CS^2

CSM Jobe 2 Final Report

Field Session Summer 2021

Client: Dr. Zane Jobe

Project: Carbon sequestration dashboard

Team Members:

Matt Plumb, Grant Falkner, Ryan Armstrong, Patrick Schassberger

Introduction

Dr. Zane Jobe at the Colorado School of Mines wanted to expand the school's presence in the field of Carbon Sequestration, the long term capture of CO_2 gases. The project aims to provide geologists with an overview of various datasets gathered from individual government websites such as basins, power plants, well platforms, layered over a map of the U.S.

Currently this data is spread across various agencies and their respective websites, making it a challenge to visualize the data as a whole. Thus, Dr. Jobe proposed an interactive dashboard that hosts all of the diverse datasets and plots their locations and summary features. Further subplots then break down individual basins and their total storage potential and recent emissions. This will give users a comprehensive overview of major geological areas, their total storage potential, and their typical emissions. This will aid users in their research or planning related to the Carbon Sequestration process.

The website is set up to be hosted on a Colorado School of Mines web server as a means to display the school's commitment to combat climate change. This project is intended to be updated in the future as the field of CCUS grows and users' needs call for more information to be displayed.

Requirements

High Level Description

Dr. Zane Jobe's career goal is to transform traditionally qualitative geological observations into quantifiable, usable, and visualizable data. We aimed to contribute to this goal by compiling data surrounding the subject of geological carbon sequestration and plotting it on an interactive map residing on a website. This map is intended to clearly organize and present data to users in the field of carbon sequestration. It should allow users to make better informed decisions surrounding the subject.

Functional Requirements

- A main web page that holds all information on carbon sequestration being presented
- A collection of data related to carbon sequestration (sources, sinks, etc.) in the U.S.
- An interactive map that displays the collected data over a world map
    - User can select which data is shown/hidden
    - User can mouse over data points to see detailed information about the feature
- Graphs that highlight meaningful data comparisons
- An about page that refers the user to the respective website each dataset was pulled from

Non-Functional Requirements

- We needed to display maps and graphs on a website, so we chose Dash, which is a Python framework built on Flask, Plotly.js, React, and React.js, and is intended for building web dashboards
- We needed to write some logic to handle our data so we used Python for our back end since our front end was already based in Python
- We decided to host our website on a server in the Mines domain since the client wanted the project to be clearly associated with Mines

User Story

Dr. Jobe shared his struggles about being able to view a cohesive overview on carbon sequestration features, namely carbon sources and sinks grouped by geological basin. Currently all of the data is spread across various websites and databases. Our goal was to produce a map of the U.S. that displays all of the relevant information for industry professionals. They should be able to use the map to inform themselves and make educated decisions about potential sites for carbon sequestration.

System Architecture

Description

Figure 1 displays the general layout and design of how our website functions. The user will be able to access the page on the Colorado School of Mines web domain. The project flow is as follows:

- User accesses website
  - User directed to the main page
    - Main map of US displayed
      - User can show/hide datasets
      - User can inspect data points for more information
    - Graphs highlighting key CCUS data
      - User can inspect data points for more information
    - About page button
      - User directed to the about page
        - Links to websites on which each dataset was found

