Florida International University

Knight Foundation School of Computing and Information Sciences

Software Engineering Focus

Final Deliverable

Project Title: Biscayne Bay Quality of Underwater Data Analytics (BBQUDA)

Team Members:

Capstone II: Catherine Angelini, Meleik Hyman, Adrian Perez, John Quitto-Graham, Muddasir Riaz

Capstone I: Damani Cephas, Brian Figueroa, Gersan Landero, Faran Majeed, Steven Reynoso, Gabriel Albuquerque Ribeiro, Fabio Velasquez

Product Owner(s): Gregory Reis

Mentor(s): Gregory Reis

Instructor: Masoud Sadjadi

Abstract

The BBQUDA web app is a service that allows users who are conducting research on the environmental conditions in the Biscayne Bay area to upload their mission data directly and view all of the corresponding data points along with the trajectory of the mission. The tool also provides additional resources such as GPS mapping needed to run the missions and useful information that researchers may require. Researchers are able to utilize a heatmap tool to view broad areas of water quality based on their missions. Researchers also have access to a developer API and can access crowd-sourced water quality data from around the world.

Table of Contents

Introduction

Current System

The current system BBQUDA system is in its 1st phase of development. The webapp currently allows users to make a general account where they can upload, store, and visualize mission datasets all in one place after logging in with an account.

Mission data is stored as CSV files on media storage. Mission data can be visualized on a map as trails. Each mission is also accompanied by general summary statistics.

Users can create custom trails and generate coordinates in CSV files for later robotic missions. These generated coordinates are not currently compatible with any robotic operating system and the tool is foundational in nature.

Purpose of New System

The purpose of the 2nd iteration of the BBQUDA web app is to provide a more fleshed out resource for researchers to conduct water quality surveys via autonomous vehicles. Ultimately, this 2nd iteration aims to improve the overall user experience, introduce user roles, and provide more useful data visualizations for users.

The goal of BBQUDA as a whole is to be a crowd-sourced data pipeline for water quality data in the Biscayne Bay or any other water resource with its own niche aspects of water quality. Moreover, with the improved UI/UX, more accessible and functional database, research API, and advanced, easy-to-use data visualization software, BBQUDA aspires to be the go-to place for oceanic researchers around the world.

User Stories

Implemented User Stories

Story ID	Story Title
001	As a developer, I want to choose a tech stack by fully understanding the requirements and finding what would work best to fulfill them.
002	As a developer, I want to set up a Github repository so I can version control along the development process and have effective collaboration among the team
003	As a developer, I want to familiarize myself and set up all dependencies needed for Django
004	As a developer I want to be familiarized with the current system and its shortcomings
005	As a developer I want to design the UI elements for the different pages of the web app
006	As a developer, I want to collect real life robot data like users would
007	As a developer, I want to increase the security of the log-in and auth functions
008	As a developer I want to develop a new database that can collect uploaded user information
009	As a developer, I want to archive data stored in the database that has been untouched for 5+ years.
010	As a developer I want to create kriging implementations to interpolate geospatial data
011	As a developer, I want to develop user roles to control how users can interact with uploaded data
012	As a developer, I want to create an API that is secured by a tokenization process

013	As a developer I want to refactor the existing UI into a more streamlined look and flow
014	As a user, I want to be able to interactively draw heatmaps for my mission data
015	As a user, I want to be able to change between what missions and parameters I am drawing heatmaps for
016	As a user I want to have varying sizes for my heatmaps

017	As a developer I want to separate users into Admins, Researchers, and General roles
018	As a developer/user I want to have a page dedicated to user visualization
019	As a user, I want to be able to easily contact the admins of BBQUDA
020	As a developer, I want to begin the process of creating a Firebase hosted database
021	As a developer I want to determine the permissions of each user role

Suggested User Stories

S-01	As a developer, I want the web app to be fully deployed on Firebase and Heroku
S-02	As a user, I want to be able to see specific data points that were interpolated along the heatmap

S-03	As a user, I want to be able to sign-up for the API to gain access to crowdsourced data
S-04	As a user, I want to be able have inferences made and suggested to me regarding the water quality data I provided to the web app

Project Plan

This section describes the planning that went into the realization of this project. This project incorporated the agile development techniques and as such required the sprints to be planned. These sprint plannings are detailed in the section. This section also describes the components, both software and hardware, chosen for this project.

Hardware and Software Resources

Hardware:

Development devices:

	Desktop PC	**Macbook Pro (Mid 2015)**	**Desktop PC**
Processor	3.2 GHz AMD Ryzen 1600	2.2 GHz Quad-Core Intel Core i7	3.4 GHz AMD Ryzen 2600
Capacity	500GB SSD	250 GB Flash Storage	500GB SSD
Memory	16 GB RAM	16 GB 1600 MHz DDR3	16 GB RAM

Autonomous Vehicles:

- EcoMapper Iver3 EP Open System UUV
- Clearpath Heron

Software:

- Windows 10
- Visual Studio Code
- Django
- Python
- Javascript
- HTML5
- CSS
- jQuery
- SQLite
- Github
- Vectormap
- Google Firebase

Sprints Plan

- Sprint 1
    - User Story 001
        - Review the data collected from a robot a group of students built
    - User Story 002
        - Github repository
    - User Story 003
        - Research on whether to migrate to Flask or stay with Django
    - User Story 004
        - Go over the current repository and make sure it works on everyone's system
        - Note down any problem areas or bugs
        - Note down any improvements that have to be made, such as improving login system, improving UI, etc.
    - User Story 006: Conduct missions to collect more data
- Sprint 2
    - User Story 005: UI/UX (1 Cap II, 2 Cap I)
        - Evaluate current UI
        - Inspiration for UI changes
        - Preliminary designs
    - User Story 007: API Team (2 Cap II, 2 Cap I)
        - Verifying current Log-in and Auth functions
    - User Story 010: Data Visualization (1 Cap II, 1 Cap I)
        - Publicly viewable data vis
        - Privately viewable data vis
        - Generating actual visuals (i.e. Heatmaps, interpolation)
    - User Story 008: Databases (1 Cap II, 1 Cap 1)
        - Design DB for the data users collect

    - User Story 009
        - Ensure that the live data that is viewed can show up to 5 years
        - Create the ability to archive data past 5 years

- Sprint 3
    - User Story 017
        - Determine what can be viewed by each of the API roles (Admin, Researcher, General)
    - User Story 013: UI/UX (1 Cap II, 2 Cap I)
        - Refactor the website to start new designs
        - Preliminary re-designs for
            - Homepage
            - login/registration
            - Data visualization
        - Work with API and backend team for functionality
        - Create CI/CD pipeline for seamless integration
    - User Story 012: API Team (2 Cap II, 2 Cap I)
        - API Key creation for access to data
        - Generating the appropriate API calls

- User Story 010: Data Visualization (1 Cap II, 1 Cap I)
  - Kriging for real data values

- User Story 011: Databases (1 Cap II, 1 Cap 1)
  - Determine what can be viewed by each of the API roles (Admin, Researcher, General)

- Sprint 4
  - User Story 013: UI/UX (1 Cap II, 2 Cap I)
    - With the chosen new UI design, have:
      - login/registration page
      - Data Visualization page
      - Team Page
      - Contact us Page
    - Working with API team, have
      - CSV/log file upload page
      - Other functions only allowed for approved researchers

  - User Story 012: API Team (2 Cap II, 2 Cap I)
    - API Key creation for access to data
    - Generating the appropriate API calls
    - Modify bbquda site API so that CSV upload can be put into a bigger database for public access
    - Start new API creation to access larger DB

  - User Story 014: Data Visualization (1 Cap II, 1 Cap I)
    - Map view for different interpolated layers

  - User Story 017: Databases (1 Cap II, 1 Cap 1)
    - Design DB for the data users collect
    - Determine what can be viewed by each of the API roles (Admin, Researcher, General)
    - Ensure that the live data that is viewed can show up to 5 years
    - Create the ability to archive data past 5 years
- Sprint 5
  - User Story 013: UI/UX (1 Cap II, 2 Cap I)
    - With the chosen new UI design, have:
      - login/registration page
      - Data Visualization page
      - Team Page
      - Contact us Page
    - Working with API team, have
      - CSV/log file upload page
      - Other functions only allowed for approved researchers
  - User Story 011/012: API Team (2 Cap II, 2 Cap I)
    - API Key creation for access to data
    - Generating the appropriate API calls
    - Modify bbquda site API so that CSV upload can be put into a bigger database for public access
    - Start new API creation to access larger DB

  - User Story 015/016: Data Visualization (1 Cap II, 1 Cap I)
    - Map view for different parameters
      - Area selection
      - Loading heatmap

- Mission Selection
- Connect to HTML to Flask server

- User Story 020: Databases (1 Cap II, 1 Cap 1)
  - Design DB for the data users collect
  - Determine what can be viewed by each of the API roles (Admin, Researcher, General)
  - Ensure that the live data that is viewed can show up to 5 years
  - Create the ability to archive data past 5 years
- Sprint 6
  - Presentation Slides
  - Project Documentation/Research Paper
  - Videos
  - Posters
  - User Story 013/018/019: UI/UX (1 Cap II, 2 Cap I)
    - With the chosen new UI design, have:
      - login/registration page
      - Data Visualization page
      - Team Page
      - Contact us Page
    - Working with API team, have
      - CSV/log file upload page
      - Other functions only allowed for approved researchers
  - User Story 011/012: API Team (2 Cap II, 2 Cap I)
    - Finish the Front end part for API
    - Start working on the final deliverables
  - User Story 015: Data Visualization (1 Cap II, 1 Cap I)
    - General bug fixes
      - Button selection names
      - Turbid+ NTU getting passed through URL
      - UI bugs
  - User Story 021: Databases (1 Cap II, 1 Cap 1)
    - Design DB for the data users collect
    - Determine what can be viewed by each of the API roles (Admin, Researcher, General)
    - Ensure that the live data that is viewed can show up to 5 years
    - Create the ability to archive data past 5 years
- Sprint 7
  - Presentation Slides
  - Project Documentation/Research Paper
  - Videos
  - Posters

System Design

This section contains information on the design decisions that went into this project. The architecture patterns are outlined and explained. The entire system is shown in a package diagram and the subsystems are explained. Finally, the design patterns used in the project are discussed.

Architectural Patterns

For our architecture, we had a choice of various services that we could build our architecture on. Our decision came down to our need for users and a unique database design. Also choosing something that all members were familiar with or had knowledge of would allow the development process to be greatly expedited. Using the type of MVC architecture allowed us to handle all types of interactions our users could potentially have with the site.

System and Subsystem Decomposition

We chose to build on Django 3.1.2 since it is built on Python and comes with support for many features which we deemed critical. Django's template engine would allow for ease of creating and displaying website pages to the frontend. Django's built in user and authentication system was also a great time saver in this project since they were all features we would have to manually implement otherwise. Using Django as our backend and media storage would allow for quick expansion and ease of manipulation with CSV files.

In order to have dynamic maps, we are using Leaflet and jQuery in order to pull the proper information from the files stored in the database. Changes were pushed to Github in order to have different developers work on various components at the same time and have appropriate version control.

Deployment Diagram

Design Patterns

The model, view, controller architecture is commonly used for applications because it divides program logic into 3 interconnected parts. Django does a good job at providing boilerplate for many of the model-view-controller functionality as well as making it simple for expansion. Our Model component consists of two separate data handling techniques, mainly a database to store user information and a Django media folder to hold user uploaded CSV mission files. Our views are handled through Django's template engine and Bootstrap.

System Validation

Because of our desire to create the most streamlined interface for our web application, the testing of it was easy enough to complete manually. The site is only usable for users who have registered and are logged in, so we only have to worry about one type of user. Test cases include trying to upload files of the wrong type in the upload sections of the site. We also attempted to have users attempt to register with login info that was already in use and making sure it would not work. The functionality of every feature was also verified, such as the proper display of the maps in various pages on the site.

Developer API with Django REST

In order to encourage environmental research in the Biscayne Bay area, researchers and developers need to share collected field data in order to provide diverse and unique insights on the mission metrics. BBQUDA seeks to enable developers and data scientists to upload and access mission data using underwater robotics to draw unique insights on the dynamics of the bay's water quality. We use the Django REST framework along with the Django REST Authentication package to securely verify GET and POST requests on the datasets. In the website we set up an authentication scheme on a per-view and per-viewset basis, like the documentation implies. We use TokenAuthentication scheme to leverage token-based HTTP Authentication scheme. Every registered user has an assigned token attached to their account and this token will grant them access to the API.

Token Authentication

TokenAutentication is a scheme defined within the Django REST framework and allows us to authenticate users using their uniquely assigned tokens. TokenAuthentication is ideal for client-server setups like the locally run setup this semester's iteration of BBQUDA features. We configured our authentication classes to include the following in the "INSTALLED_APPS" setting of the Django project:

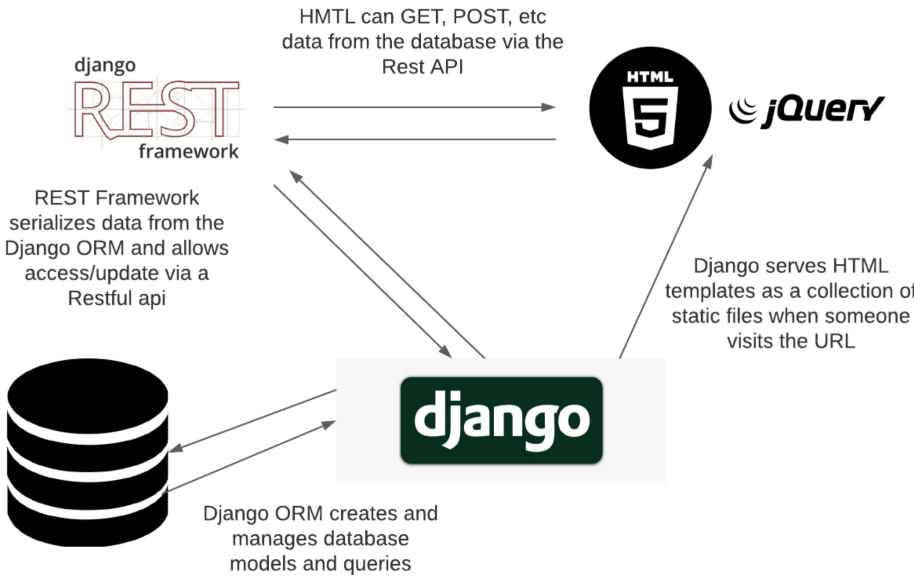
-TokenAuthentication
-rest_framework.authtoken

We then migrate our database, since the authtoken class of Django REST framework provides database migrations.

Developer Access

The goal of this API is to enable researchers to access not only their uploaded mission stats, but the stats of every other researcher that conducts missions in the Biscayne Bay. Therefore, the API would act as a way for developers to access the insights from other researchers in the field. To get the results of the API, users must first register into their account in order to be authenticated with a token. From there users may receive their token using the drf_create_token function while passing in their email for credentials. From there, users will be able to copy their token and append it to the API endpoint "bbqudasite_api" in the url in order to access the data. The api generates a JSON response of the repository of datasets uploaded by researchers where the key is the name of the dataset and the value is the dataset information itself.

