Weather Station Project Summary

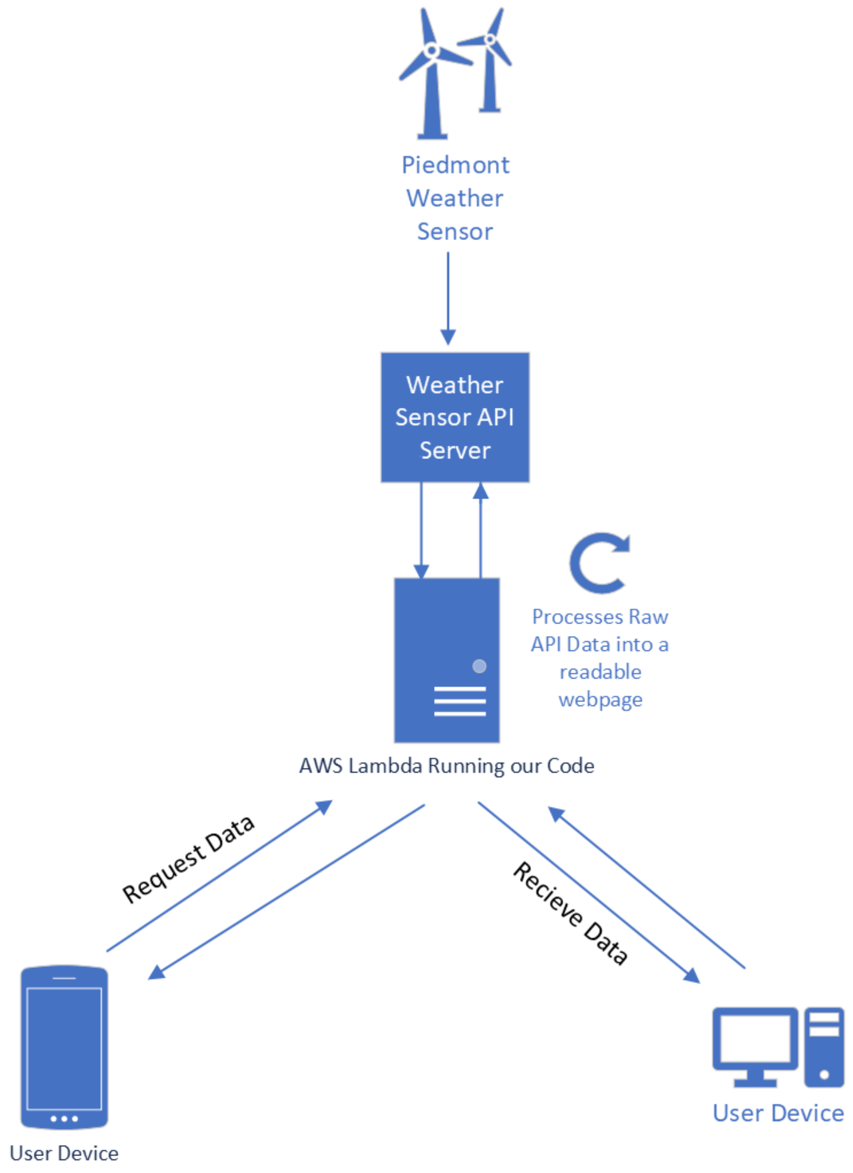
By Anthony Newton and Jeremy Cooper

Abstract: Our project aimed to provide students with access to a branded weather portal, offering real-time data from an on-site weather sensor. The system's core components include an initial backend, a method for configuring the backend into HTML code, and a streamlined design eliminating the need for file creation, relying on object manipulation.

Major Milestones:

1. **Creation of Initial Backend:**
   - Developed the foundational backend infrastructure to facilitate data retrieval from the on-site weather sensor.
2. **HTML Code Configuration:**
   - Implemented a method to translate the backend functionality into HTML code for seamless integration with the front end.
3. **Streamlined Design:**
   - Optimized the system by adopting a design philosophy that utilizes objects, eliminating the need for excessive file creation.

Code Diagram:


Piedmont Weather Sensor
Weather Sensor API Server
Processes Raw API Data into a readable webpage
AWS Lambda Running our Code
Request Data
Recieve Data
User Device
User Device

Processes and Procedures:

1. **Project Source Code and Artifacts:**
   - Code was initially roughed out, followed by iterative testing and refinement.
   - Emphasis on using industry-standard libraries to ensure code robustness and future maintainability.

   link to Source Code

2. **Documentation:**

- Comprehensive documentation created to facilitate future development and troubleshooting.
- Documentation includes system architecture, API references, and usage guidelines.