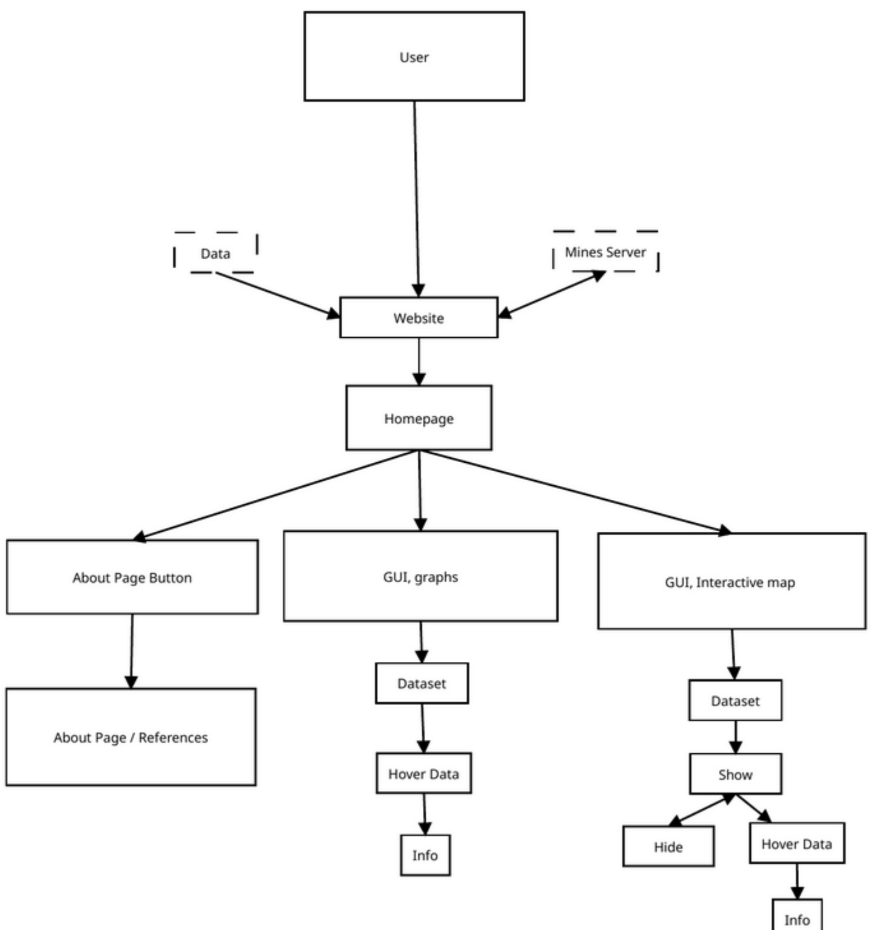

Figure 1

Technical Challenges

- General Difficulties
  - During development, finding the correct web-framework to host our project was difficult due to our lack of experience in web development. Initially, we used a Django framework. However, the Django project was more heavy duty than necessary and gave us issues with integrating Plotly graphs. Thus, we moved to a Dash framework, an extension of our Plotly packages.
  - Deploying the project on a Mines server became a time consuming process. ITS has a lengthy process to host a website and, although the project is ready, we are still in communication with the team to get SSL certificates and approval for deployment.
- Data Wrangling
  - Although finding data was fairly straightforward, we had a very diverse set of data types including csv, txt, gdb, and shapefiles. The Geopandas package allowed us to work with all data-types and their geometry (latitude, longitude) columns and use the resulting dataframe to plot the results.
- Graphing
  - With the large spread of data present, finding a common graphing utility was quite difficult. Making custom plots for each dataset would have led to slow performance and load times. The Plotly graphing library for Python provided us with an interactive map of the US and the ability to add unique traces and description for each set of data.

Technical Design

Figure 2 below shows the interactive map, the main feature of our website. The user can show or hide a dataset using the legend on the right. All datasets are hidden by default to reduce load time and eliminate cluttering. They can also hover over any geological point or shape to display information about the feature. The map is rendered by Plotly on top of an open source map called OpenStreetMap. The individual datasets were read into Geopandas dataframes that allowed us to plot the geolocation of each data point on scatter maps. These individual scatter maps were then superimposed onto the OpenStreetMap. Users can zoom and pan to any area on Earth similar to the interface on Google Maps. Any user-added datasets will automatically appear in the legend assuming the proper configuration procedure is followed (refer to the appendix to see this procedure).

