation of geographic data, then show up as the layer.

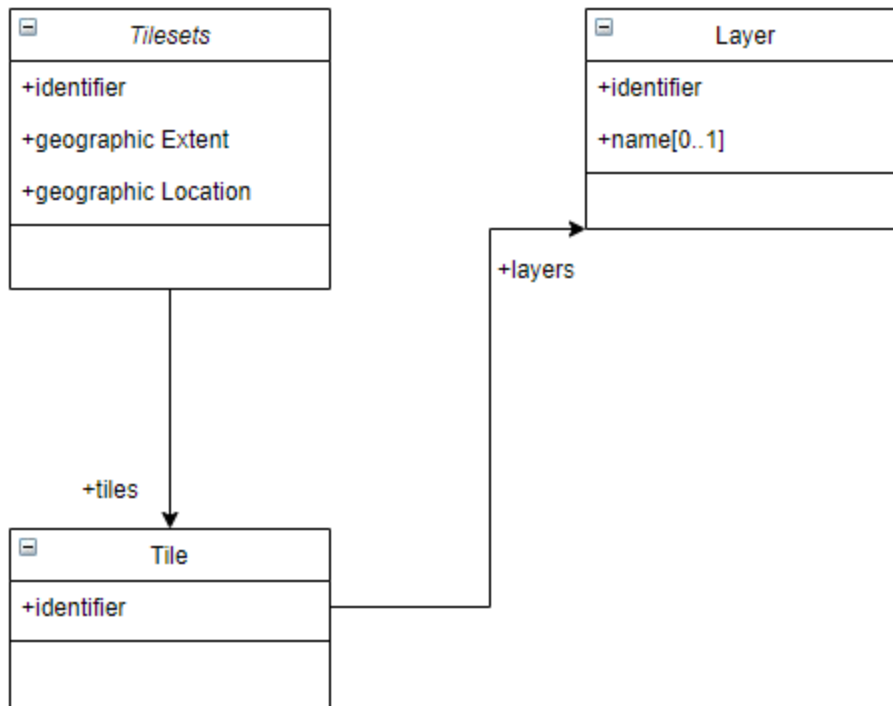


Figure 2. The Tileset diagram

4.3 Mapbox styles

The Mapbox styles take data that has broken up into tiles and convert it into layers. The render layer contains all classes that are responsible for rendering the geo-data on the screen. It has a powerful data infrastructure that supports custom data, format conversion, and processing. It allows us to change the background layer, add as many layers as we want, and toggle style like color, pattern, and opacity. The process is outlined in figure 3 below.