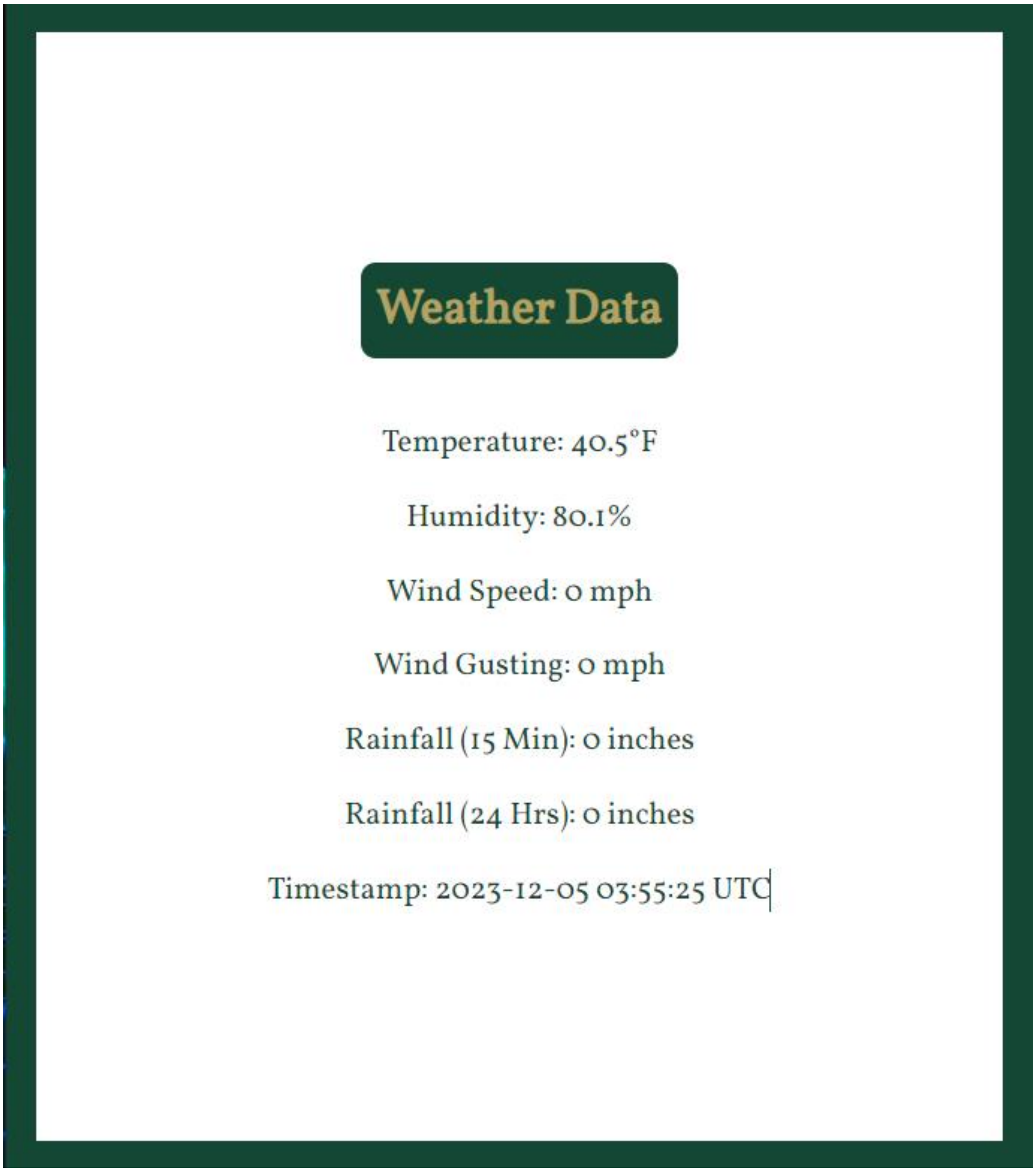
Link to Documentation

3. **Testing:**

- Rigorous testing conducted at each development stage.
- Focus on refining code to meet industry standards and ensuring the reliability of real-time data retrieval.