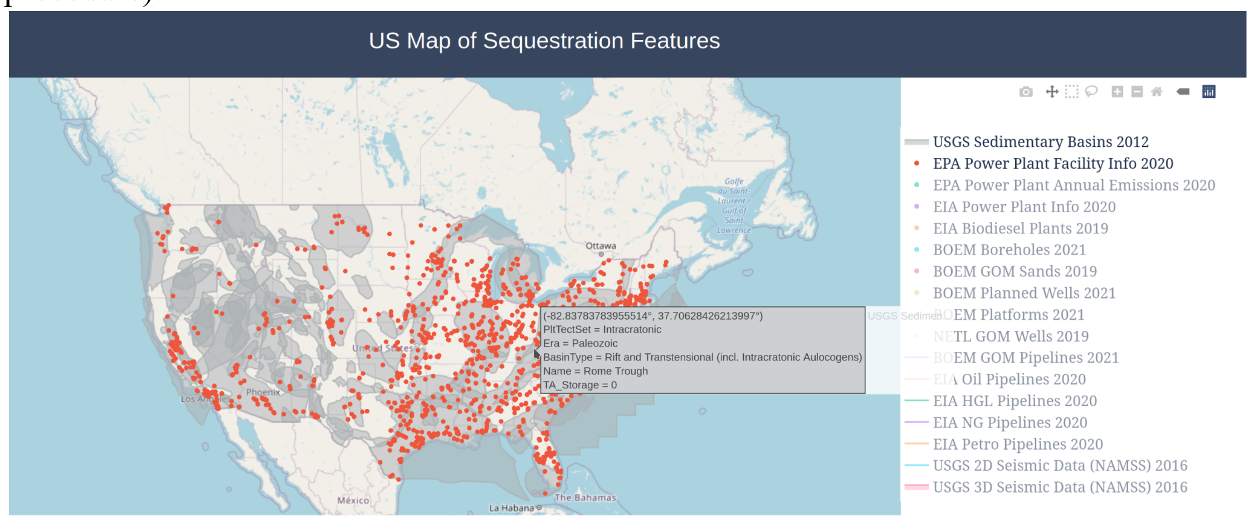

Figure 2

Figure 3 below implies the total basin sequestration potential through plotting the 2020 emissions of a particular basin vs. the total amount of storage of the basin. This plot was created by calculating the area for each basin and cross checking if a power plant (CO^2 source) is located within that area. Each collection of power plants' CO2 output is then summed and the resulting value is associated with the respective basin. For each basin, this calculated emissions value is then plotted against the total storage available in that basin, a value already calculated for us by the USGS. The user can hover over each point to view the basin it represents and the total storage and 2020 emissions for that basin (Mt stands for megatonnes). A logarithmic scale was applied to the graph to improve readability since there are a wide range of values.

Figure 3

In our last graph, shown by Figure 4, the user may select any basin from the dropdown menu and the two bar graphs will be populated with that basin's 2020 emissions and total storage, respectively. The user can hover over each bar to clarify the basin attribute being displayed and view its exact magnitude. A logarithmic scale was applied to this graph since most emissions values are so much smaller than their corresponding storage values that the emissions bar was not visible to the user. The main purpose of this graph is to showcase the massive amount of space available in each basin compared to the amount of CO2 emitted in a recent year. Hopefully this will encourage users to support the blooming field of carbon sequestration.