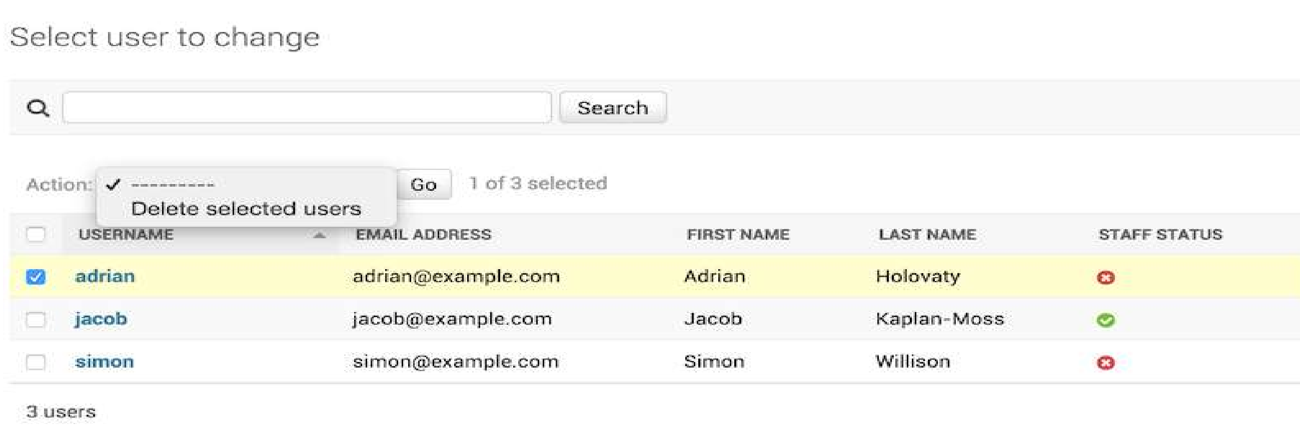
The technology stack we have chosen played an important role in the development of the API. As indicated below in the diagram, Django communicates with it's ORM for database queries, allowing us to aggregate the uploaded mission datasets into one dictionary and return it as a JSON. The REST framework then accesses the ORM with a RESTful API. We use GET requests from the database and display that data to the HTML frontend.



Developing User Groups and Permissions

The goal of the Data Base team was to create a user database that receives data from the user creation form/user change form, and stores this data in a secure and effective way. Before registering a new user into the database a security check must be done to ensure this user does not already exist with the user's database as either custom or superusers. The security check was done on 1 major parameter, the Email field, ensuring that only one email can belong to each user to reduce duplicated and possible attacks on our site.

The users can be manually selected for groups that have special permissions to access particular aspects/control of the mission data within the other databases. A group is a collection of privileges that can be granted to one or more users or other roles. Groups help you grant and manage sets of privileges for various categories of users, rather than grant those privileges to each user individually. In order to reduce possible data leaks, a function was made to delete all logged mission data over 5 years old. As part of future proofing the site and data access, a firestone account was created along with a schema to better help store future data on a cloud service.



Data Visualization Tools

One of BBQUDA's main goals is to provide accurate data visualizations for oceanic researchers to solve real world problems in the waters of Florida (and by extension any body of water). This creates a need from our team to offer fast, lightweight solutions to the common data visualization needs of our users. Below we discuss the tools currently developed for utilization in any and all water quality assurance missions. The design philosophy is the same for each tool: it should be easy to use, flexible, and above all accurate to the standards of the researchers depending on it.

The following dependencies and tools were used in or production process:

- Dependencies:
  - Leaflet
  - WebGL Heatmap
- Tools:
  - Mission Data Statistics
  - Kriging Heatmap Tool
  - Mission Trail Generation

Leaflet

Leaflet is an open-source Javascript library designed to handle integrated map displays and coordinate based data visualization. This library was chosen for it's lightweight nature, out-of-box tools, and ease of integration with Python and Django. Similarly, there is a decently sized community around the product that provides open-source plugins for more complex data visualizations than, say, simply placing markers on a map. This community driven work was greatly helpful in developing the heat maps and trail generator described further in this document.

WebGL Heatmap

One of the most important features offered by BBQUDA is its fast, lightweight heatmap approximations for water data across areas of interest. These heatmaps need to generate thousands of points with a high degree of accuracy. In order to handle the displaying of the heatmap data handled by the Python backend, we utilized the community driven WebGL Heatmap plug-in to showcase the results of the kriging calculations for each mission uploaded to the system. The plug-in was chosen for its simplicity and ability to generate many points on-screen without much hassle and overhead. Other competing leaflet plug-ins were far too heavy and complex and given that the actual calculations are done in the backend side, we chose the most efficient tool for our needs.

Mission Data Statistics

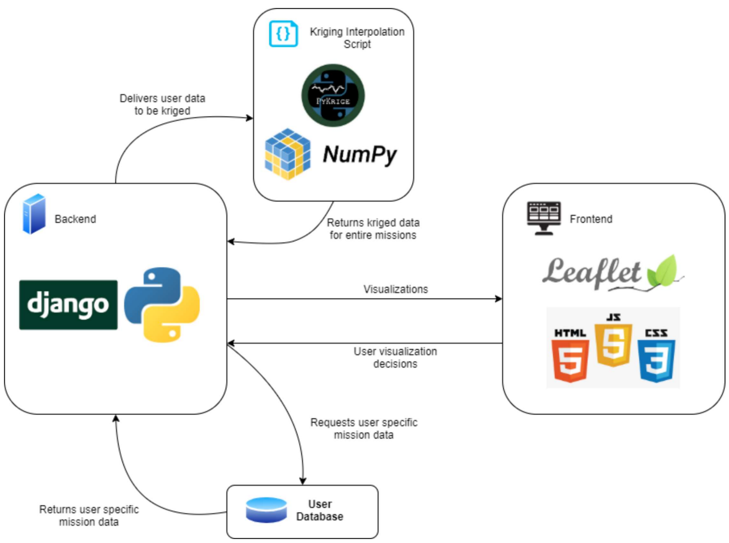
A major part of the pipeline in BBQUDA is the ability for researchers to upload their completed missions on the water to our database for ease of access. In doing this, researchers can gain access to a variety of statistics regarding each mission making it far easier to quickly iterate on the data they collect. Normally, an oceanic researcher would have to develop their own scripts to handle much of the overhead that goes into analyzing mission data. BBQUDA handles this data analysis the moment the data is uploaded.

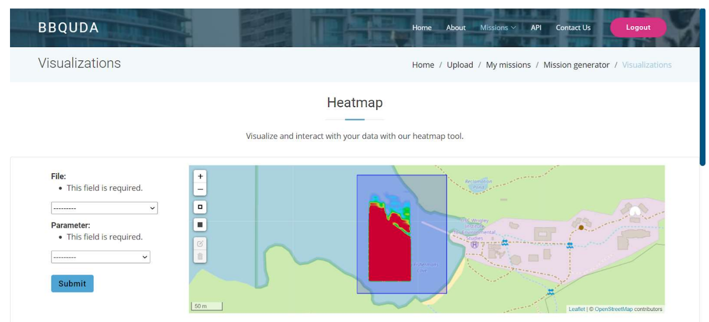
Researchers also have the ability to see these stats along each point of the mission they conducted. This means detecting abnormal behavior along the mission path is easier than before. Using a python back-end for the statistical analysis and Leaflet for the map visuals, the path from mission data to actionable results is quick and consolidated to the BBQUDA web application.

Kriging Heatmap Tool

When a mission is conducted, there are often many areas that the robot misses due to size and battery constraints. This leaves the data scientists have to work with sparse and makes it difficult to know general trends in a given area. Often, researchers will utilize what is called kriging interpolation to estimate the gaps in a given area. This process, especially for those untrained in the particular maths, can be quite difficult. BBQUDA offers a built-in solution to make this simpler and more efficient.

Using PyKrige in the Python backend, we are able to take the data directly uploaded to the BBQUDA web application and perform the kriging analysis. Likewise, users are then able to use the Heatmap Tool to view specific areas of interest and display the heatmap points in said areas. The time from data collection to viewing the necessary heatmaps is now drastically reduced and consolidated neatly to one place. It is a future goal for the team to have these heatmaps available via the BBQUDA API for researchers to be able to implement them in their own works.





Mission Trail Generation

One of the largest hurdles for researchers is the ability to make missions for their data surveying tools. Many require separate and often clunky middleware in order to generate geo-coordinates for their robots to follow. BBQUDA allows for a universal mission generation feature where users can create and save the missions they want to make only through map interaction and clicking.

This feature is still in development and there are future goals of adapting this into workable ROS mission code. Several UI features will be developed to give researchers more control over their missions all consolidated to a single web app and tool.

Robotics Ecosystem

This project's ultimate goal is to assist the robotics research community in being able to conduct successful underwater missions. That being said, it is important that we take into consideration the actual usage of a robot when designing and building the BBQUDA application. Our team is unique in that this application serves to not just assist others, but to also assist our very own water quality endeavors. To accurately model our task and provide working data for the team to use, we conducted several missions with the EcoMapper underwater robot and utilized the development stack surrounding it. Likewise, we began work on ways of giving users and ourselves more ease of access for creating missions end-to-end. To do this we also made use of the Robot Operating System, ROS, and the Heron surface water vehicle.

Here we will discuss the use of the following in our work with the robots intended to be used in conjunction with BBQUDA:

- EcoMapper
- VectorMap
- QGIS
- ROS
- Field Work
- Data Collection

EcoMapper

The YSI EcoMapper AUV is our underwater vehicle of choice when it comes to data collection in the Biscayne Bay. The AUV comes equipped with several sensors such as YSI conductivity and temperature sensors, YSI depth sensor, blue-green algae, pH/ORP, and many more which were able to provide us with useful data in our missions. We chose this particular AUV for several reasons but primarily it is because of its ease of availability, prominence in the underwater robotics research community, and the support we have from the PO's prior work. Normally, to operate the EcoMapper, one would need to use VectorMap, its proprietary software for mission planning. Much of the information learned from using the EcoMapper was reflected in the design choices for this project.

VectorMap

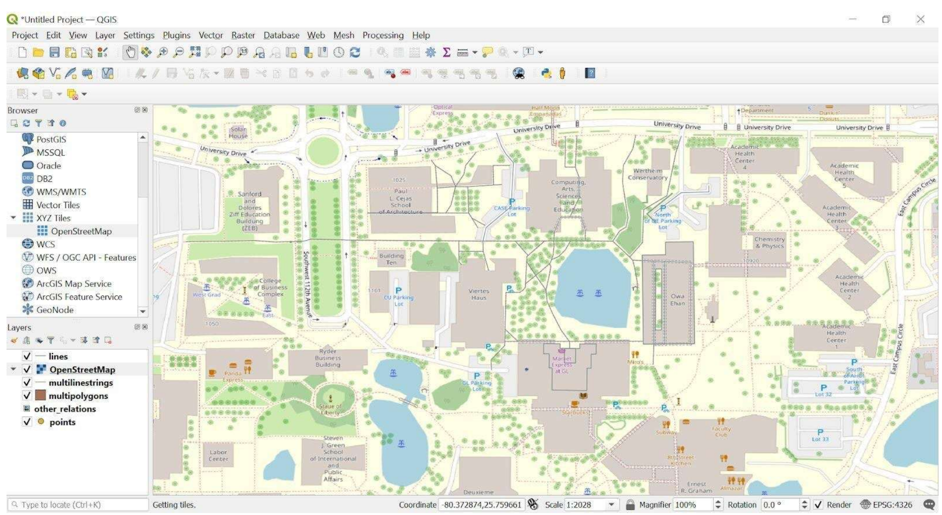
VectorMap is a proprietary software used by the EcoMapper to plan missions for data collection and water traversal. The software enables a user to generate waypoints along geo-coordinates for the robot to follow and perform desired tasks. This can include operations such as diving, sonar mapping, and active/passive data collection to name a few. This software, however, suffers from some key flaws.

Mainly, the software is simply in-accessible and clunky to use. Waypoint editing and geo-mapping prove to be a challenge through not only the user-interface but also in the actual conversion from waypoint movement to real life execution. Also not to mention that the system is a complete blackbox in nature. So if a researcher desires something more precise or autonomous, it is near impossible to do so.

For this semester, we chose to work with VectorMap as a means to produce intermediary results. We do, however, expect that our work with ROS will allow us to create an easier, more user friendly, and precise solution to not just the EcoMapper, but other water robotic solutions. Still, though VectorMap was hard for the team to use, we were able to conduct many successful missions with the program and provide important data to the team, the local government, and FIU.

QGIS

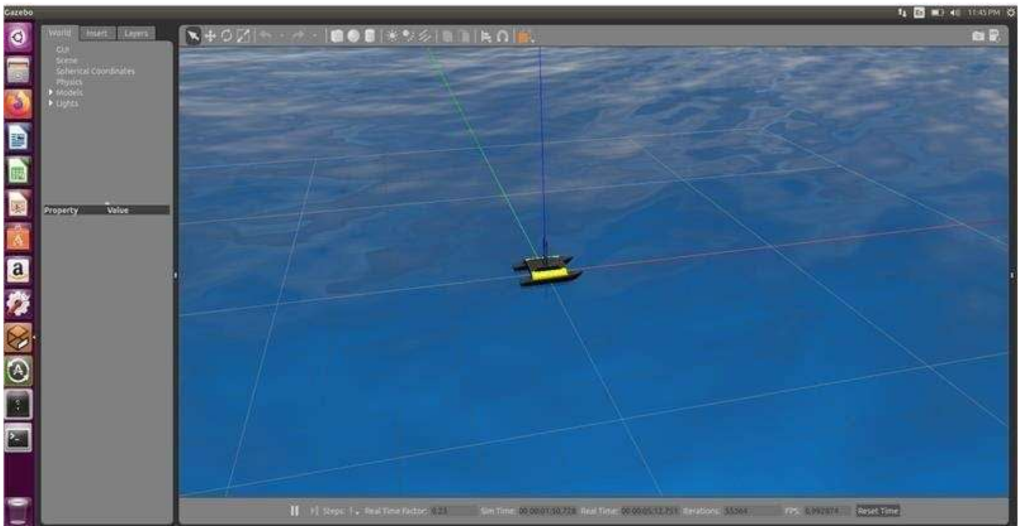
QGIS is a free geographic information system (GIS) software that allowed us to create and export a Tagged Image File Format (TIFF) file needed for VectorMap. In order to create the missions in VectorMap, a TIFF file was required of the area the EcoMapper will cover with coordinates of each part of the map embedded into it. A jpeg or png image would not be enough, as the missions we had to create for the EcoMapper work through the use of waypoints that have latitude and longitude coordinates. Our missions covered several areas such as FIU's Modesto A. Maidique (MMC) lake, a dockside location in FIU'S Biscayne Bay campus, Little River Pocket Park, etc. Hence, for every location, we would have to export a map from OpenStreetMap as an OSM file and import this file into QGIS as a vector layer to be able to export it as a TIFF file that can be used for VectorMap. As seen below, this is the overall map of the FIU's MMC lake in QGIS, which was created when we had to conduct a preliminary mission there.



ROS

Robot Operating System (ROS) is a robotic middleware that allows developers access to sets of useful libraries and packages that simplifies the development of robot behavior. As we faced a lot of challenges using VectorMap, a better option we encountered was to set up a ROS development environment for the Heron vehicle. While there is no existing support for the Ecomapper with ROS, we still are aiming to be able to collect water quality data, which the Heron will allow us to do.

We set up our environment by using an Ubuntu 16.04.7 LTS (Xenial Xerus) virtual environment and installing Gazebo, the Heron simulator, ROS Kinetic, and the uuv simulator package, which can be found in https://github.com/uuvsimulator/uuv_simulator. This is a simulation of the Heron vehicle in Gazebo, which can be controlled using ROS scripts.



Field Work

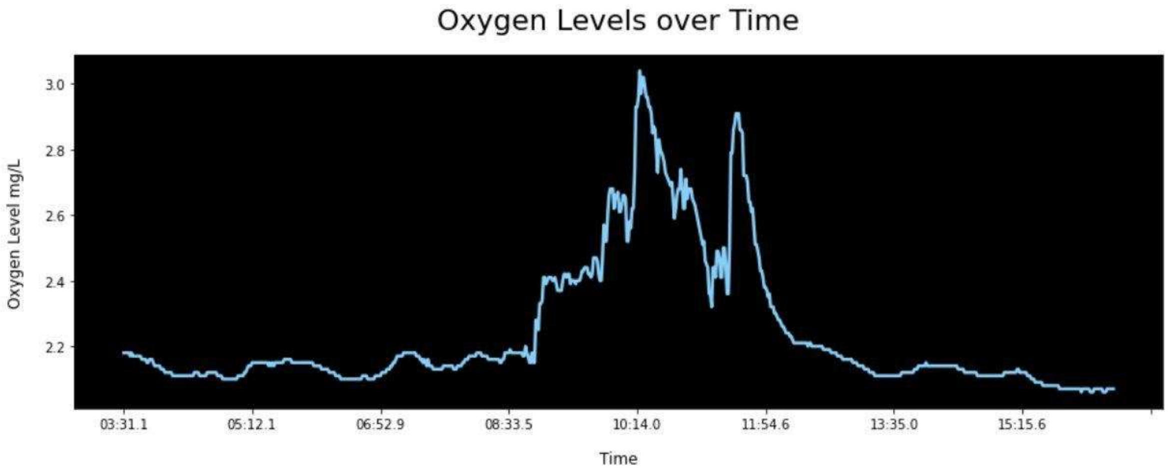
In order to collect preliminary data for the BBQUDA application as well as for the research team, we conducted several live missions with the YSI Ecomapper. Most of these missions were conducted in the Biscayne Bay as per our research and client goals. The data collected here was integral to development and testing of what the BBQUDA application should aim to help. These missions collected a series of data points ranging from temperature to pH levels.

