Updating, Improving, and Maintaining the Online Watershed Learning System Website

Colden Bobowick[1], Michelle Ramsahoye[2], Yunus Naseri[3], Akshat Kothyari[3], Kang Xia[4]

[1] Department of Computer Science, Brown University, [2] Department of Computer Science, University of Colorado, Boulder,

[3] Department of Engineering Education, Virginia Tech, [4] School of Plant and Environmental Sciences, Virginia Tech

Objectives

Improve and redesign the Online Watershed Learning System (OWLS) website of the Learning Enhanced Watershed Assessment System (LEWAS) Lab.

This included:

- aesthetic updates to the website overall
- improvements to the interactive live data graph

Programming Languages Used

- Hypertext Markup Language (HTML):
  - Structured, straightforward language that is used to add elements to the website.
  - Text, dropdowns, and images are some HTML elements that appear in the OWLS website.
- Cascading Style Sheets (CSS):
  - Markup language used to style HTML sheets.
  - Each HTML page references a CSS stylesheet that contains different parameters like background color and borders for each class.
  - Bootstrap v5.2:
    - a popular toolkit that utilizes HTML, CSS, and Javascript (JS) to make responsive web apps and pages.
- Javascript (JS):
  - Used to execute different functions on the website.
  - Examples of JS functions for the OWLS site include creating the graph and downloading data.

Aesthetic Changes

The previous website used its own classes to divide the pages into different sections. All of the sections in the OWLs website had a constant size that was chosen to ensure that all of the elements within a section would be visible to all users, regardless of the size of the user's screen (desktop, mobile, etc.). However, the constant size left a lot of empty space. Overall, the stylesheet characteristics resulted in a "dated" look (see Figure 1).

Web development frameworks are sets of tools that are used by developers to build and manage web applications, services, and websites. The framework chosen to update the look of the OWLs website and add **responsiveness** was Bootstrap v5.2. In web design, "responsive" implies that that web page is flexible, and its layout can detect a visitor's screen size and orientation, and then adjust accordingly.

By implementing the use of Bootstrap features, we are able to maintain the original function of all the existing JS scripts while providing a modern look with larger font, eye-pleasing colors, and a responsive layout that fully functions on any device.

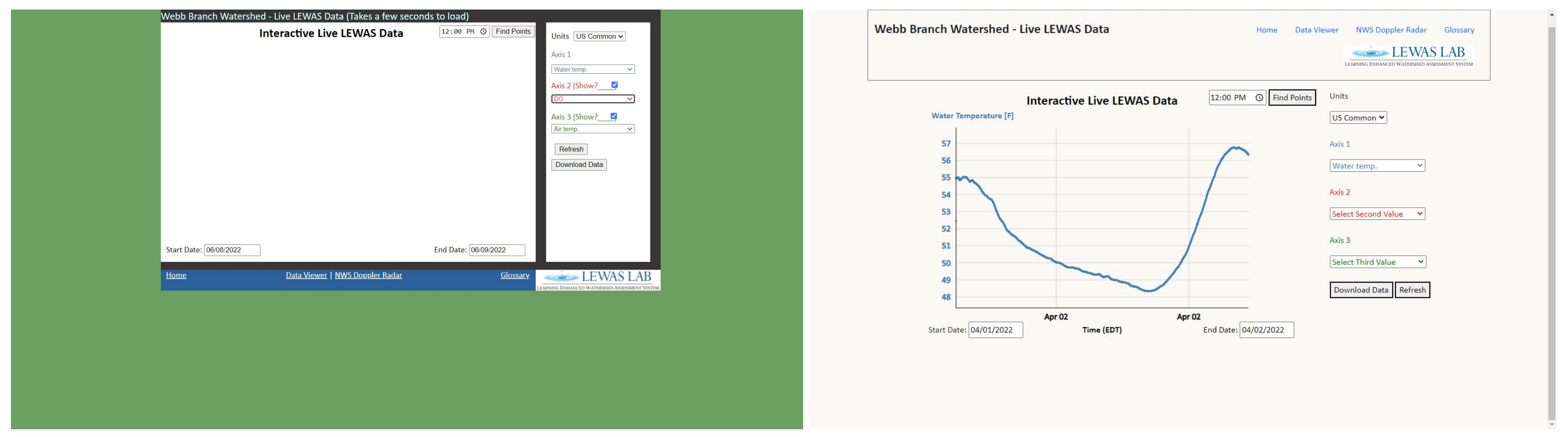
Webb Branch Watershed - Live LEWAS Data
Interactive Live LEWAS Data
Download Data
LEWAS LAB

Figure 1. The previous layout for the OWLS website (left) and the current, updated version (right).

Graph Improvements

The JavaScript responsible for the LEWAS data graph functionality was completely overhauled. We found the implementation of the code to be redundant with significant potential to be optimized.