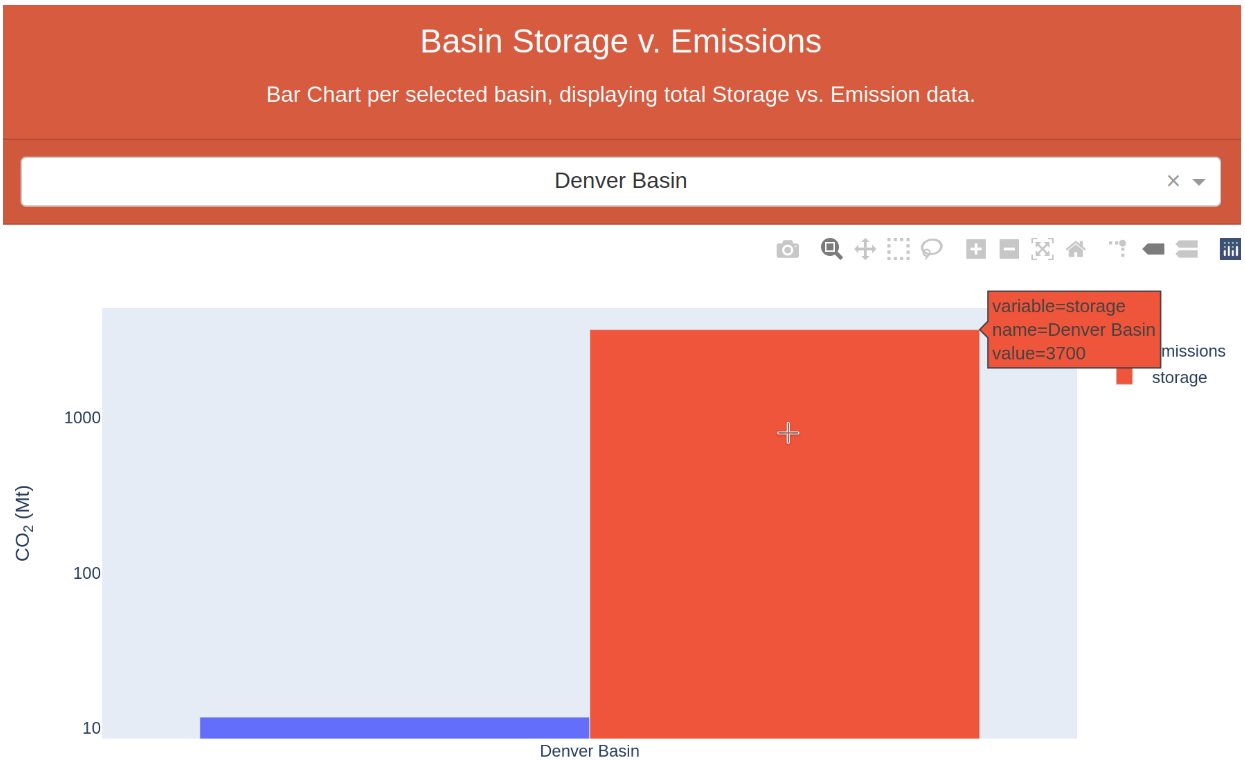

Figure 4

Quality Assurance

Code Quality:

To make sure we have the highest quality software, we as a group have done numerous code reviews each week to ensure that the quality of our software is not only up to our standard, but up to the standard that our client and the general public expect. We also performed various types of testing to ensure the functionality and implementation of our project is working as expected and as efficient as possible.

Testing:

- Dash Testing
  - Uses Pytest and Selenium
  - Ensure project is properly hosted
  - Browser API
  - Client API
  - Testing graph callbacks
- Pytest Unit Testing
  - Run a number of unit tests when the website is built
  - Unit tests ensure that all the functions are performing as expected
- User Acceptance Testing
  - Mitch Schneider, a graduate student in the geology department at the Colorado School of Mines used our website to give us great feedback
  - Weekly deployment test
  - Ensure new features work as expected on our server

Communication:

Consistent communication between all of the team members has been our greatest QA asset. Adhering to our team schedule, meetings, and code reviews have enabled us to consistently challenge our ideas and assumptions about the product and its intended functionalities.

- Weekly code review meeting
- Team communication/meetings
- Weekly meetings with client

Results

The initial goal of this project was to create a functional CO2 sequestration dashboard that gives users a general overview of sinks, sources and possible transportation routes. After our third sprint, the initial requirements were met with a functioning website, interactive map and representing summary features. Our team was then able to present a prototype to our client and were able to move on to a user testing phase. We tested our product with multiple users including a geology graduate student as well as our client, Dr. Jobe.