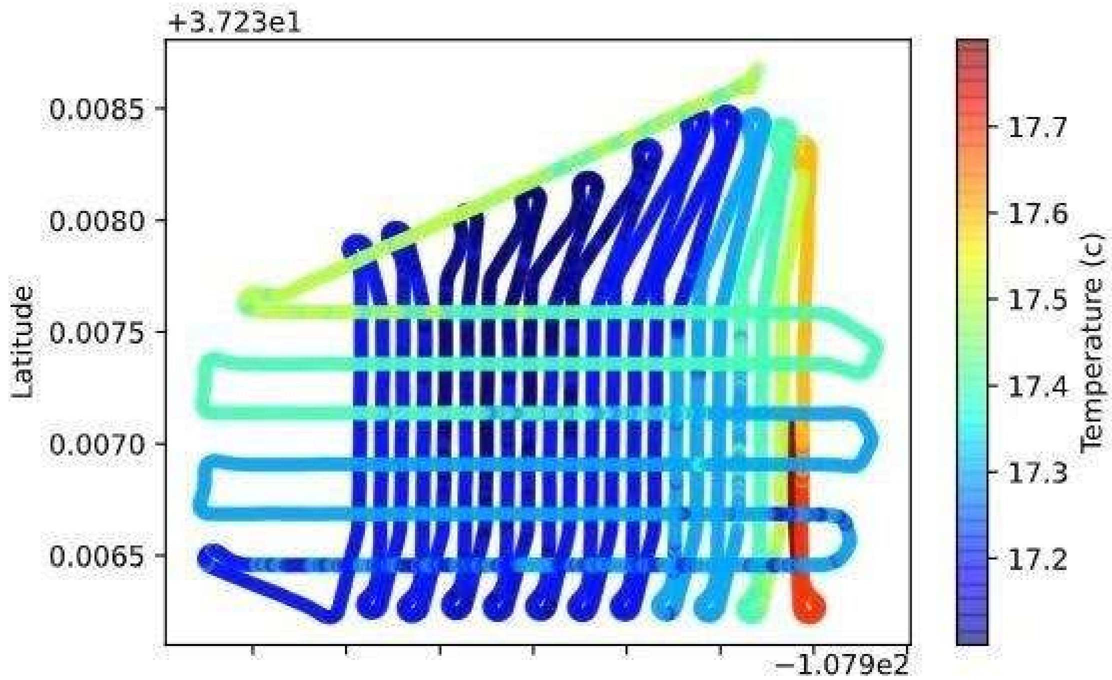
The field work also allowed us to identify flaws that exist with some of the current research methods when using underwater autonomous vehicles. We were also able to identify flaws in VectorMap and determine ways we could improve the BBQUDA application. This was one of the reasons why BBQUDA has implemented a waypoint mechanism to allow researchers to map out their research mission with a robot.

	Total Water Column (m)	Temperature (c)	pH	ODO mg/L	Salinity (ppt)	Turbid+ NTU	BGA-PC cells/mL
count	320	320	320	320	320	320	320
mean	4.0960	27.6028	7.8249	2.0867	2.8400	4.1328	-1,904.3250
std	0.7802	0.0166	0.0231	0.0683	0.3310	1.5204	1,371.9033
min	2.1800	27.6000	7.7600	1.9800	0.4900	0	-5751
25%	3.5700	27.6000	7.8100	2.0400	2.6600	3.4000	-2,748.2500
50%	4.5900	27.6000	7.8200	2.0900	2.9100	3.9000	-1898
75%	4.7300	27.6000	7.8400	2.1100	3.0500	4.5000	-1019
max	4.9700	27.7000	7.8900	2.6800	3.2200	13.7000	2034

Data Collection

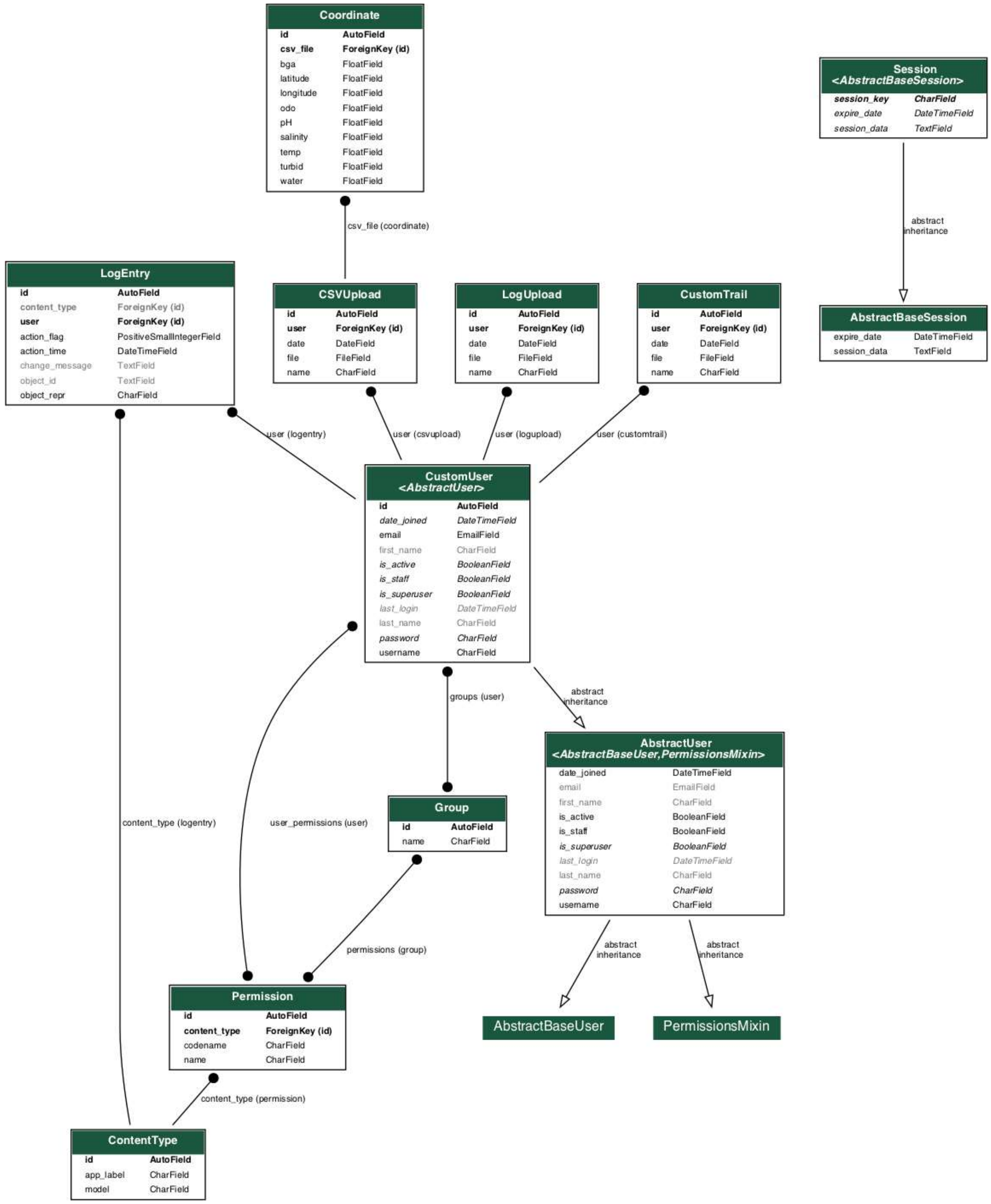
During each deployment of the EcoMapper, various types of data would be collected of quantitative water quality metrics such as salinity, pH levels and temperature for each second of the entire mission, as seen in the figure below. All of this data would then be saved as a csv file that could be visualized to see the oxygen levels over time, the change in pH, etc. While our focus this semester for Capstone I was on the data collection and understanding VectorMap and the EcoMapper, we were still able to generate scripts using Python and data visualization libraries that allowed us to apply statistical measures to this data and will help us for next semester, which is when we want to create more efficient and precise missions. These preliminary scripts will be improved upon and actually integrated into the BBQUDA system in the following iteration.


Oxygen Levels over Time
Oxygen Level mg/L
3.0
2.8
2.6
2.4
2.2
03:31.1
05:12.1
06:52.9
08:33.5
10:14.0
11:54.6
13:35.0
15:15.6
Time

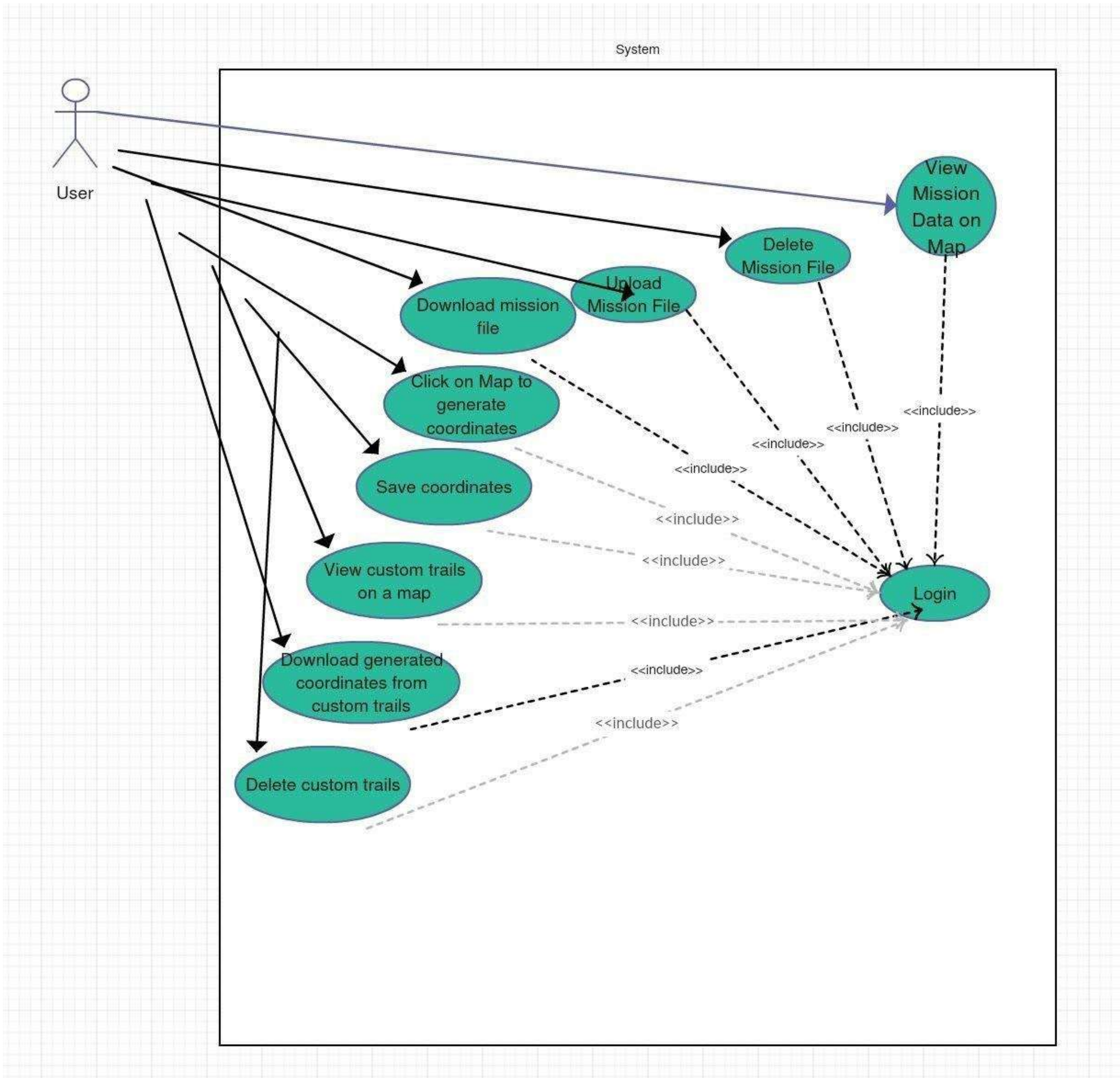

+3.723e1
0.0085
0.0080
0.0075
0.0070
0.0065
Latitude
−1.079e2
17.7
17.6
17.5
17.4
17.3
17.2
Temperature (c)