Issues:

- Graph would not load or refresh when expected.
- Users would stare at a blank graph waiting for it to load.
- The LoadCSV() function was called every time the graph loaded, adding 400 ms to the load time.
  - LoadCSV() generates and stores the entire contents of the CSV file that would be outputted if the user chose to download the data.
- LEWAS database contains millions of data points, and trying to plot millions of these points on a website is simply unfeasible.

Solutions:

- Added functionality so that changing the units, adding another line to the graph, or changing the date correctly refreshed the graph.
- Added a default setting that displayed data upon loading the website.
- Design choice was made to only generate and store the contents of the CSV if the download button was selected.
- Automatic data thinning algorithm: to find the optimum amount of points, we ran some benchmark tests and determined 7500 data points as the maximum for peak performance (see Figure 2).

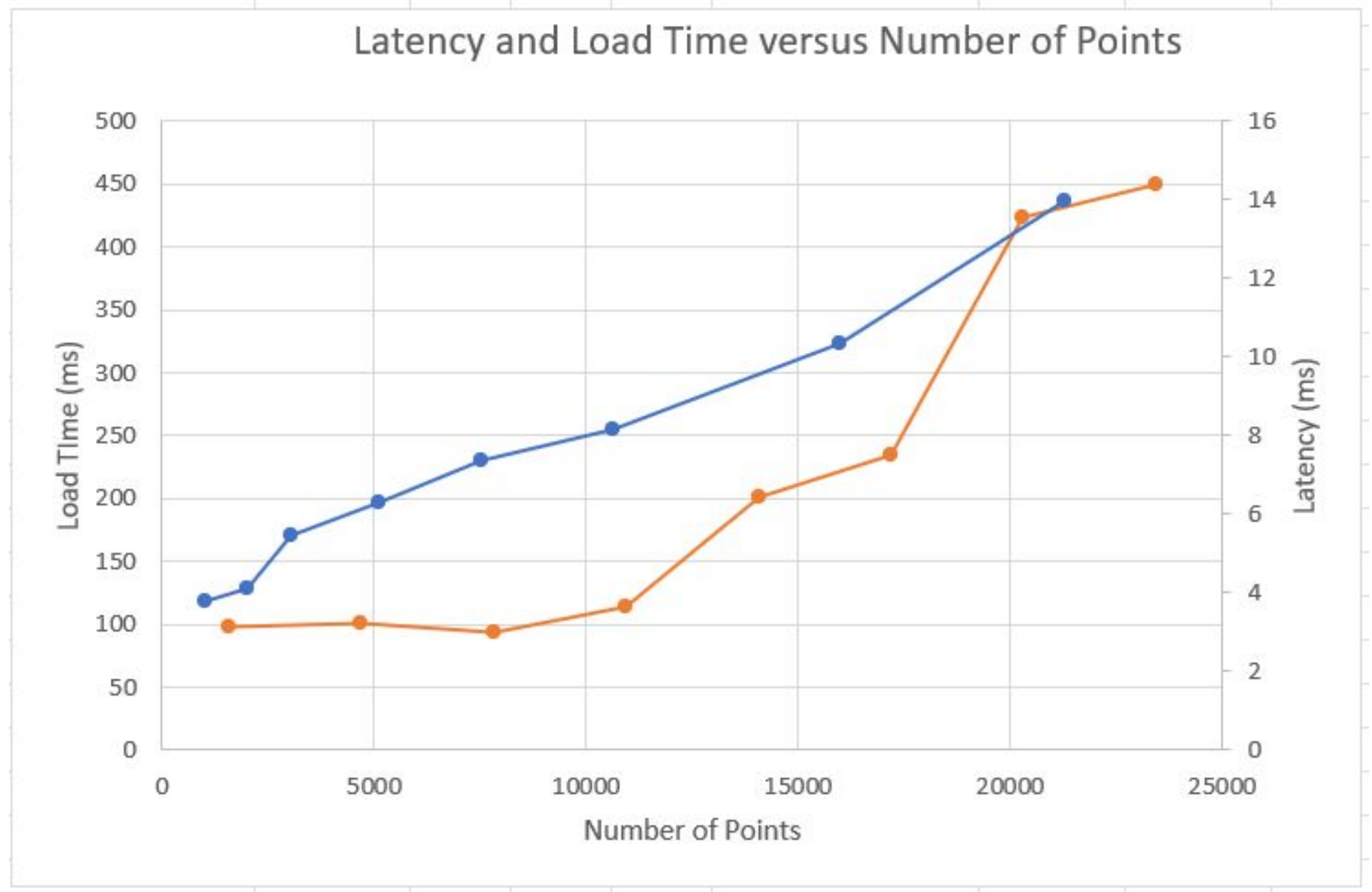

Figure 2. Graph showing how number of points can affect the loading time (blue) and latency (orange). Load time had a linear trend while latency had a sharper increase past 7500 data points.

