DSCI 560: Data Science Professional Practicum

Laboratory Assignment 3 Part 1

Team Details:

Team Name: Team PVC

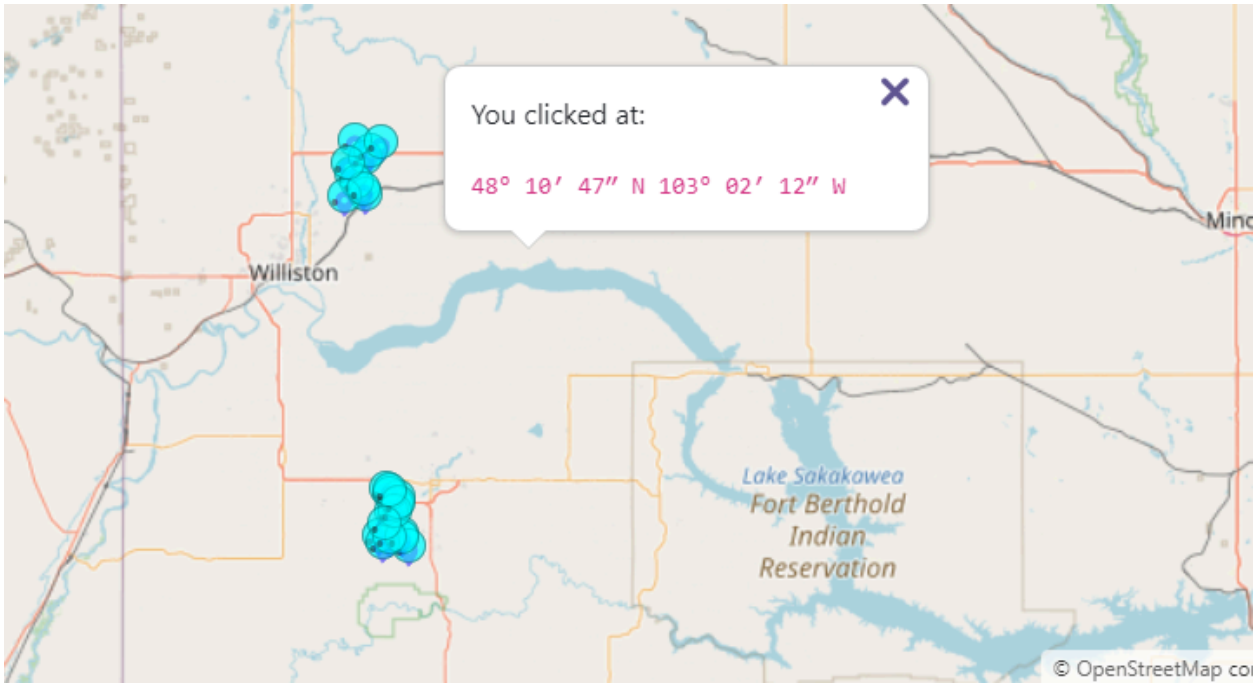
Team Member	USC ID
Haoran Zhang	3502885083
Venkatesh Dharmaraj	6773159759
Yichen An	2780765696

Overview:

For this part of the lab assignment, we managed to map the data points we extracted/scraped from the wells API to the world map on overlay based on their longitude and latitude.

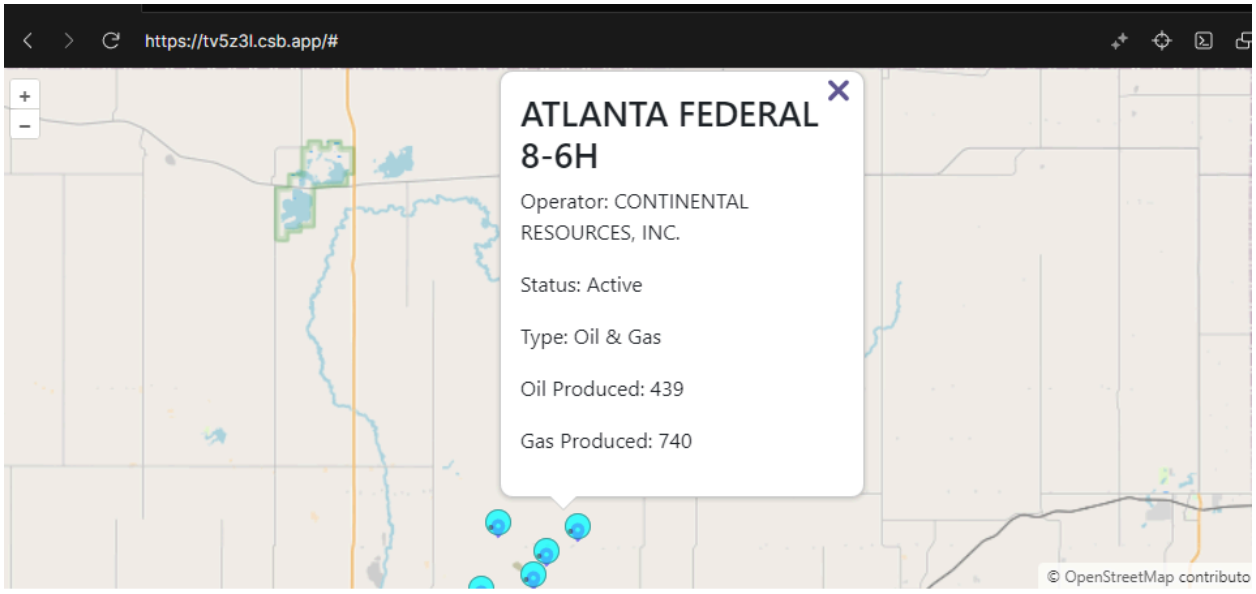
Step 1:

We created a map layer first then add the click events layers on the map, so whenever you click somewhere on the map, it will show you the coordinates of that particular click event, as demoed here:



Step 2:
Then we create another layer to create the markers for the wells location we extracted from part 1 of the lab based on their coordinates, as shown in the blue round circles.

Step 3:
We add the content to the marker element once we clicked on the wells being mapped into the map to show more details, as shown below



Approach:
Because the dataset wasn't too large so we decided to embed it into the main.js, then we iterate over each well's location to map it into the map, then we iterate it over again to get the information we needed and add that as content for those markerelements the to show in the popup window for the wells.

Snippet of the code:

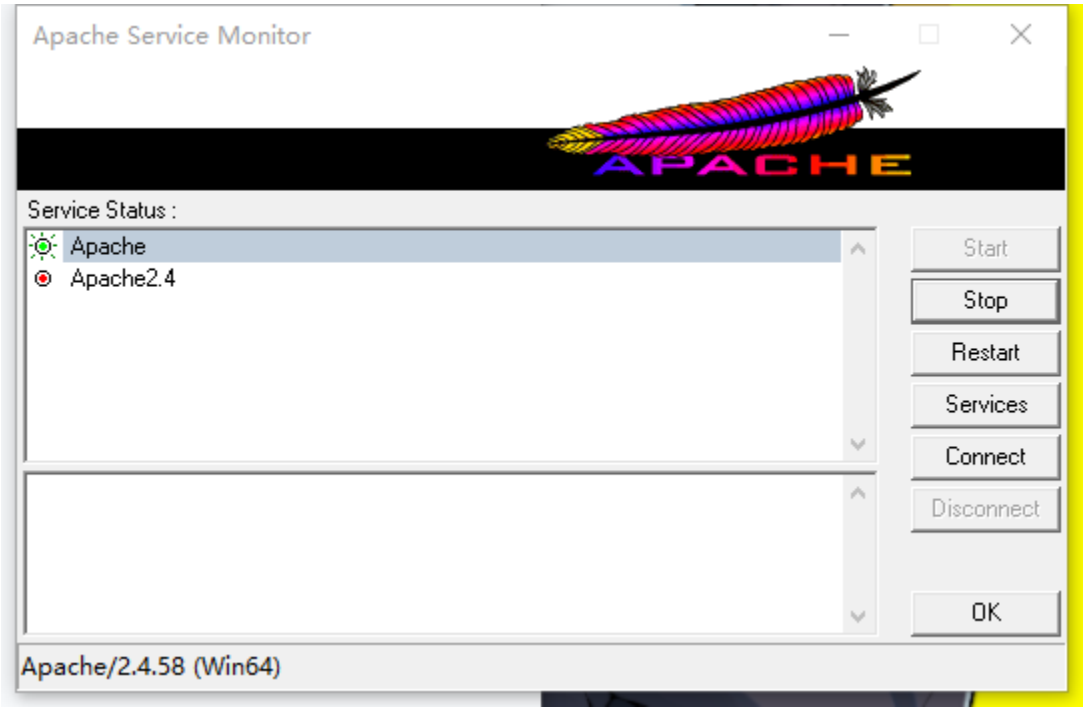
```
const data = {
  Latitude: [
    47.761754, 47.819371, 47.797499, 47.784166, 47.739902, 47.739899, 47.73
    47.810918, 47.784412, 47.795107, 47.726358, 47.821291, 47.807544, 47.74
    47.742482, 48.356754, 48.313052, 48.302505, 48.390889, 48.392563, 48.37
    48.326461, 48.305767, 48.364423, 48.340015,
  ],
  Longitude: [
    -103.44106, -103.428976, -103.406924, -103.416205, -103.430277, -103.39
    -103.445451, -103.430186, -103.422764, -103.413793, -103.380882,
    -103.439433, -103.407977, -103.400159, -103.454755, -103.530196,
    -103.492882, -103.538961, -103.451468, -103.516522, -103.476148,
    -103.511229, -103.490393, -103.487729, -103.523915,
  ],
};

// Convert the data into an array of [longitude, latitude] pairs
const positions = data.Latitude.map((lat, index) => [
  data.Longitude[index],
  lat,
]);

// Iterate over positions to create and add markers
positions.forEach((coords) => {
  const pos = fromLonLat(coords);

  // Dynamically create a marker element
  const markerElement = document.createElement('div');
  markerElement.className = 'marker';
  markerElement.innerHTML =
    '<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24" fill="blue
```

```
const wellsData = [
  {
    API: '33-053-04853',
    'Well Name': 'TALLAHASSEE 3-16H',
    'Lease Name': 'CONTINENTAL RESOURCES, INC.',
    Location: 'McKenzie County, ND',
    Operator: 'CONTINENTAL RESOURCES, INC.',
    'Well Status': 'Active',
    'Well Type': 'Oil & Gas',
    'Township Range Section': '153 N 101 W 16',
    'Closest City': 'Williston',
    'Oil Produced (as of Dec 2023)': 478,
    'Gas Produced (as of Dec 2023)': 828,
    Latitude: 47.761754,
    Longitude: -103.44106,
  },
  {
    API: '33-053-05924',
    'Well Name': 'CHALMERS 5301 44-24 2TR',
    'Lease Name': 'OASIS PETROLEUM NORTH AMERICA LLC',
    Location: 'McKenzie County, ND',
    Operator: 'OASIS PETROLEUM NORTH AMERICA LLC',
    'Well Status': 'Active',
    'Well Type': 'Oil & Gas',
    'Township Range Section': '153 N 101 W 24',
    'Closest City': 'Williston',
    'Oil Produced (as of Dec 2023)': 338,
```

```
wellsData.forEach((well) => {
  const position = fromLonLat([
    parseFloat(well.Longitude),
    parseFloat(well.Latitude),
  ]);

  const markerElement = document.createElement('div');
  markerElement.className = 'marker';
  markerElement.innerHTML = '<span class="marker-icon">•</span>';

  const marker = new Overlay({
    position: position,
    element: markerElement,
  });
  map.addOverlay(marker);

  // Attach a click event to the marker element
  markerElement.addEventListener('click', (e) => {
    e.stopPropagation(); // Prevent map click event from firing
    const content = `
      <h3>${well['Well Name']}</h3>
      <p>Operator: ${well['Operator']}</p>
      <p>Status: ${well['Well Status']}</p>
      <p>Type: ${well['Well Type']}</p>
      <p>Oil Produced: ${well['Oil Produced (as of Dec 2023)']}</p>
      <p>Gas Produced: ${well['Gas Produced (as of Dec 2023)']}</p>
    `; // we can include other details if needed

    // Set the popup content and position
    document.getElementById('popup-content').innerHTML = content;
    popup.setPosition(position);
  });
```