Appendix

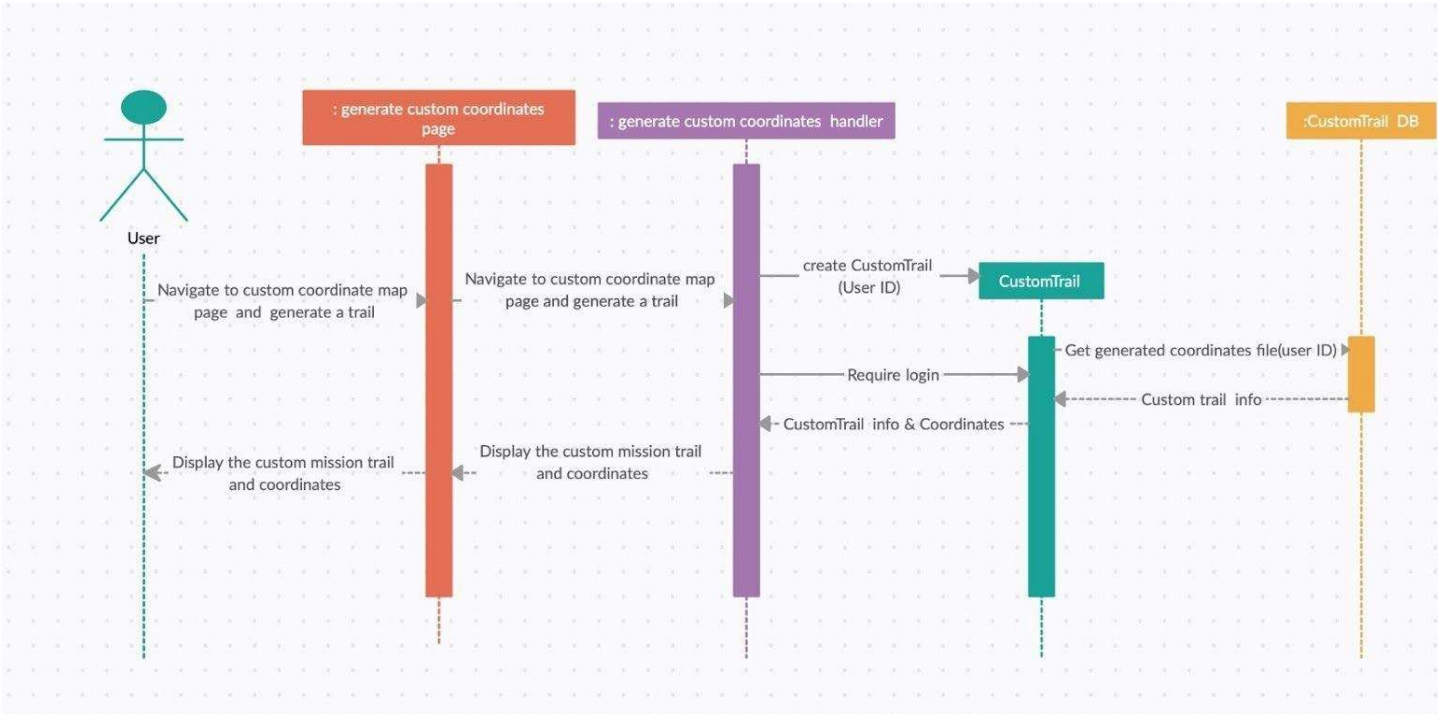
Appendix A - UML Diagrams



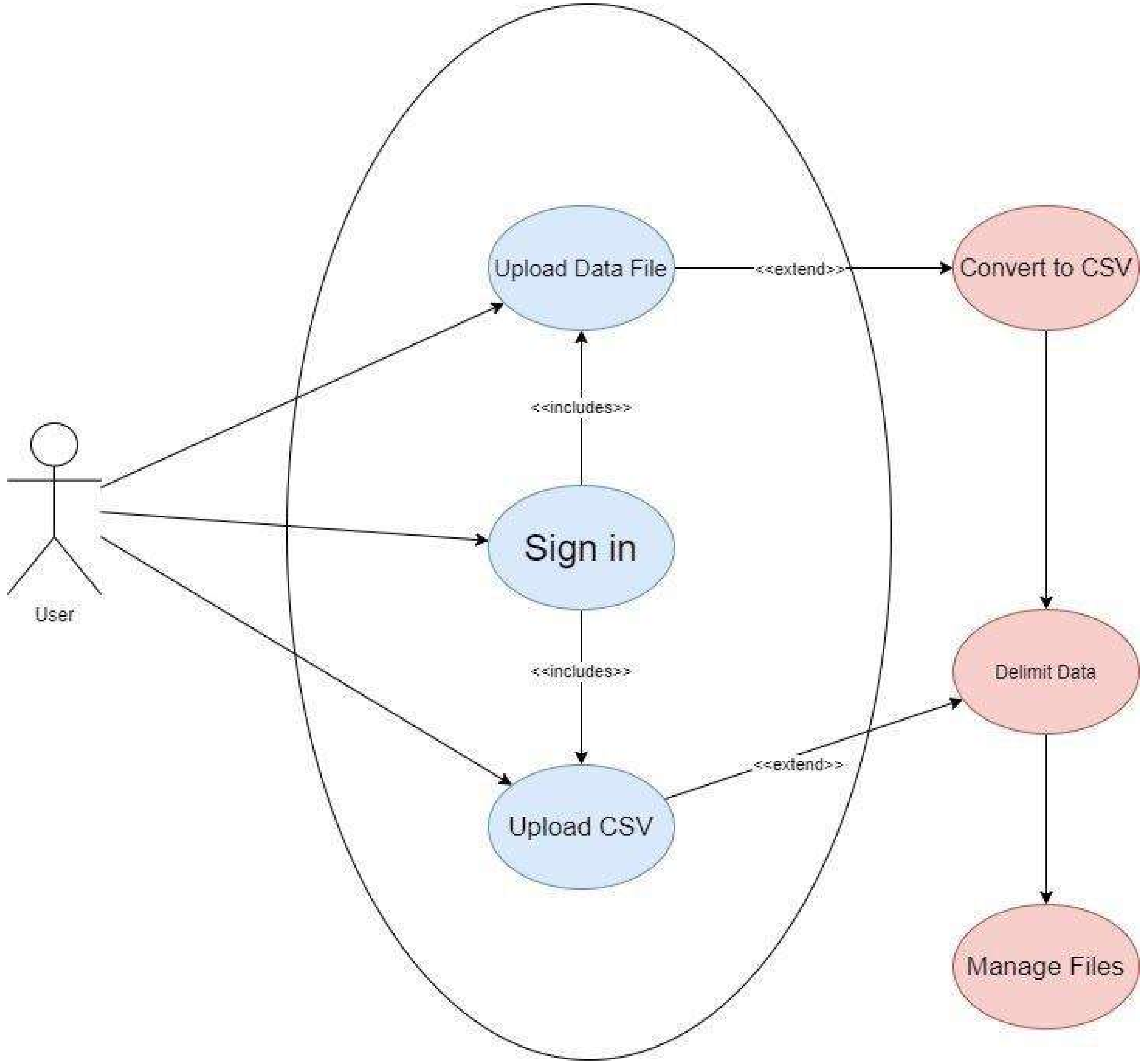
-Full Class Diagram (Ariel Omar Arias, Orson Meyreles, Leonardo Pantaleon)



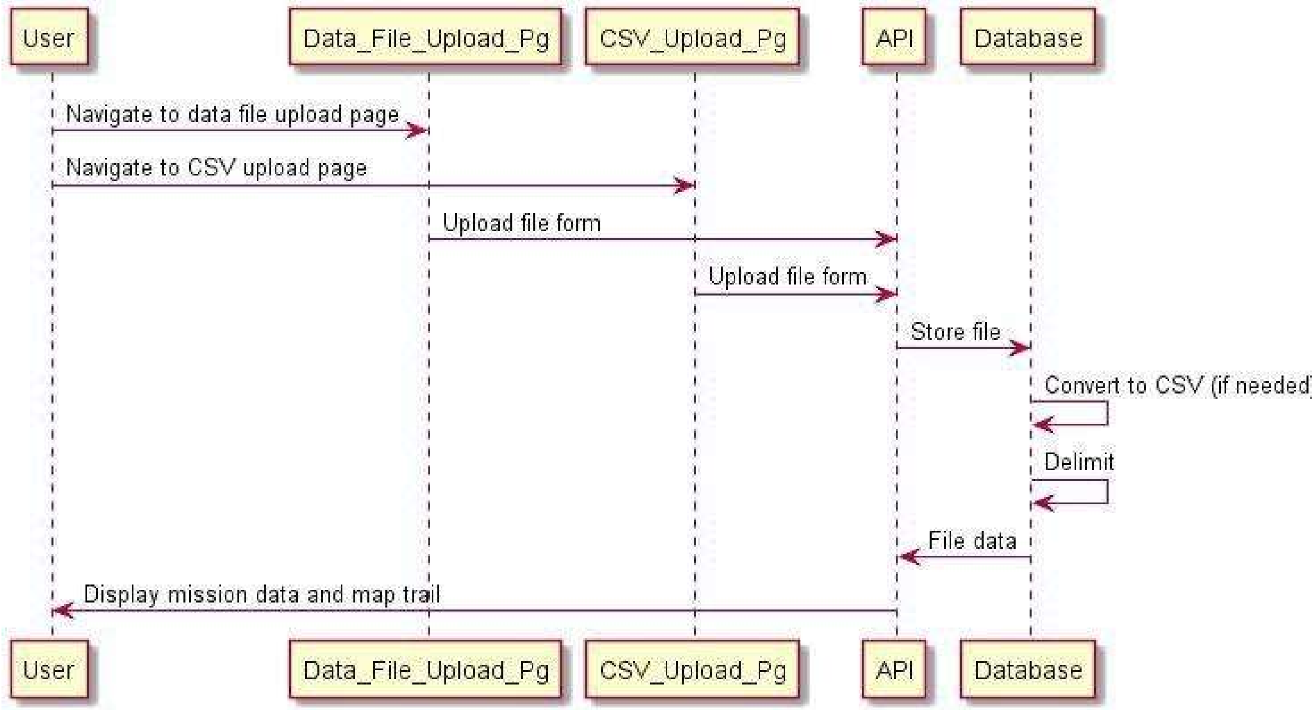
-System Use Case Diagram (Ariel Omar Arias)



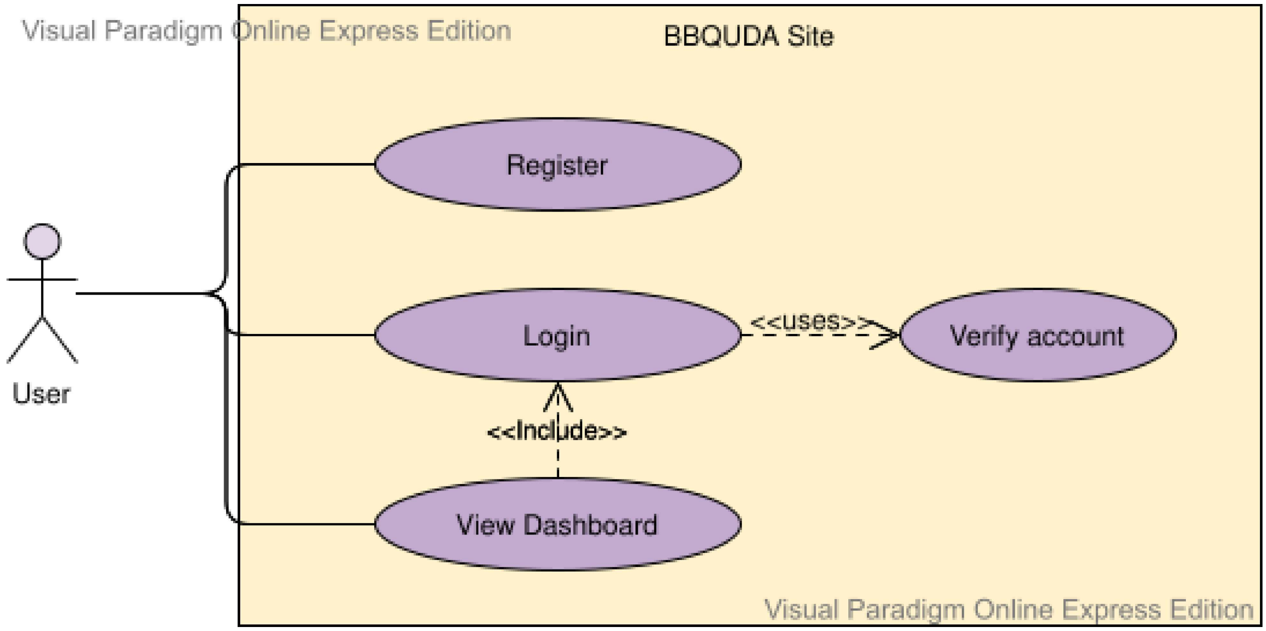
-Map Coordinate Generation Sequence Diagram (Ariel Omar Arias)



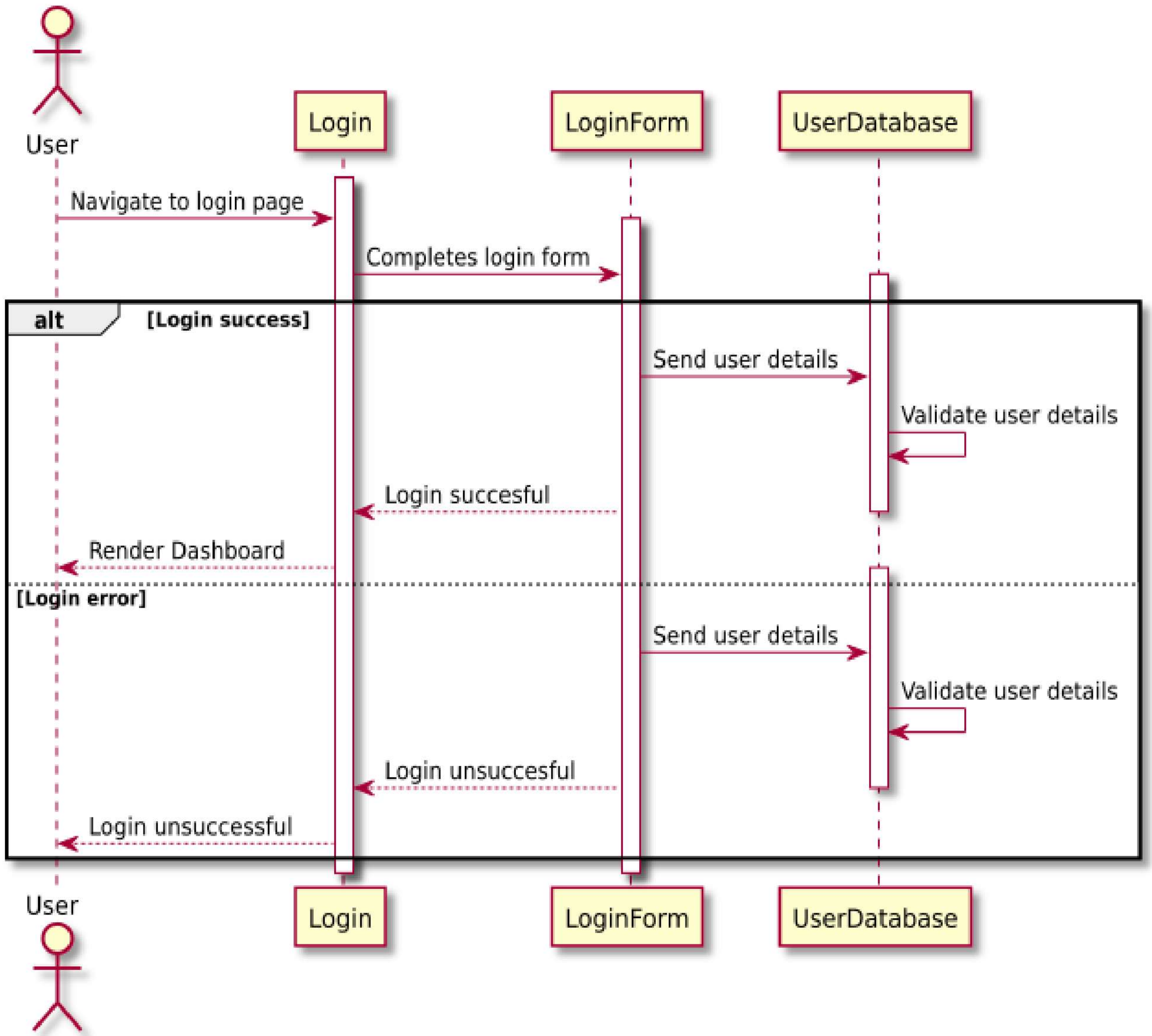
-Mission Data Upload Use Case Diagram (Orson Meyreles)



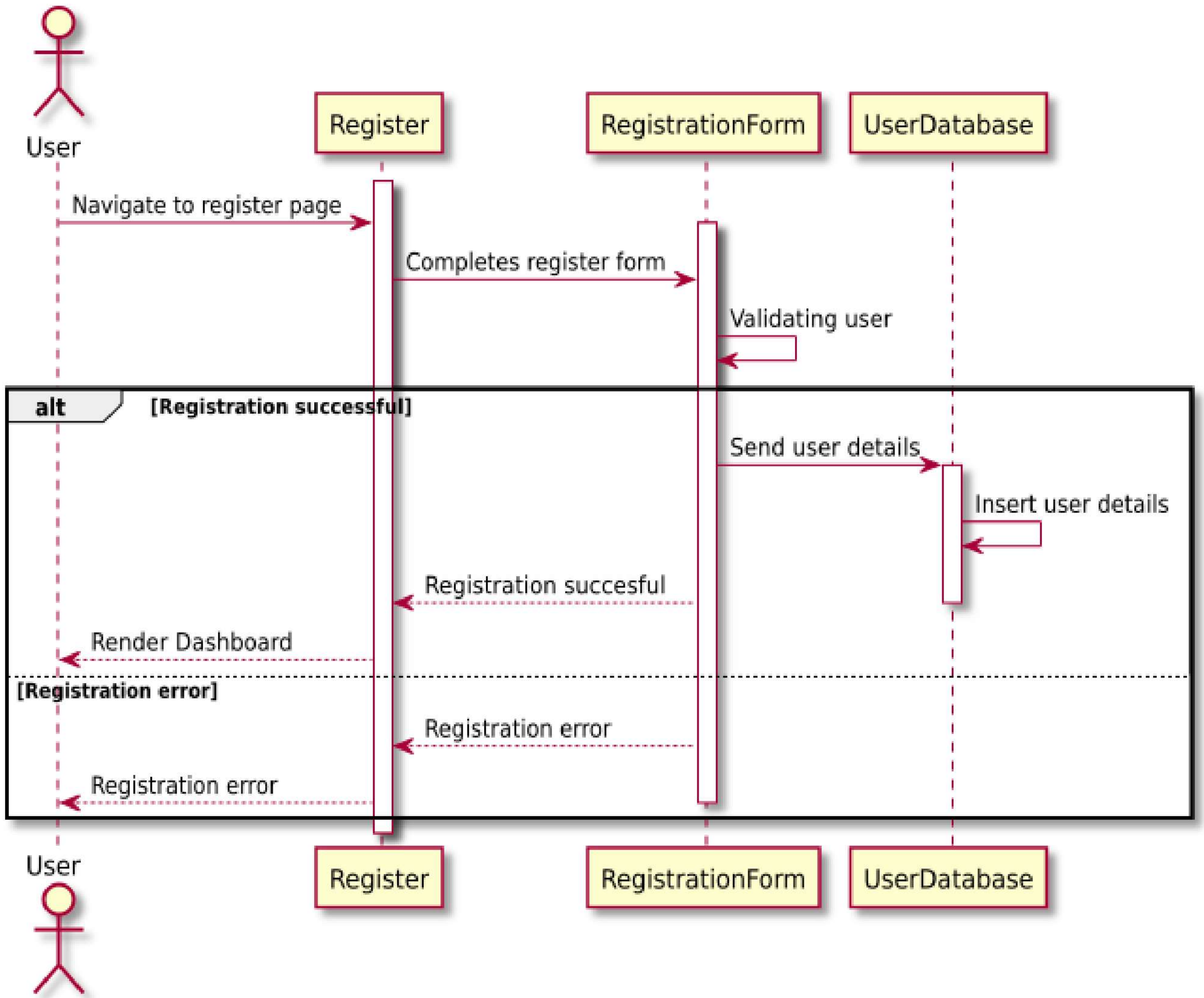
-Mission Data Upload Use Case Diagram (Orson Meyreles)