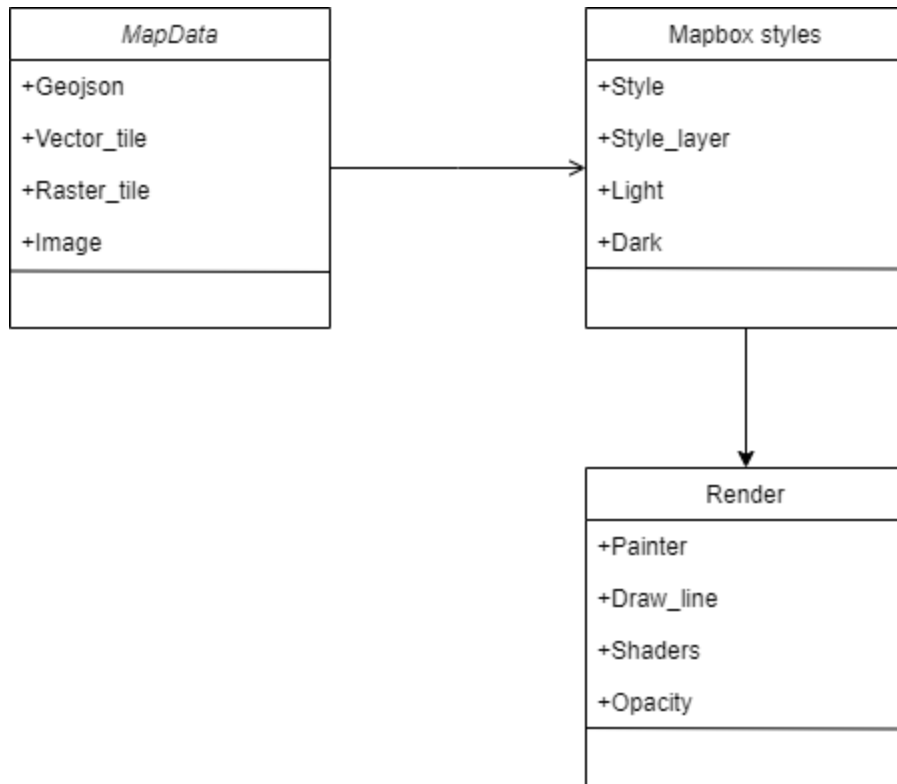


Figure 3. Mapbox styles diagram

4.4 Backend Code

Mapbox GL JS is a JavaScript library that renders interactive maps from vector tiles and Mapbox styles. It is part of the Mapbox GL ecosystem, which includes Mapbox Mobile, a compatible renderer written in C++ with bindings for desktop and mobile platforms. It will be what we use to write the majority of our backend code. The following list is all the functions we plan to implement in our project.

- Changing map being displayed - function in javascript that will make certain Geotiff files become visible and invisible
- Multiple maps can be shown in an overlay - function adjusts transparency to make all maps equally visible, i.e. if there are two makes shown transparency will be 50% for each

- Display information about certain geographic areas (state, counties) - certain icons on the map can be clicked to display things such as total emissions and per capita emissions. Data will ideally be loaded in using a csv, each line associated with the id number of a map icon
- Change color and transparency - function that allows r,g,b,s values to be changed based on user input
- Slider to view changes over sequential time (months, years) - slider can be clicked and change the image being shown to the next one in the time sequence, calling the function for changing the map being displayed
- Show hestia data - will be represented as another icon on the map over the few cities there is data for. Clicking on the icon will bring up the hestia data map, since it is 3D rather than 2D
- Show vector data - polygon based data. Will be uploaded and used to provide more detail to maps, such as lines for roads
- Show hourly emission changes - will be implemented if the client generates the data by the end of the semester, which at the moment is unlikely
- Download data files - a different page on the clients website separate from the Mapbox map. It will be made with simple HTML and CSS, allowing users to click on files to download
- View emission rankings - a different page on the clients website separate from the Mapbox map. It will be made with simple HTML and CSS, showing a simple list ranking of areas with the highest emissions

4.5 Map GUI

In the interactive map of carbon dioxide emissions page, users can see a map of the United States that takes up the entire page. In this map, the user can see the different CO2 emissions in different areas displayed by different colors. There is also timeslider, and different data sources so that users can choose to view certain kinds of emissions at a time. The map is also equipped with some small functions such as search for location and display the summary information under the mouse pointer, so as to give users a

better and more convenient user experience. And figure 4 show below is based on technical limitation and client meetings, so the user interaction will keep changing during the development.