Future Directions

- Filling the large gaps in the data through the use of machine learning. As seen in Figure 3, there are large sections of data that are missing. This was usually due to the network being down. These gaps could be inferred using ML.
- Regular site maintenance and sensor calibration. The sensors need to be calibrated once every two weeks, and the site should be cleaned at least that often.
- Finding a more efficient SQL query. To load a month of data, the query to the database takes 4.7 seconds. Of that time, the SQL query takes 4.5 seconds, which is a long time for a website. If there were a way to access this data faster, the graph would perform much faster.
- Adding additional apps or widgets that interact with other APIs like the National Weather Service to explore possible connections between other sets of data and the LEWAS data.

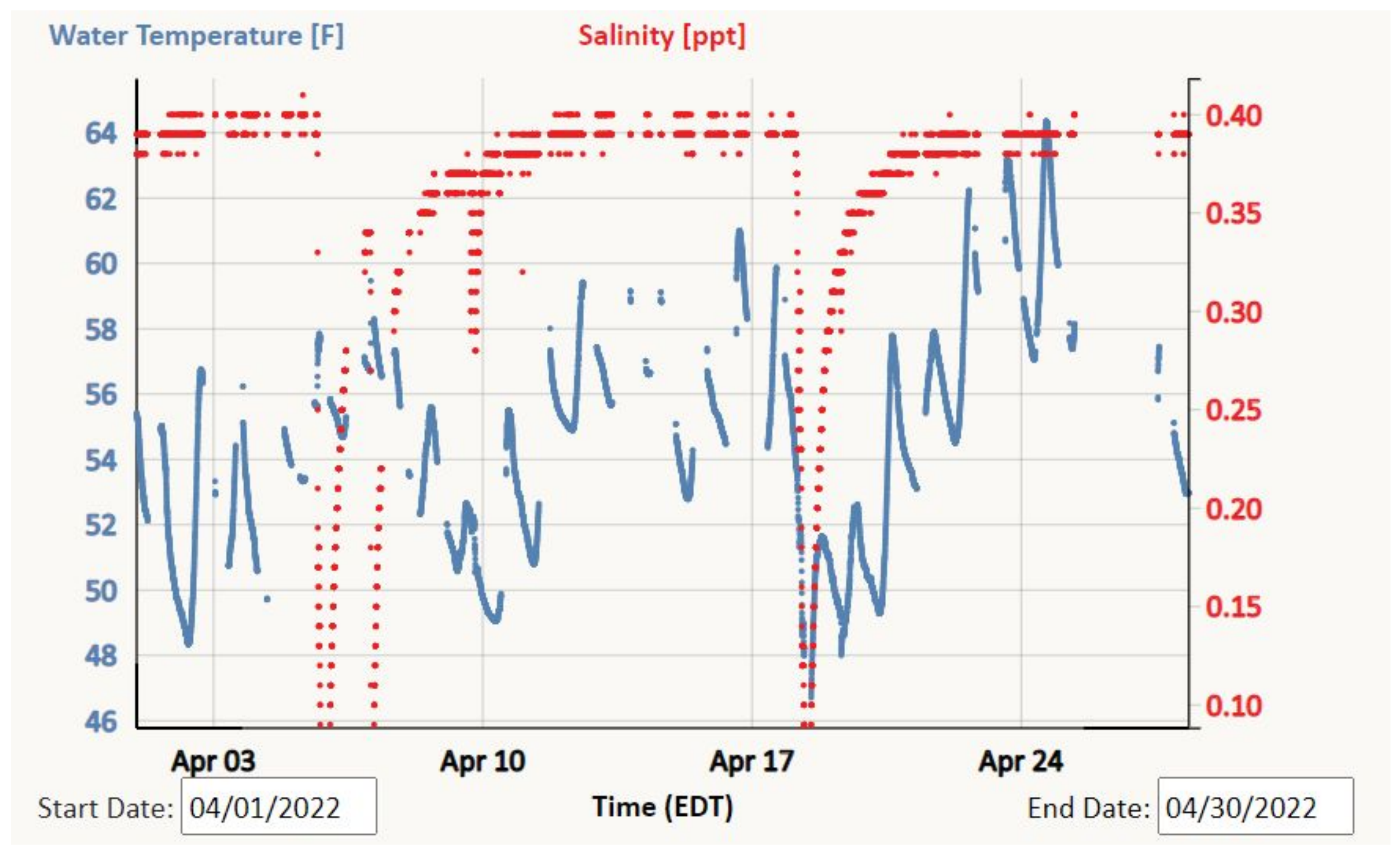

Figure 3. An example of a graph created on the OWLS website using water temperature and salinity data in April 2022. Gaps in the graph lines indicate times when the sensor system was down.

Acknowledgements: This research was supported in part by the National Science Foundation (NSF) under grant number 1922516. We would like to thank the LEWAS Lab for granting us access to their database, API, and resources. We would also like to thank Akshat Kothyari and Yunus Naseri for their continued help and support this summer.