Link to Testing Documentation

Screen Captures:



Weather Data

Temperature: 40.5°F

Humidity: 80.1%

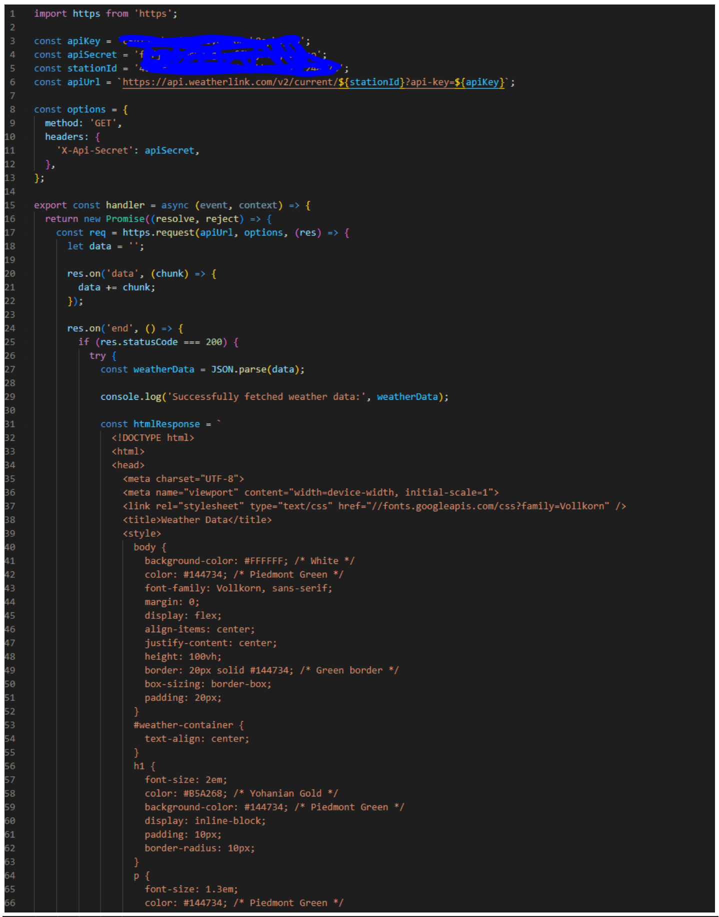
Wind Speed: 0 mph

Wind Gusting: 0 mph

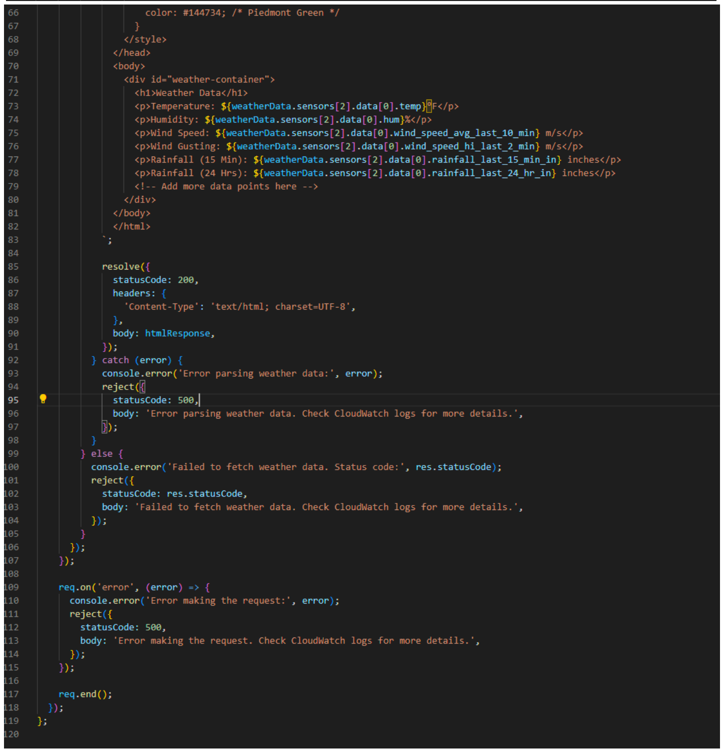
Rainfall (15 Min): 0 inches

Rainfall (24 Hrs): 0 inches

Timestamp: 2023-12-05 03:55:25 UTC



```
import https from 'https';

const apiKey = [redacted]';
const apiSecret = 'f[redacted]';
const stationId = '[redacted]';
const apiUrl = `https://api.weatherlink.com/v2/current/${stationId}?api-key=${apiKey}`;

const options = {
  method: 'GET',
  headers: {
    'X-Api-Secret': apiSecret,
  },
};

export const handler = async (event, context) => {
  return new Promise((resolve, reject) => {
    const req = https.request(apiUrl, options, (res) => {
      let data = '';

      res.on('data', (chunk) => {
        data += chunk;
      });

      res.on('end', () => {
        if (res.statusCode === 200) {
          try {
            const weatherData = JSON.parse(data);

            console.log('Successfully fetched weather data:', weatherData);

            const htmlResponse = `
              <!DOCTYPE html>
              <html>
              <head>
                <meta charset="UTF-8">
                <meta name="viewport" content="width=device-width, initial-scale=1">
                <link rel="stylesheet" type="text/css" href="//fonts.googleapis.com/css?family=Vollkorn" />
                <title>Weather Data</title>
                <style>
                  body {
                    background-color: #FFFFFF; /* White */
                    color: #144734; /* Piedmont Green */
                    font-family: Vollkorn, sans-serif;
                    margin: 0;
                    display: flex;
                    align-items: center;
                    justify-content: center;
                    height: 100vh;
                    border: 20px solid #144734; /* Green border */
                    box-sizing: border-box;
                    padding: 20px;
                  }
                  #weather-container {
                    text-align: center;
                  }
                  h1 {
                    font-size: 2em;
                    color: #B5A268; /* Yohanian Gold */
                    background-color: #144734; /* Piedmont Green */
                    display: inline-block;
                    padding: 10px;
                    border-radius: 10px;
                  }
                  p {
                    font-size: 1.3em;
                    color: #144734; /* Piedmont Green */
```



```
                    color: #144734; /* Piedmont Green */
```

```
                  }
                </style>
              </head>
              <body>
                <div id="weather-container">