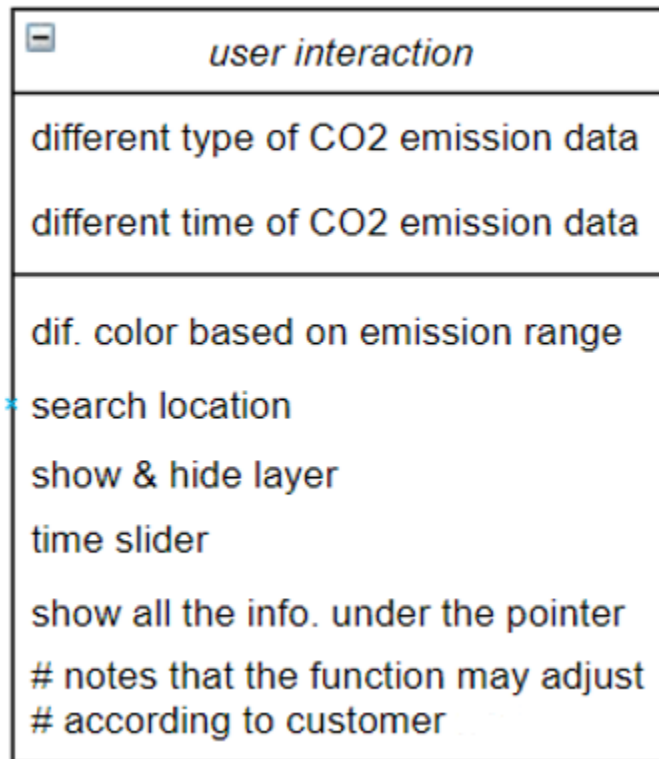


Figure 4. Some of the user interaction in the map

4.6 Client's Website

When the user goes to a client's website, there will be a tab to our project page. There will be three links, one for downloading the data, one for the interactive map of carbon dioxide emissions, and one for ranking the emissions. The user will also have a number of different things they can do on the interactive map. Figure 5 below shows this.

