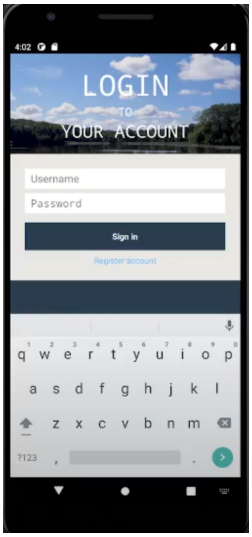
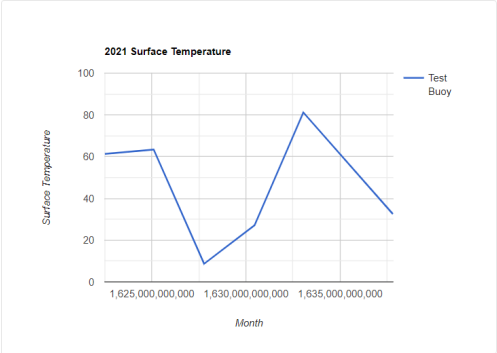
WWX Final Report Fall 2021

Jessica Avery and Douglas Woods

What's being handed off is a product with better security, ease of use, and utility than the product at the start of the quarter. The physical buoy has not changed, but more are being prepped to be given out to new members for hands-on learning and testing. Most changes have taken place on the Web-UI portion of the project, where data visualization has been a key focus. All components of the project (buoy, app, and web) can now communicate with each other.



Buoy	User	Timestamp	Surface Temprature	Surface Insolation
proto	kris	2021-06-01T00:00:00.000Z	61.357145230764345	0.3532231969475601
proto	kris	2021-07-01T00:00:00.000Z	63.403060716245285	0.3320691972549985
proto	kris	2021-08-01T00:00:00.000Z	8.695192099628857	0.07055963044950797
salish	kris	2021-09-01T00:00:00.000Z	27.108009304270908	0.5560320387949322
salish	kris	2021-10-01T00:00:00.000Z	81.18858769419823	0.7478881011714955
salish	kris	2021-11-25T01:27:43.000Z	32.51945787888955	0.6117238955306588



Implemented Features

Feature	Details	Date Completed
App Connection to User and Buoy Databases	Application now connected to database components	10/15/21
Data Visualization on Web Server	Graphs have been added to the web server that shows users the buoy data that's been collected.	11/05/21
Password Encryption	While buoy information is encrypted when sent to the app, the user database didn't have password encryption, so that's been added.	11/12/21

Features Opted Out Of

Feature	Details	Rationale for Not Implementing
Buoy GPS Component	We've discussed adding a GPS device to the buoy in order to get its coordinates.	Instead of adding more to the buoy, and therefore making it more expensive, we've chosen to add this functionality to the app, which can get the location data of the phone being used.

Maintenance

The following information is also available on the project repository, and should be kept up to date there.

Compiling the Project:

1. Run "npm install" on both web-ui and web-api root directories. Order does not matter.
2. Run "ng build" on web-ui root directory.
3. Run "docker build -t wwxs-ui ." in web-ui root directory.
4. Run "docker build -t wwxs-api ." in web-api root directory.
5. Run "docker compose up -d db" in documentation root directory.
6. Run "docker compose up -d api" in documentation root directory.
7. Run "docker compose up -d ui" in documentation root directory.

The steps above will set up the docker images for the web and database components. It's recommended to run through these steps if the code gets updated. If the code hasn't been updated, launching docker and confirming the images are running is all that needs to be done. The app, once it's either loaded onto a physical or virtual phone, is self explanatory. You can sign in or register an account, and view buoys and their information. Data can then be sent to the database to be viewed on the web. All database information can be viewed through the use of postgreSQL on the database docker image.

Bugs that still exist include:

- Graph not updating correctly when web page is refreshed
- Passwords need to be salted in database
- Buoy not sending converted measurements

Features to be Implemented

Things that need to get done	Details	Effort
Enhanced Password Encryption	Current password encryption is only implementing MD5 encryption. This should be improved with salting.	Low - Medium
Location Data	Location data collection is in the design and prototyping phase, but not yet implemented or functional	Medium
Optional Collection Notes	Similarly to location data, in the design and prototyping phase but not yet implemented or functional	Medium
Improved Data Graphs	Graphs need to be improved so that multiple data sets can be viewed on the same graph. Could also include comparing different buoys to each other.	Medium - High
Buoy Data Conversion	Buoy sensors have their data initially measured in voltage. That needs to be converted to the accurate measurement increment (i.e. Celsius for temperature)	Medium - High
Non Local Web Server	The web portion of this project up to this point has always been run locally. Setting up a server in the Cyber Range to host the website will be necessary when the project nears release.	Medium - High
Data Downloading	Currently our data is only available through the central PostgreSQL database, we need to add a method to download all/parts of the dataset in an easy to analyze format.	High

Enhanced Passwords - Salting involves adding another string to the password in order to deter dictionary attacks. This is relatively easy, just need to decide on the best way to approach implementing this.

Location Data - This will need to be done in the app portion of the project. Two options are to either add another button on the buoy options screen for location uploading, or to prompt the user with a dialog box that asks if they want to include buoy data, with yes or no options. Either way, when the user confirms they want to include location data, the app should then prompt them again, this time saying to be as close as possible to the buoy for accuracy, with a button to confirm they are close. Then, before sending the coordinate data, ask the user for a location name (i.e. Bellingham Bay), and upload that as well. The database will need to be adjusted to include this information.

Optional Collection Notes - This will also be done in the app portion of the project. When uploading data, there should be an optional space to add comments on the data being uploaded, in case of any unusual circumstances worth noting. The database will need to be adjusted to include these notes.

Improved Graphs - In order to be able to compare different data sets, our graphs will need to be able to display multiple data sets. This can be messy if all done on the same graph, so multiple graphs can help alleviate that clutter, especially if comparing multiple buoys. Some graph work is already done on the web-ui repository, just build off of that and add that functionality (easier said than done!).

Buoy Data Conversion - This is high up on the todo list for the project, and is already being worked on. The range of possible values needs to be established, as the voltage given off is also in a limited range (0-3.3V). Work already exists to convert this to an integer (0-4095), but depending on the sensor, these numbers should map to proper units. Research needs to be done in numerical conversions, and there may already be possible documentation for how to do it for each sensor available from the sensor's vendor/manufacturer.

Non Local Web Server - In order for this project to be useful to others, users will need to be able to both upload buoy data and view said data on the website portion of the project, which up to this point has only been ran and hosted locally. The Cyber Range will be able to accommodate the website, which will be very helpful to the developers. Developers will just need to acquaint themselves with the Cyber Range, and become familiar with using it to host the website and updating it whenever features are implemented.

Data Downloading - For researchers to make use of our data, there needs to be a way to download it to run it through analysis scripts. Ideally, we should be able to select only the data needed and download it as CSV, JSON, etc. No progress has been made on selecting, converting, or downloading the data yet, so the final form of this is to be determined by future design documentation.

Test Results

Much of our work this quarter has been focused on UI, so testing has primarily been done manually by visual inspection. There is an existing testing suite on our repository, but it is out of date, and only for a specific portion of our project (web). We highly recommend that tests are improved and more broadly implemented. This is further accentuated considering the work that needs to be done for the buoy portion of the project. A transition to test driven development would help developers figure out what goals they need to achieve.

Once the product is closer to a final release, more manual testing will be needed. Alpha testing the product will allow developers to see how the product performs in real world scenarios, and will likely expose issues and bugs that will need to be addressed. For example, previous buoy designs were prone to leaking, which can destroy the buoy components.