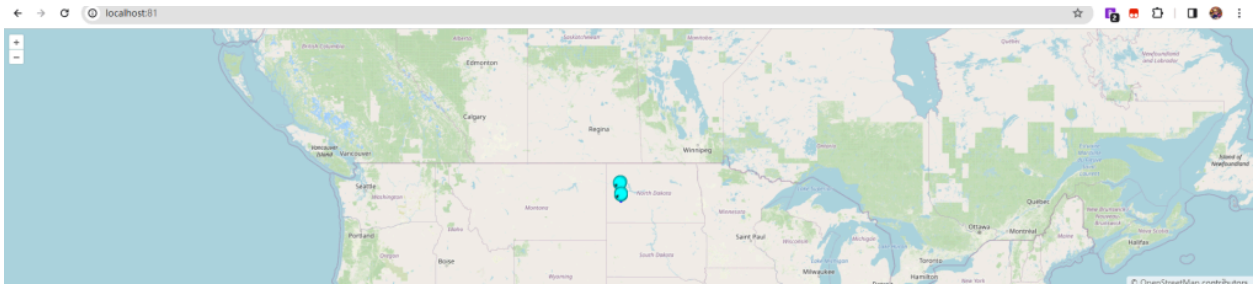
Apache Web Service set up:

Deploying a web application involves several key steps. First, we start by developing the application locally, often using tools like Vite or Webpack to help streamline the development process. Once the application is ready, we run a build script—typically `npm run build`—which compiles the application into static files that can be served by a web server.



Next, we set up the web server that will host the application. For instance, if we are using Apache, we ensure that it's properly installed and configured on my server machine. This includes setting the `DocumentRoot` to the directory where my application's files will reside, and ensuring the server is configured to handle my app's routing needs, especially if it's a single-page application (SPA).

After that, we upload the built files to the server. This could be through FTP, SSH, or a continuous integration/continuous deployment (CI/CD) pipeline that automates the process. Once the files are in place, we navigate to the domain where my application is hosted to check if everything is running as expected. If there are issues, we examine the server logs and the browser console for any errors and debug accordingly.



Throughout the deployment process, we make sure to keep security in mind, configuring HTTPS and setting appropriate headers for content security policies. Deploying a web application is not just about getting it online but also ensuring it operates reliably, securely, and efficiently for all users.