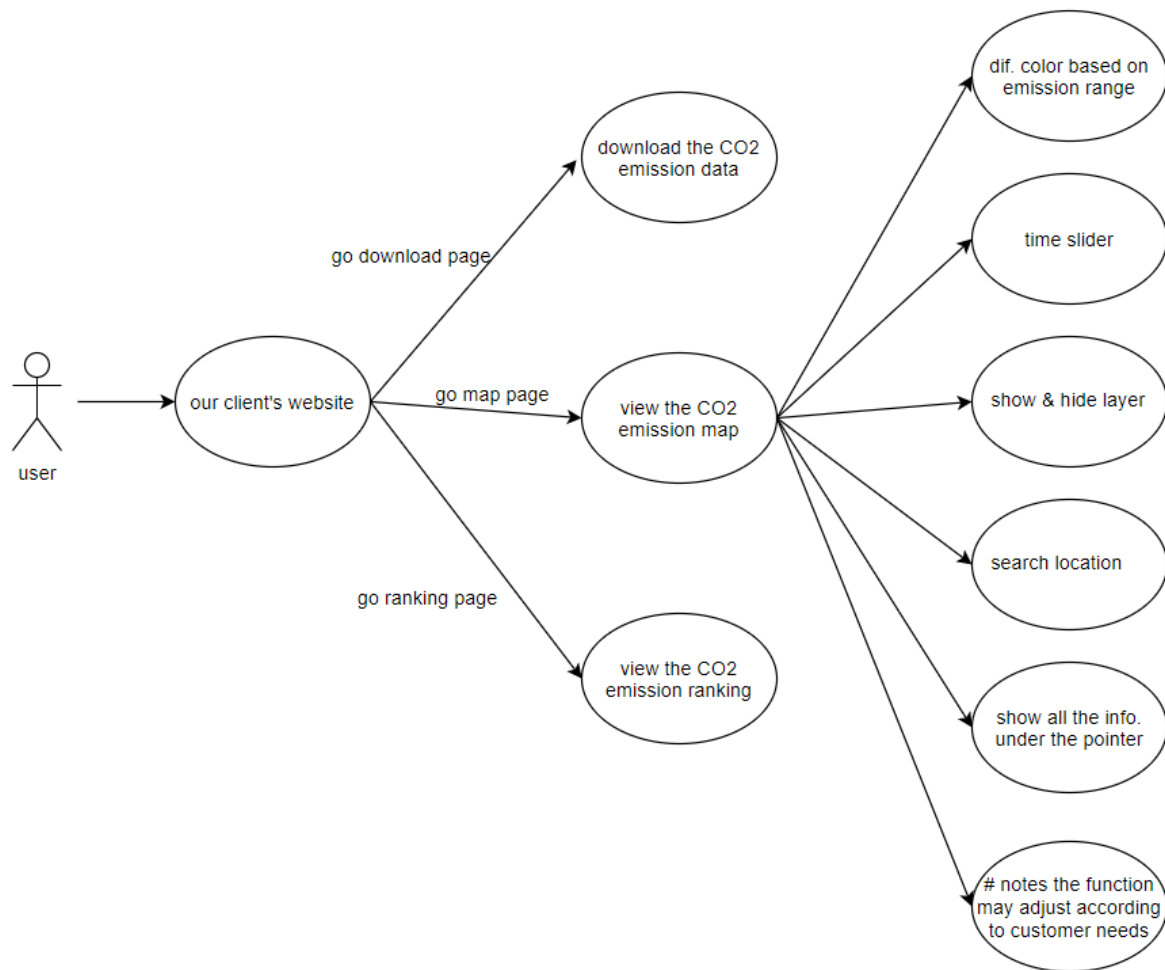


Figure 5. UML of user Use Case Diagrams

4.6.1 Transfer of authority

We will put our code files on the server where the client's website is located, and modify the customer's webpage slightly to link it with our map page, ranking page, and download page. Also, the Mapbox account and the email that links to the Mapbox account will be given to our client. At the same time as the delivery of the website, we will also accompany it with the delivery management maintenance and use manual.

4.6.2 Management and maintenance

Like the figure 8 show below, the clients should be able to manage and maintain these three pages. Also, they can add new data if they wish to in the future. You can upload

the data to Tilelets in Mapbox, and then simply add and modify the code according to the management maintenance and use manual to display the new layer on the map. Modifying rankings and downloading pages will be easier than the map. The instructions will also be contained in the management maintenance and use manual.

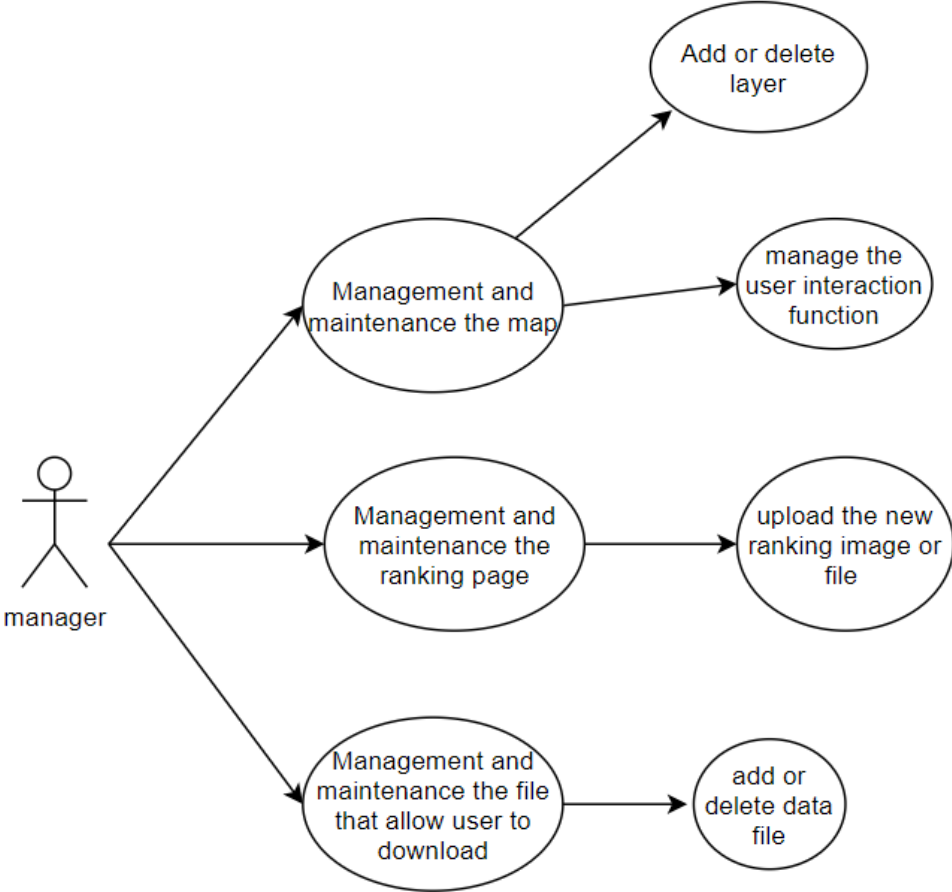


Figure 6. UML of manager Use Case Diagrams

4.7 Emissions Ranking

This data will also be obtained from the client, then shown on the website as pictures or tables. Users can pull up summary statistics such as emissions ranking, sector totals, and per capita averages. Figure 7 shows some of this user interaction.