                  <h1>Weather Data</h1>
                  <p>Temperature: ${weatherData.sensors[2].data[0].temp}°F</p>
                  <p>Humidity: ${weatherData.sensors[2].data[0].hum}%</p>
                  <p>Wind Speed: ${weatherData.sensors[2].data[0].wind_speed_avg_last_10_min} m/s</p>
                  <p>Wind Gusting: ${weatherData.sensors[2].data[0].wind_speed_hi_last_2_min} m/s</p>
                  <p>Rainfall (15 Min): ${weatherData.sensors[2].data[0].rainfall_last_15_min_in} inches</p>
                  <p>Rainfall (24 Hrs): ${weatherData.sensors[2].data[0].rainfall_last_24_hr_in} inches</p>
                  <!-- Add more data points here -->
                </div>
              </body>
              </html>
            `;

            resolve({
              statusCode: 200,
              headers: {
                'Content-Type': 'text/html; charset=UTF-8',
              },
              body: htmlResponse,
            });
          } catch (error) {
            console.error('Error parsing weather data:', error);
            reject({
              statusCode: 500,
              body: 'Error parsing weather data. Check CloudWatch logs for more details.',
            });
          }
        } else {
          console.error('Failed to fetch weather data. Status code:', res.statusCode);
          reject({
            statusCode: res.statusCode,
            body: 'Failed to fetch weather data. Check CloudWatch logs for more details.',
          });
        }
      });
    });

    req.on('error', (error) => {
      console.error('Error making the request:', error);
      reject({
        statusCode: 500,
        body: 'Error making the request. Check CloudWatch logs for more details.',
      });
    });

    req.end();
  });
};
```

Demonstration / Expo Video:

Submitted by Anthony to Canvas

Future Work:

The primary focus for future work is enhancing the frontend by connecting it to a more visually appealing user interface. The current basic HTML and CSS design will be upgraded to provide a more user-friendly and aesthetically pleasing experience. Future efforts will involve refining the user interface to align with industry standards and incorporating advanced web development technologies.

Conclusion: In the course of the project, we successfully developed a weather portal, achieving significant milestones in backend creation, HTML code configuration, and design streamlining. The emphasis on industry standards, comprehensive documentation, and rigorous testing positions the project for future enhancements. Moving forward, attention will be directed towards refining the frontend to create a more polished and user-centric experience.

Link to final Project:

Weather Station Link