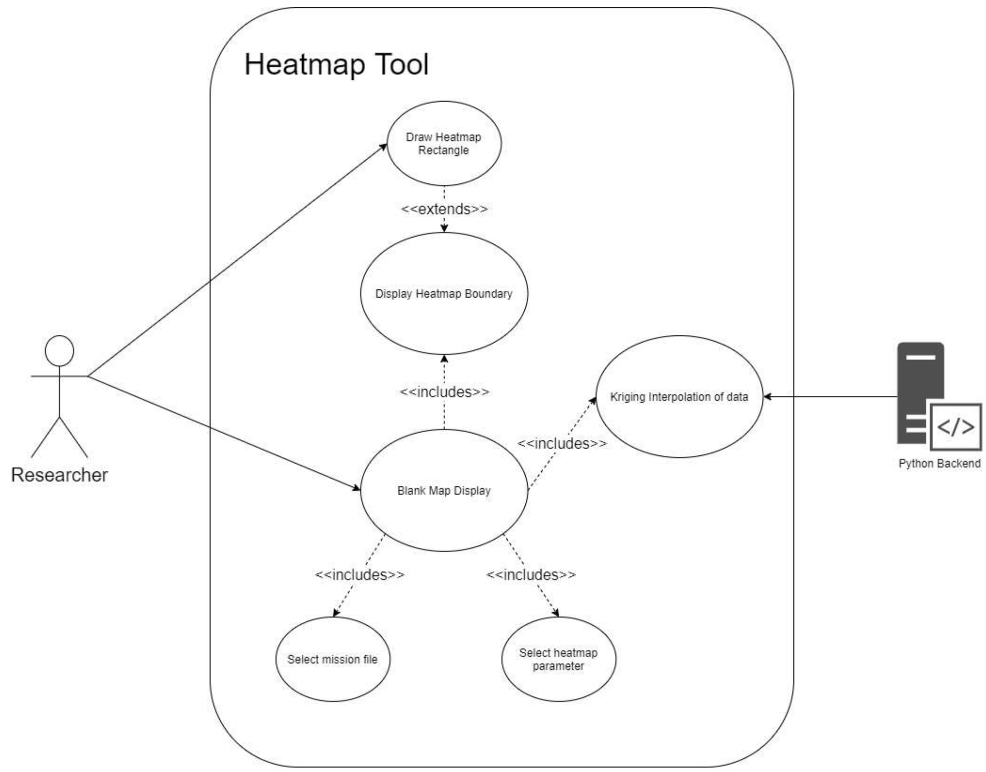
-User Login / Registration Use Case Diagram (Leonardo Pantaleon)



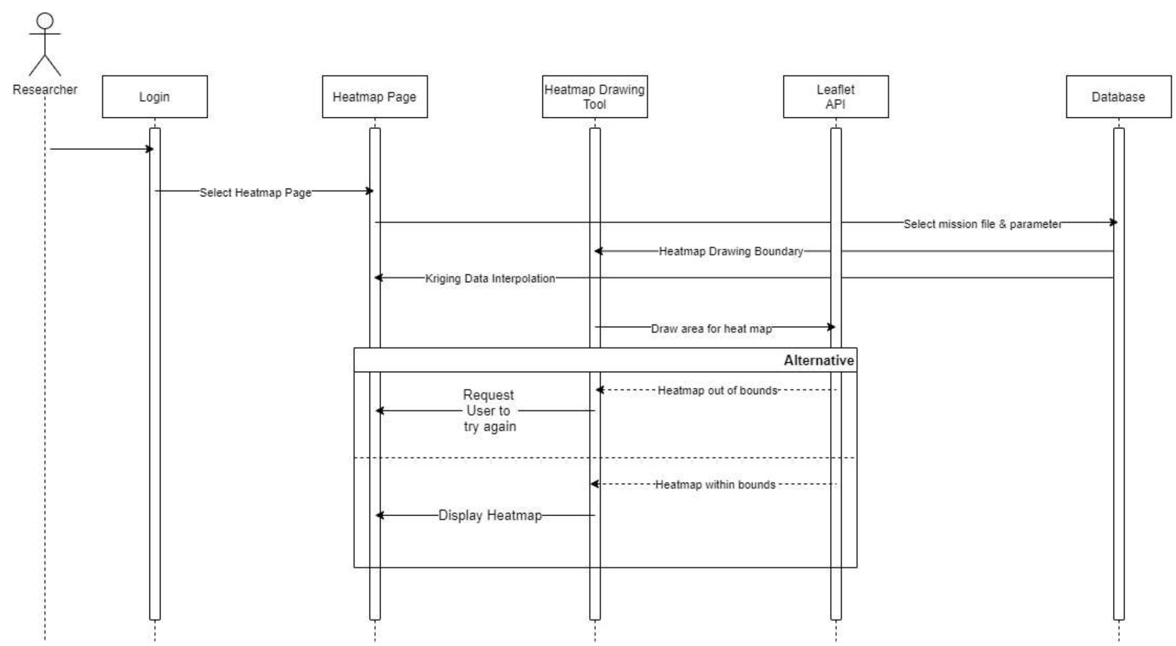
-User Login Sequence Diagram (Leonardo Pantaleon)



-User Registration Sequence Diagram (Leonardo Pantaleon)



-Heatmap Tool Use Case Diagram (Adrian Perez)

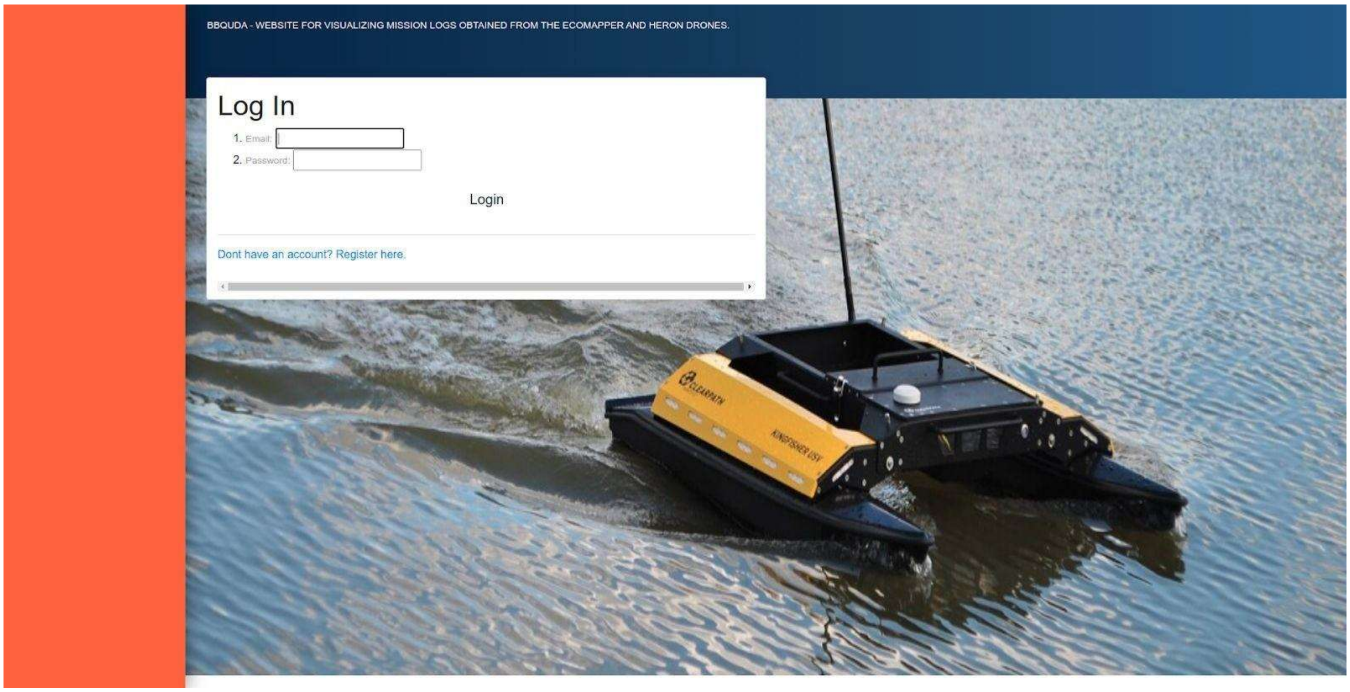


-Heatmap Tool Sequence Diagram (Adrian Perez)

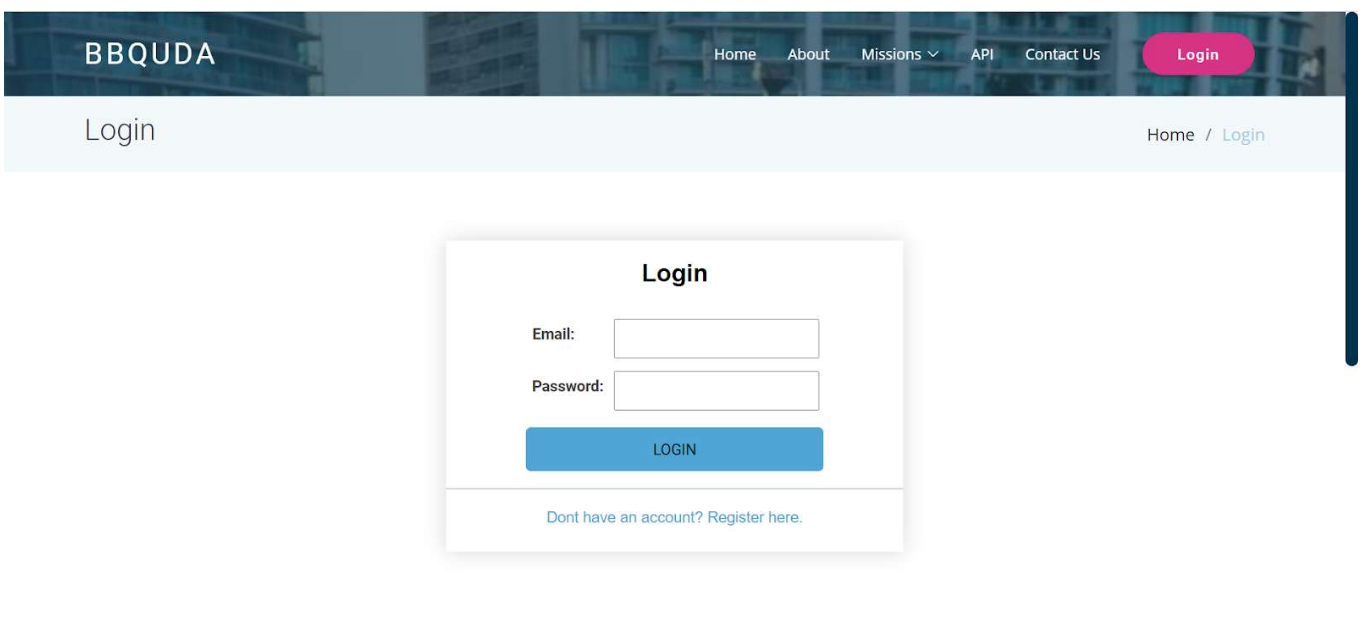
Appendix B - User Interface Design

Login Page (Before and After):

Before:



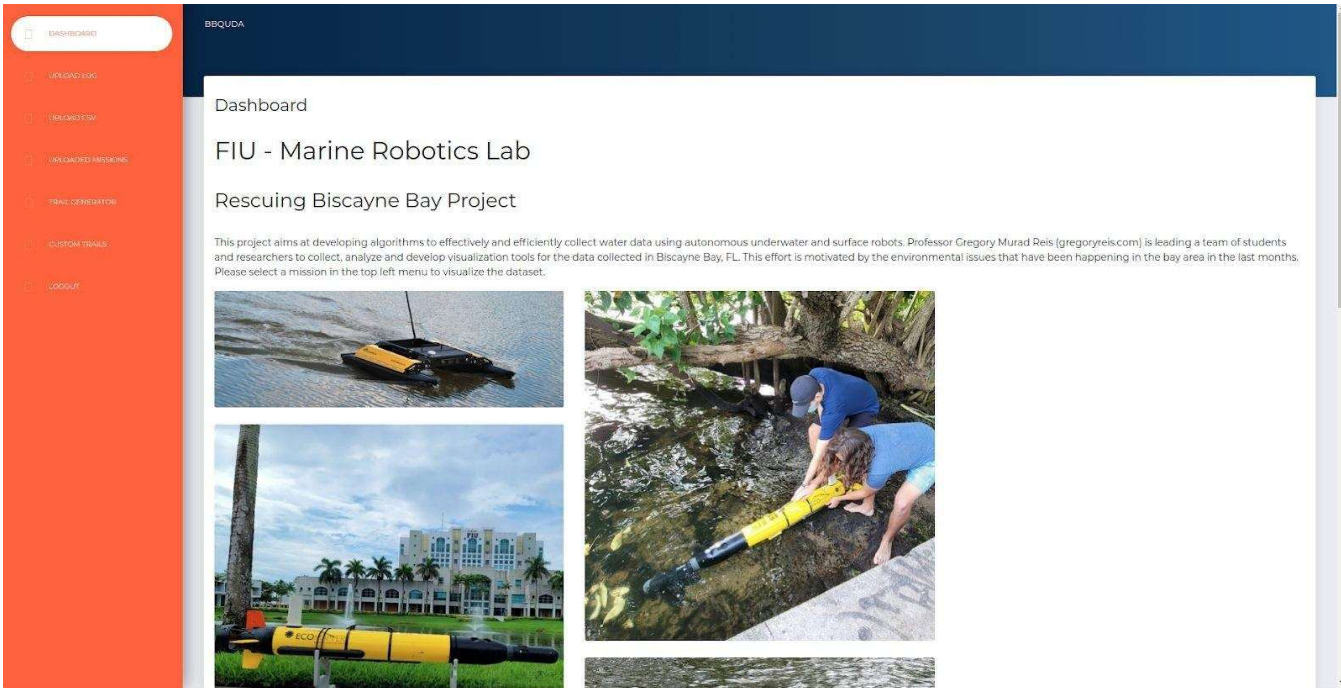
After:



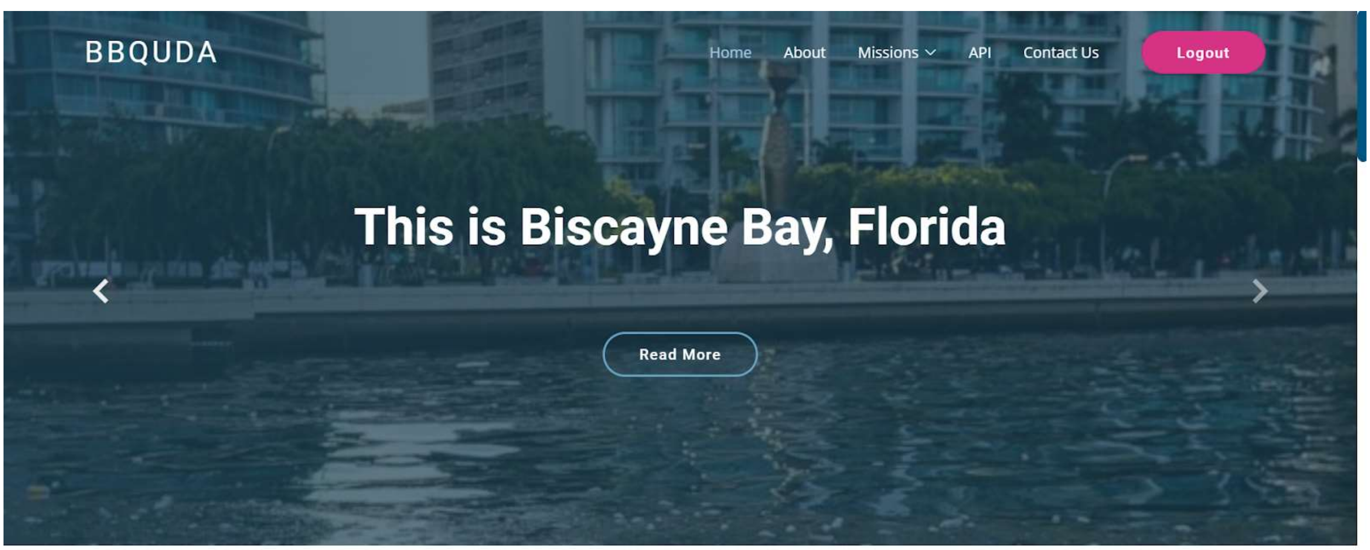
Dashboard (Before and After):

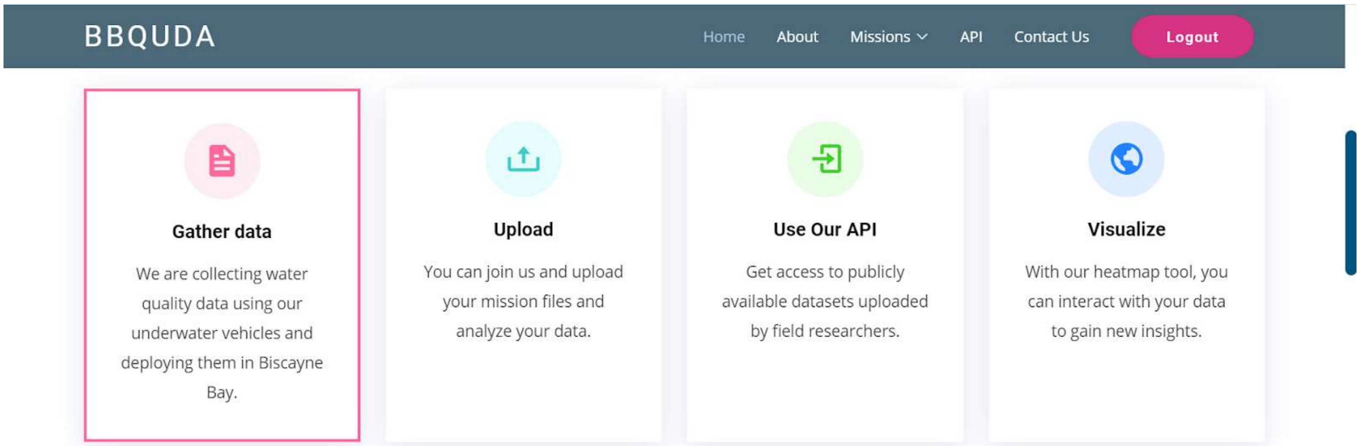
Main page when you first login:

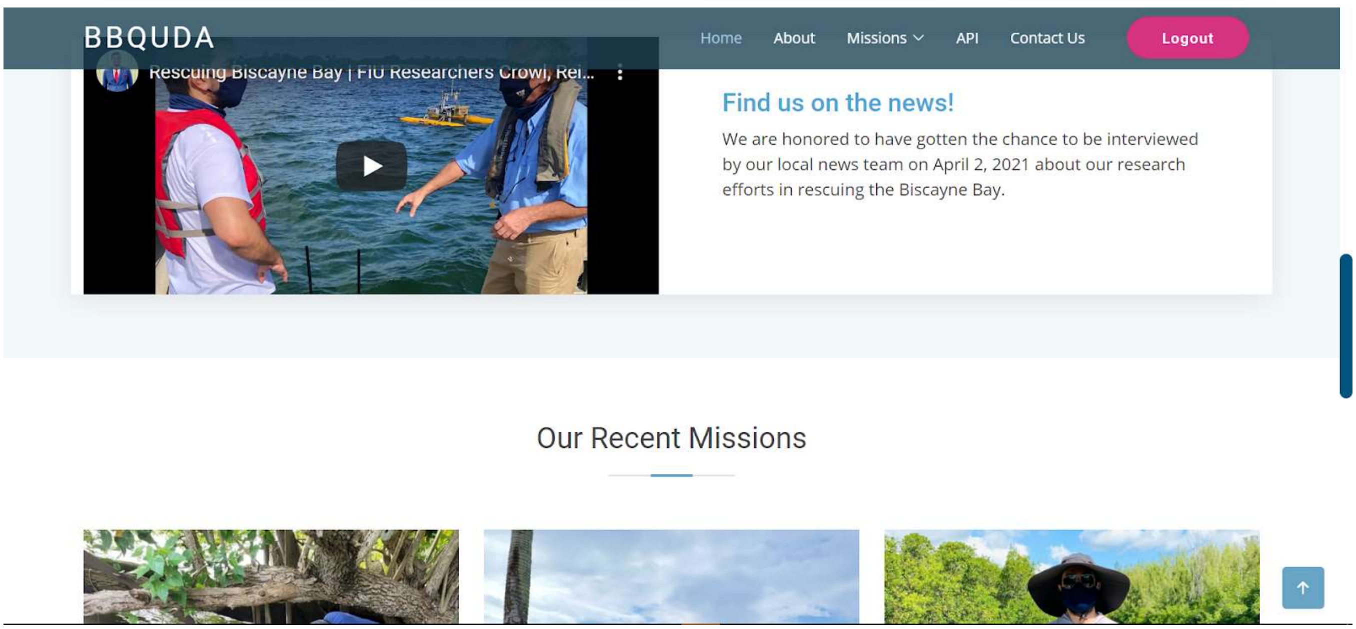
Before:



After:



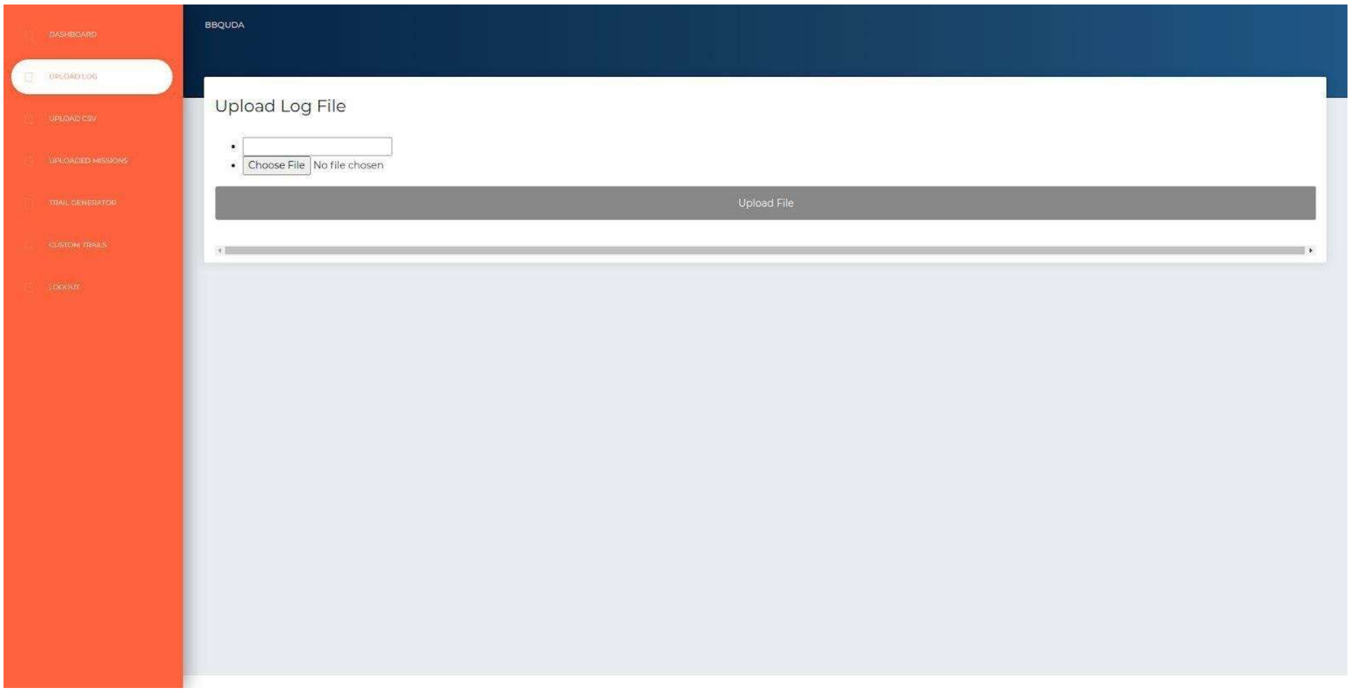

BBQUDA
Home
About
Missions
API
Contact Us
Logout
Gather data
We are collecting water quality data using our underwater vehicles and deploying them in Biscayne Bay.
Upload
You can join us and upload your mission files and analyze your data.
Use Our API
Get access to publicly available datasets uploaded by field researchers.
Visualize
With our heatmap tool, you can interact with your data to gain new insights.


BBQUDA
Home
About
Missions
API
Contact Us
Logout
Rescuing Biscayne Bay | FIU Researchers Crowl, Rei...
Find us on the news!
We are honored to have gotten the chance to be interviewed by our local news team on April 2, 2021 about our research efforts in rescuing the Biscayne Bay.
Our Recent Missions

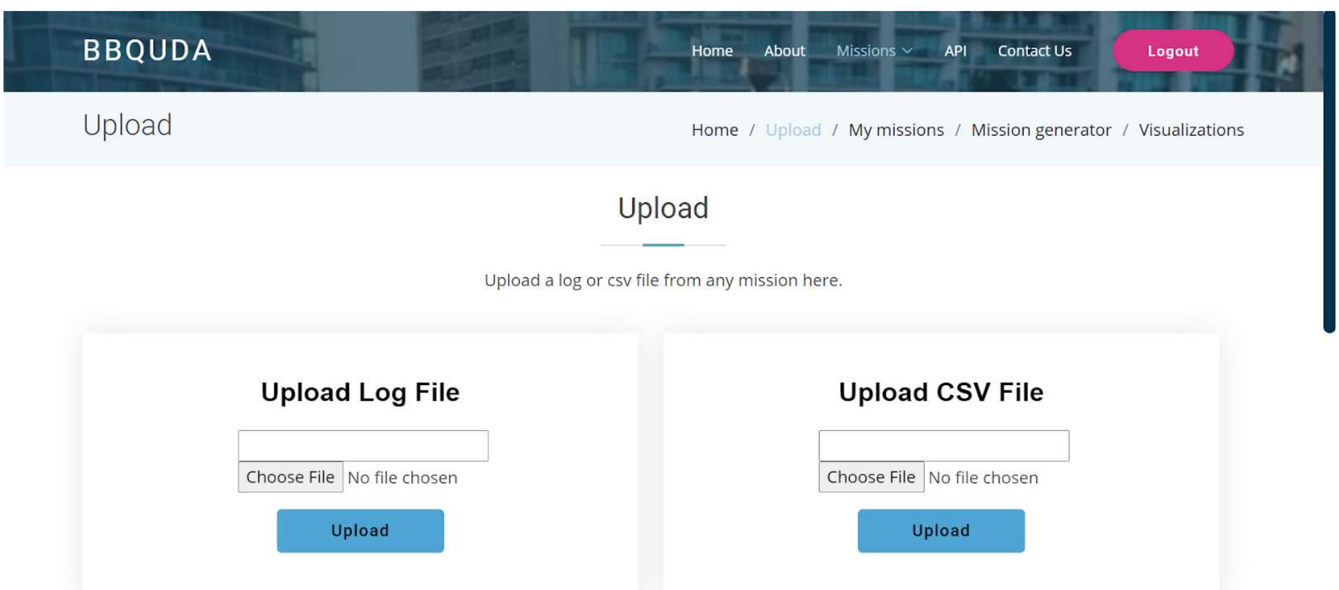

Little River
FIU
Biscayne Bay canal

Upload (Before and After):
Allows the user to upload a log file in order to be stored in the site. Opens a dialog box for the user to find the file on their system. Does not allow for a file type other than log format.

Before:



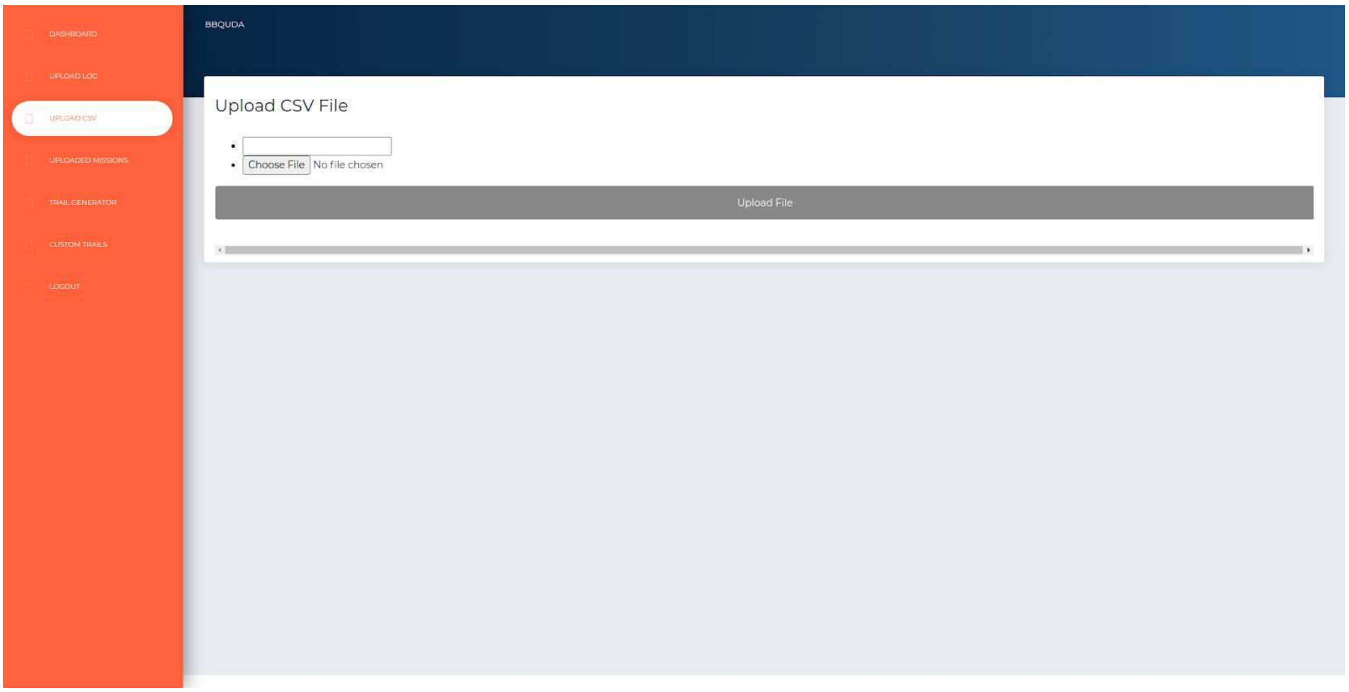
After:



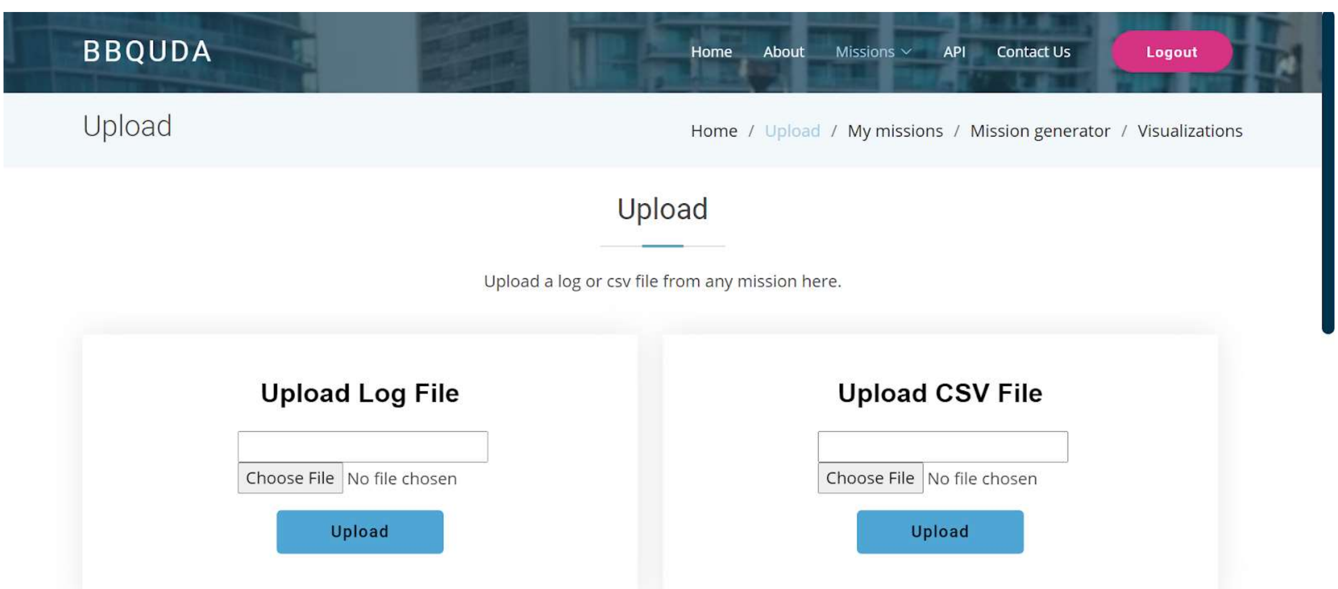
Upload CSV File:

Allows the user to upload a csv file in order to be stored in the site. Opens a dialog box for the user to find the file on their system. Does not allow for a file type other than a csv file.