 <i>user interaction</i>
Compare emissions rankings

Figure 7

4.8 Download page

The webpage has links to some data files with brief descriptions. This allows users to download these data files. It contains data used for interactive maps. Figure 8 shows this interaction.


 <i>user interaction</i>
Download data

Figure 8

5. Implementation Plan

In order to plan out the schedule for implementation, there are several things on which we need to focus. The first thing is to divide the system into several components. The advantage of this is we can have a better overview of the development process, understand the importance of each component, and make sure that the important and fundamental components have top priority in the whole implementation process. The second thing is to ensure a smooth transition between each task and components, and consider the unexpected circumstances of any delays. The third is to increase parallelism with multiple components being worked on at any given time, which will help to improve efficiency.

Below is the detailed plan for the project. The green part is the things we already completed, the red part is things in progress, and the blue part is things to do in the future. We have built templates for each of our pages and outlined the duties of each of our modules. We have outlined a set of tasks which will bring us to completion of the necessary functionality of the system.

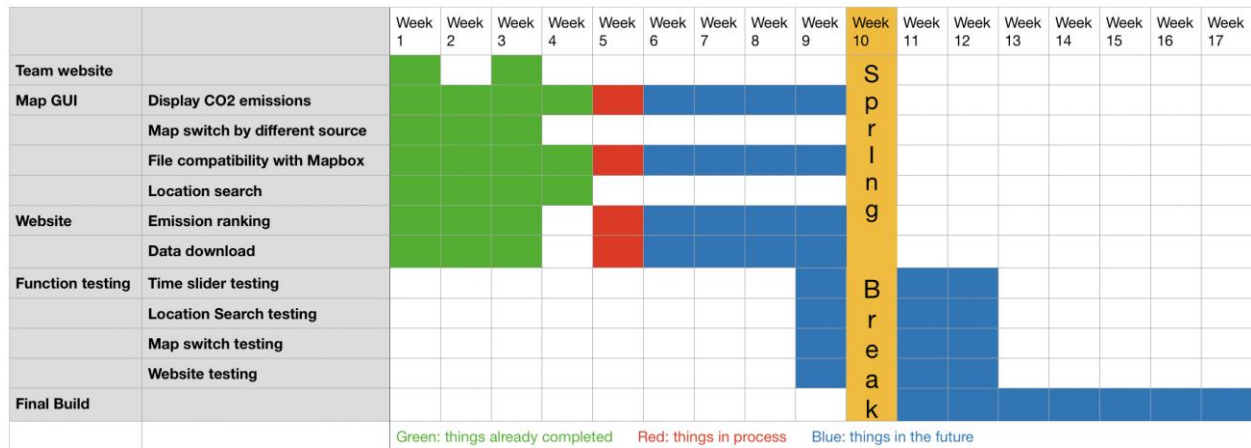


Figure 9. Implementation Plan

Before spring break, we will demo our alpha prototype, containing a fully implemented back end. Once this is done, we will begin testing. Testing at this time will consist of internally checking for major bugs, and gathering feedback from our clients. After this, we will work on polishing the front end so we can deliver a well made website at the end of the semester.

6. Conclusion

This document served as a description of the structural makeup of the CO2 emission map. In this document, we list the tools used for development. These tools are HTML, Tileset, JavaScript, Mapbox. This document outlines each component of our system and how they relate to one another. The modules described are our website, Mapbox studio, the functions we are going to write using Mapbox GL JS, and Tileset. Finally, we detailed our plan for implementation using a Gantt chart. The Gantt chart shows when each module is going to be developed, and which team member is responsible for the

module's implementation. As we continue with our implementation plan, this document will serve as our guide. Using it, we will be able to work toward one vision.

In order to let people see CO2 emission in their hometown as well as the whole America, under the sponsorship of researcher Geoffrey Roest and Professor Kevin Gurney, we have come up with a project to create an interactive map, each different part in our software design plan plays an important role in this interaction, and we planned time for us to develop each module and test them as well. Now, we are working on the map, and we believe we will have a satisfying product in the end, as it will provide a tool useful for anyone who wishes to learn about what is happening to the environment.