User testing by the graduate student of the geology department gave us the feedback necessary to refine our user interface. After general UI testing, we moved on to testing our website on multiple browsers including Safari, Firefox, Google Chrome, Microsoft Edge, and Opera. We also tested on multiple OS's including Windows, Mac, Linux. We were able to implement all of our key features, such as the interactive map, representing the data, and creating subgraphs. We also ensured that our project is robust across all major platforms and browsers.

Overall, this project asked us to create a website to host various geological datasets and to provide the user with a clean and interactive dashboard of the United States. Using Agile methodologies and Scrum methods, our team was able to implement all of the necessary features outlined to us by our client, Dr. Zane Jobe. In these past five weeks, time management and managing expectations were the greatest challenge. However, communication and collaboration allowed us to complete the project to the client's specifications.

Future Work

- Connecting backend to geological database.
    - One feature that we did not have time to implement was connecting our website to a geological database.
    - This would allow for geologists to query the data and would serve to speed up the performance of the backend.
- Adding additional carbon sequestration data to the site.
    - Due to time restrictions we did not get around to adding all of the planned datasets to the site.
    - Adding more datasets would be useful for geologists to visualize additional information about carbon sequestration.
    - The datasets we were planning to add include...
        - Brine formation Data - Information about geological formations which could support carbon sequestration.
        - Onshore Plays Data - Visualizing onshore plays would help break up onshore basins into chunks to better understand the geography of our map.
        - CCUS Facility Data - Having data on existing CCUS facilities on our map would show users where carbon is already being sequestered.
        - EPA Emissions Data - The EPA has a web map which shows the emissions of individual basins. This data could be manually entered to more accurately portray the emissions of a basin.
- Deploying the site outside of the CSM network.
    - We were unable to fully deploy our website outside of the CSM firewall.
    - Most of the infrastructure is in place for deployment, however a SSL certificate is still needed for users to verify the authenticity of our site.
    - Deploying our site will involve coordinating with Mines ITS.
- Grouping datasets by features.
    - One potentially useful feature we were unable to implement was the ability to group data points in datasets by a particular feature.
    - For example, you could group a power plant dataset by what kind of emissions they primarily produce. This would allow for a more comprehensive view of power plant emissions.
    - Grouping datasets would be useful for geologists to sort through the big datasets to extract they information that they need
- Adding a distance tool.
    - A feature that we did not end up being able to implement was a distance tool on our map.
    - A distance tool would help geologists visualize the distance between key features on our map which could be useful for planning potential carbon sequestration.

Appendix

<u>Client Instructions</u>

```
{
    "name": "BOEM Boreholes",
    "file": "CleanedBoreholes.csv",
    "file_type": "csv",
    "latcol": "LAT",
    "loncol": "LONG",
    "attributes_to_display": [
        "Spud Date", "Total Depth Date", "API Well Number"
    ]
},
```

The website should be straightforward to navigate to, view, and interact with. If the client intends to add or remove datasets or alter which columns of a dataset are displayed, they will need to edit the Data/config.json file. We created this file specifically for the purpose of future flexibility with the data. To add a new dataset, first, the client should add the file containing the data to the Data directory. Then, add a new entry to the config file with the same format as shown in Figure 5. Here is a brief description of each field in an entry:

name: The display name of the dataset when shown in the legend of the map
file: The name of the file containing the data
file_type: The file format (only csv, gdb, and shp files are supported)
latcol (csv only): This is the title of the column containing latitude values in the file
loncol (csv only): This is the title of the column containing longitude values in the file
attributes_to_display: This is a list of titles of columns in the file that will be displayed in the hover info on the map

The client would then have to push the altered config file to the GitHub repository and wait for redeployment.