Before:

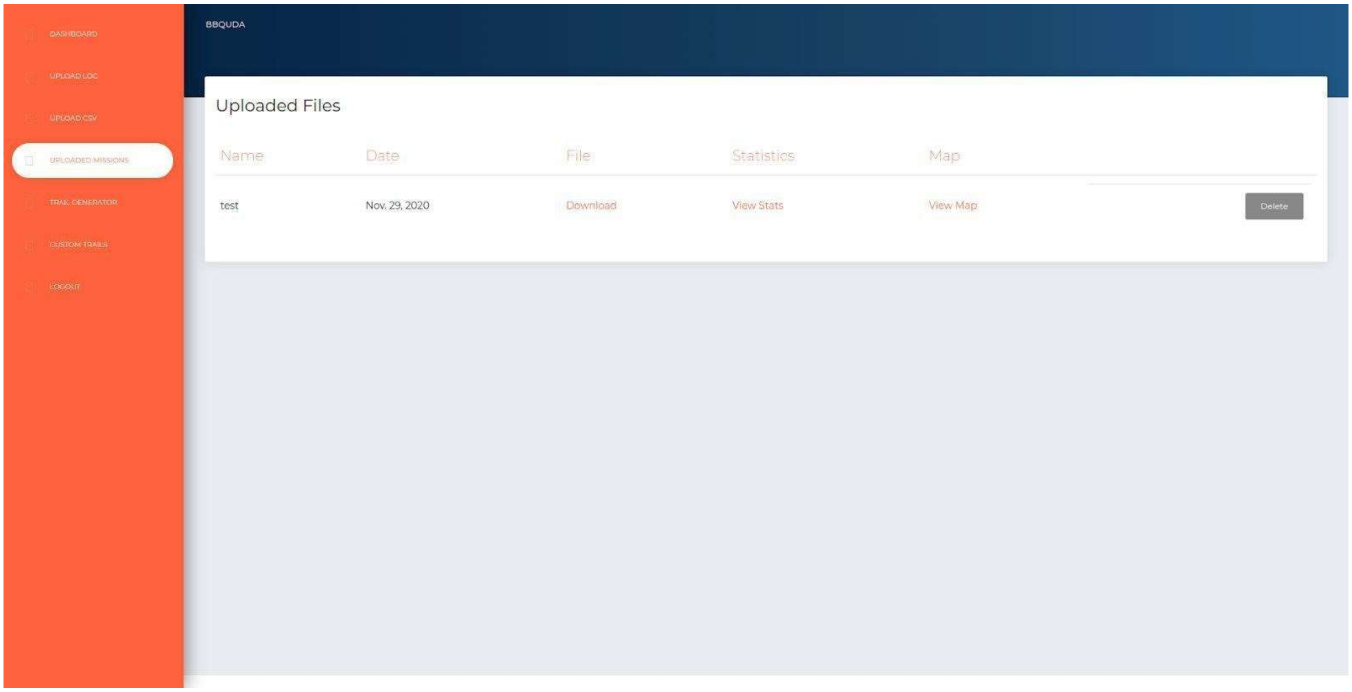


After:

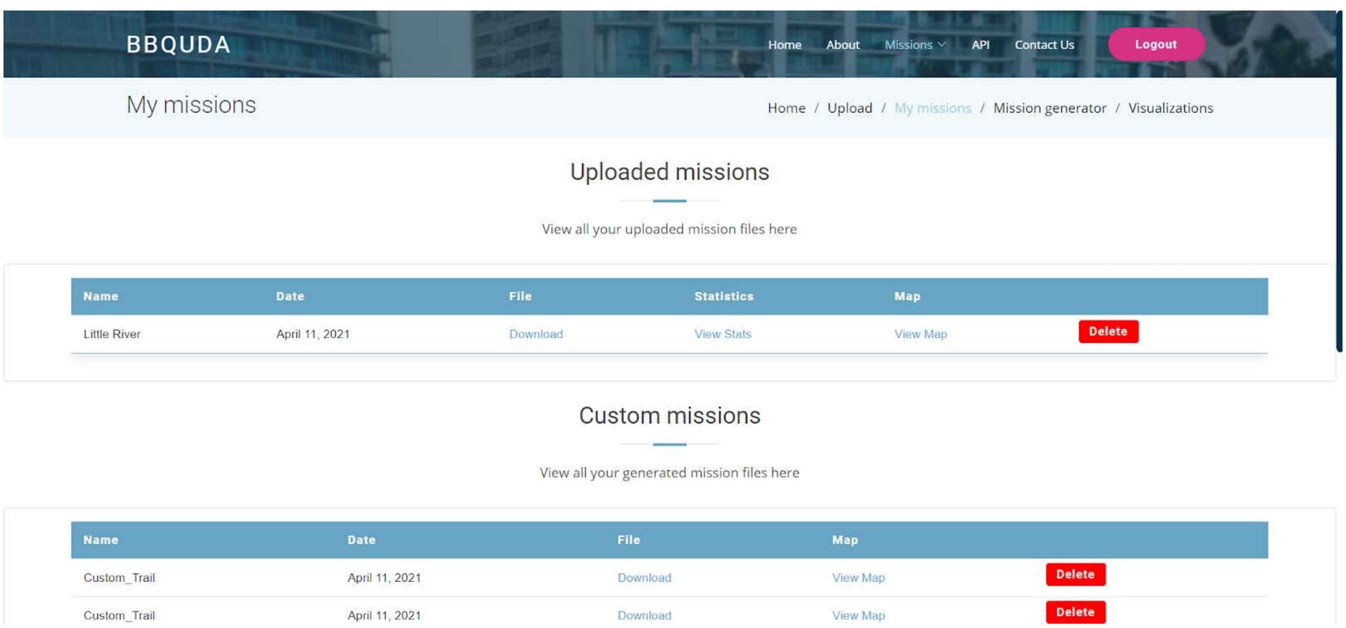


View Uploads:
User can view all of the missions that they have uploaded and choose whether to download them, view statistics, or view their path on a map.

Before:



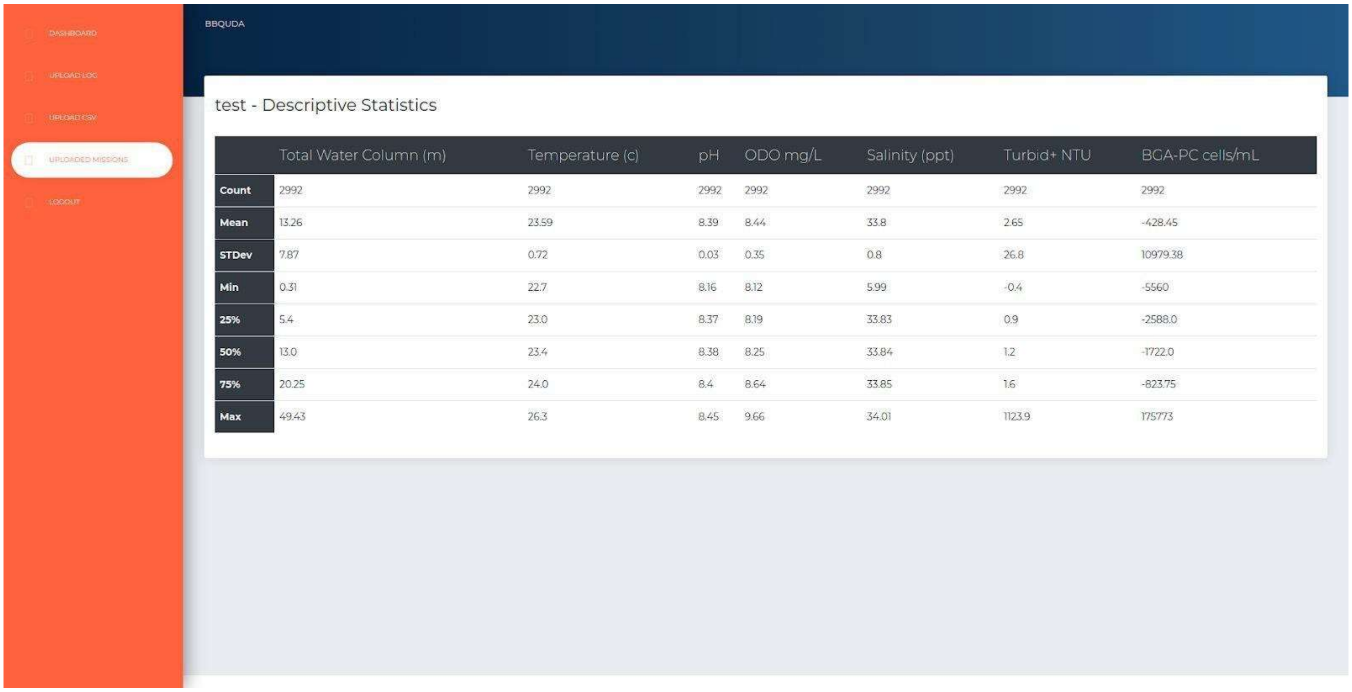
After:



View Test Data:

User can view the descriptive statistics of the data the uploaded.

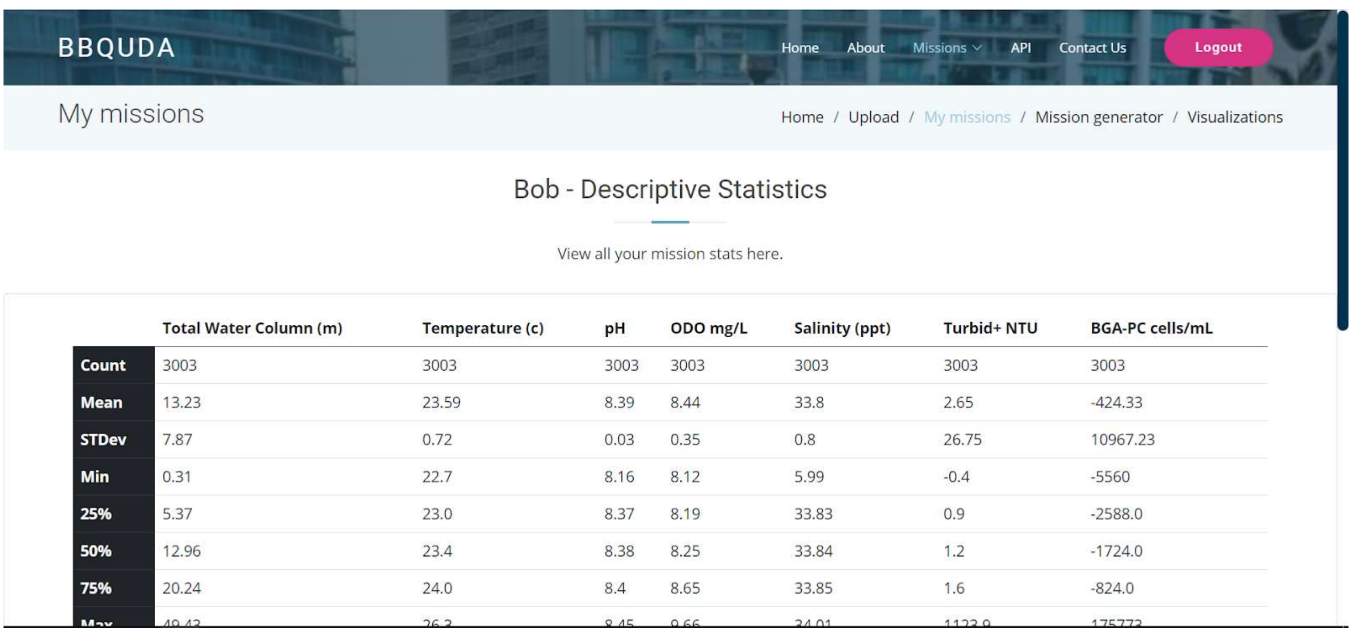
Before:



test - Descriptive Statistics

	Total Water Column (m)	Temperature (c)	pH	ODO mg/L	Salinity (ppt)	Turbid+ NTU	BGA-PC cells/mL
Count	2992	2992	2992	2992	2992	2992	2992
Mean	13.26	23.59	8.39	8.44	33.8	2.65	-428.45
STDev	7.87	0.72	0.03	0.35	0.8	26.8	10979.38
Min	0.31	22.7	8.16	8.12	5.99	-0.4	-5560
25%	5.4	23.0	8.37	8.19	33.83	0.9	-2588.0
50%	13.0	23.4	8.38	8.25	33.84	1.2	-1722.0
75%	20.25	24.0	8.4	8.64	33.85	1.6	-823.75
Max	49.43	26.3	8.45	9.66	34.01	1123.9	175773

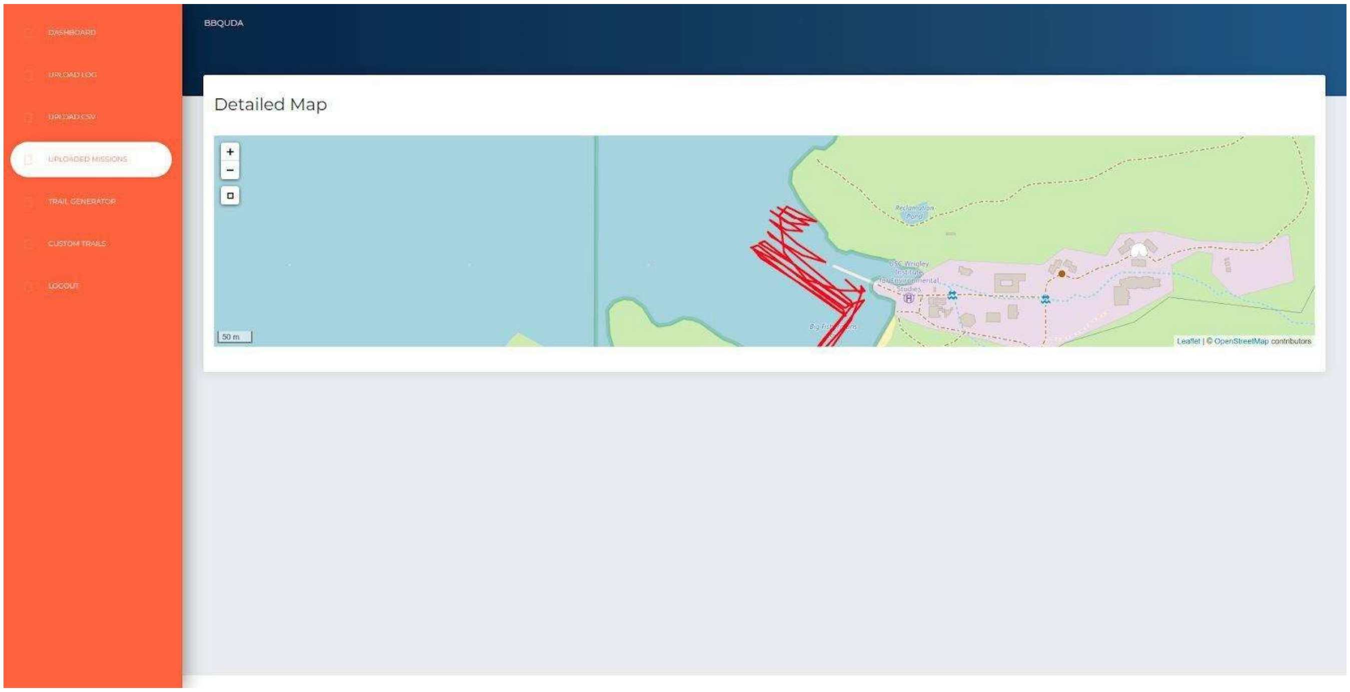
After:



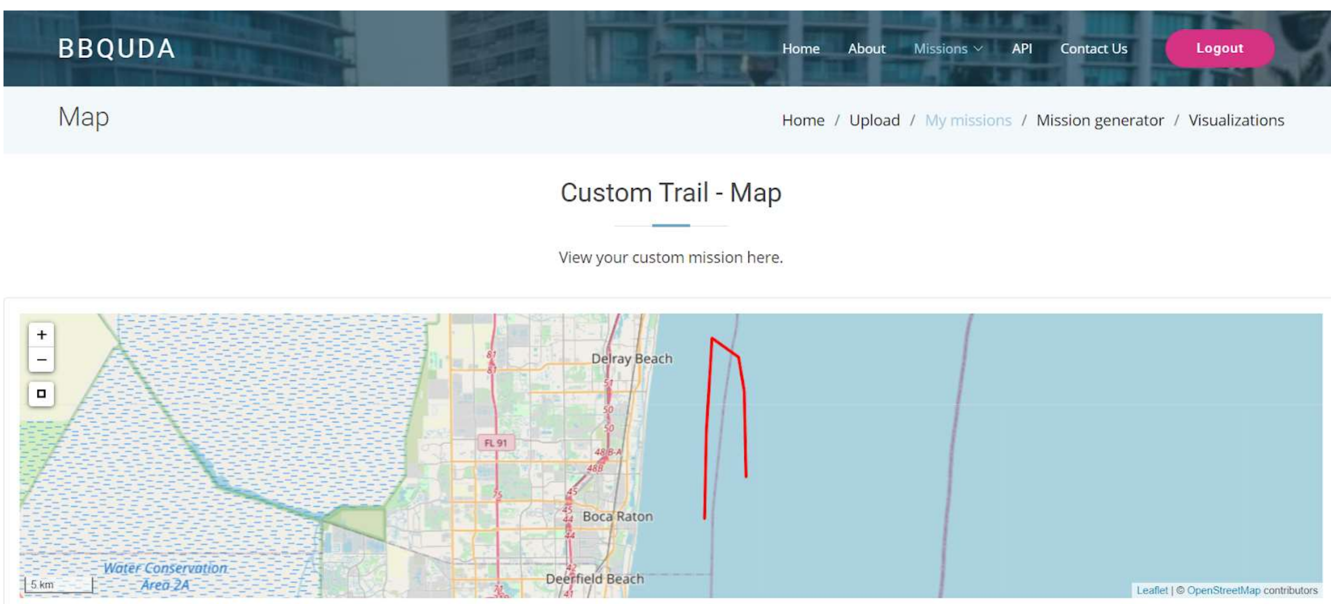
	Total Water Column (m)	Temperature (c)	pH	ODO mg/L	Salinity (ppt)	Turbid+ NTU	BGA-PC cells/mL
Count	3003	3003	3003	3003	3003	3003	3003
Mean	13.23	23.59	8.39	8.44	33.8	2.65	-424.33
STDev	7.87	0.72	0.03	0.35	0.8	26.75	10967.23
Min	0.31	22.7	8.16	8.12	5.99	-0.4	-5560
25%	5.37	23.0	8.37	8.19	33.83	0.9	-2588.0
50%	12.96	23.4	8.38	8.25	33.84	1.2	-1724.0
75%	20.24	24.0	8.4	8.65	33.85	1.6	-824.0
Max	49.43	26.3	8.45	9.66	34.01	1123.9	175773

View Map:
Shows the user a map with the path that the robot took. Hovering over the trajectory allows you to see the findings at each point.

Before:



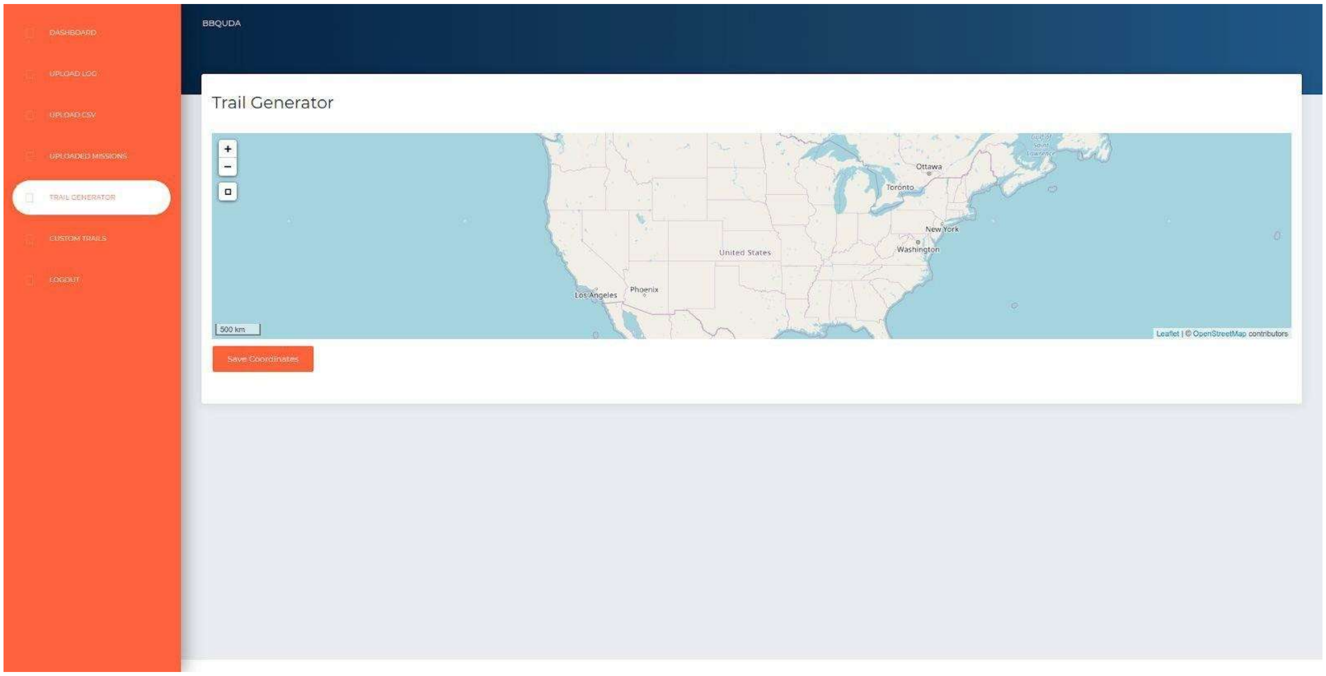
After:



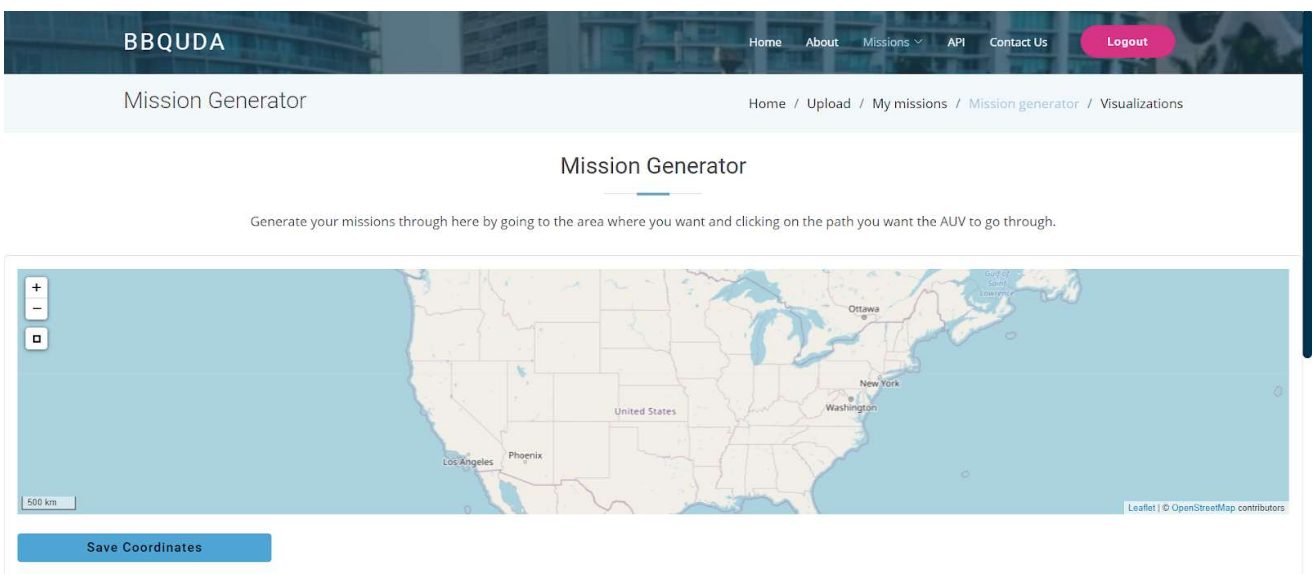
Trail Generator:

Allows the user to create their own trajectory anywhere in the world using the map. Will save after being completed.

Before:

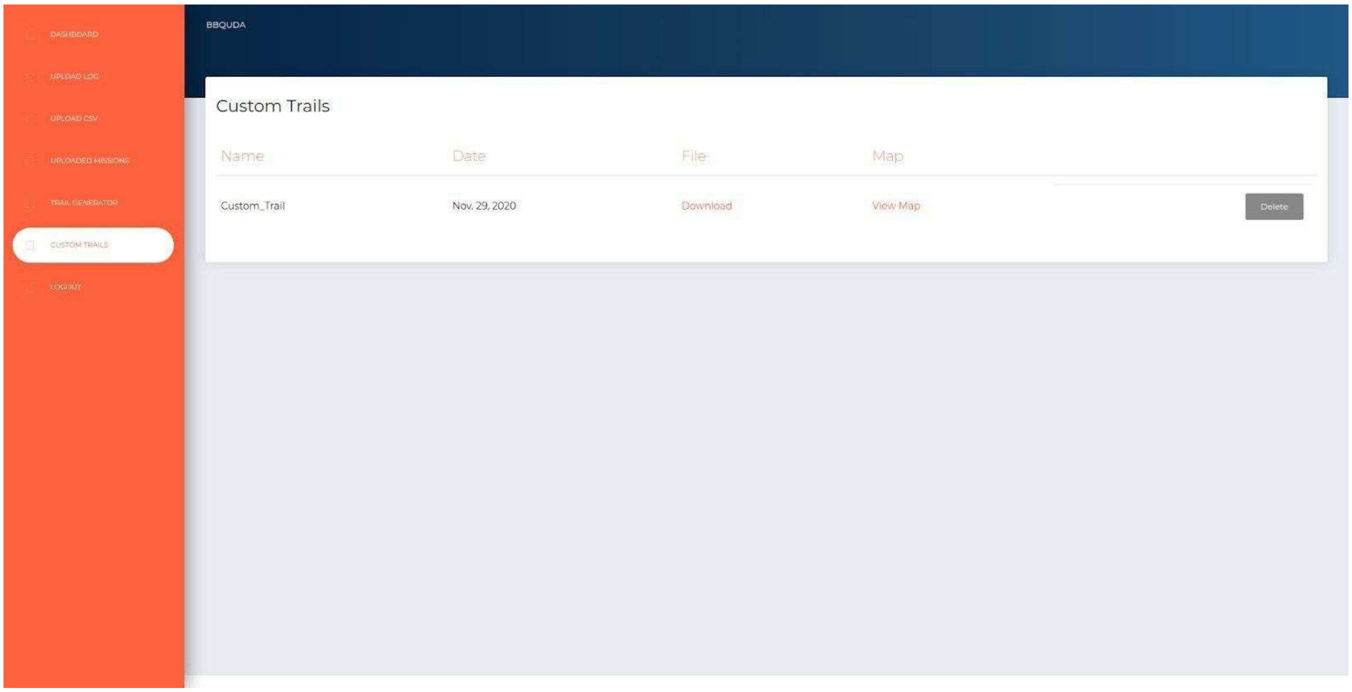


After:

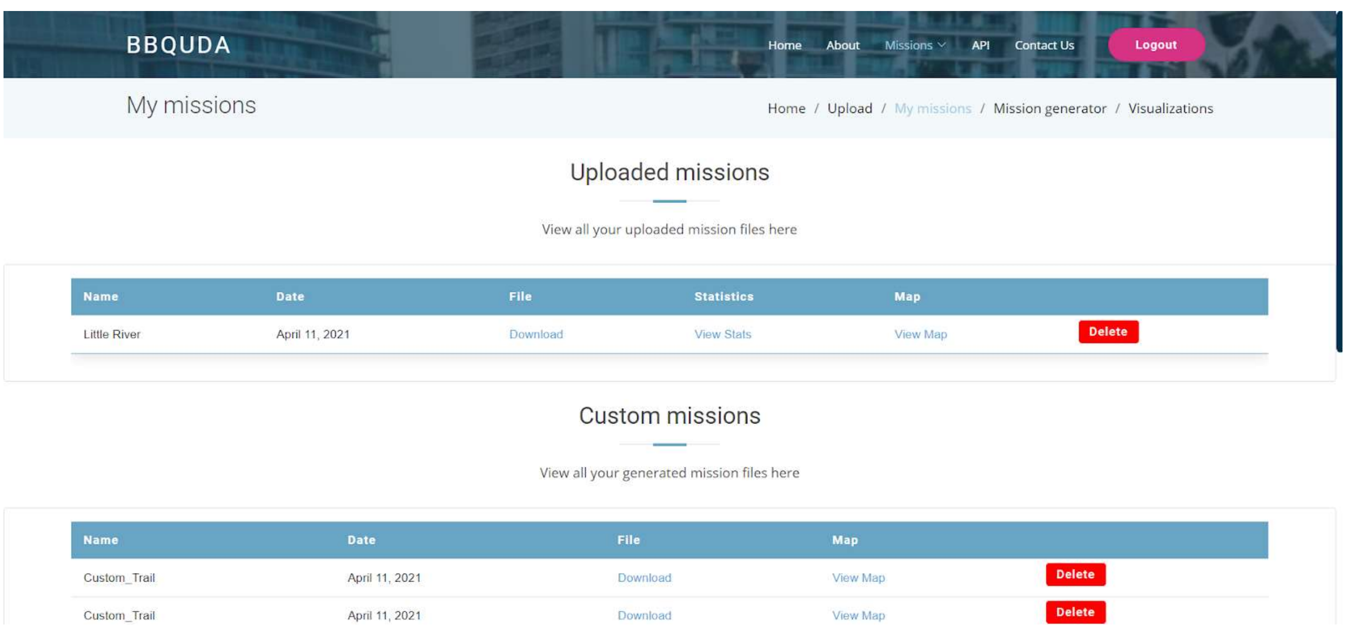


View Custom Trails:
User can view all of the custom trails that they have created and choose whether to download them as a csv of view them on a map.

Before:

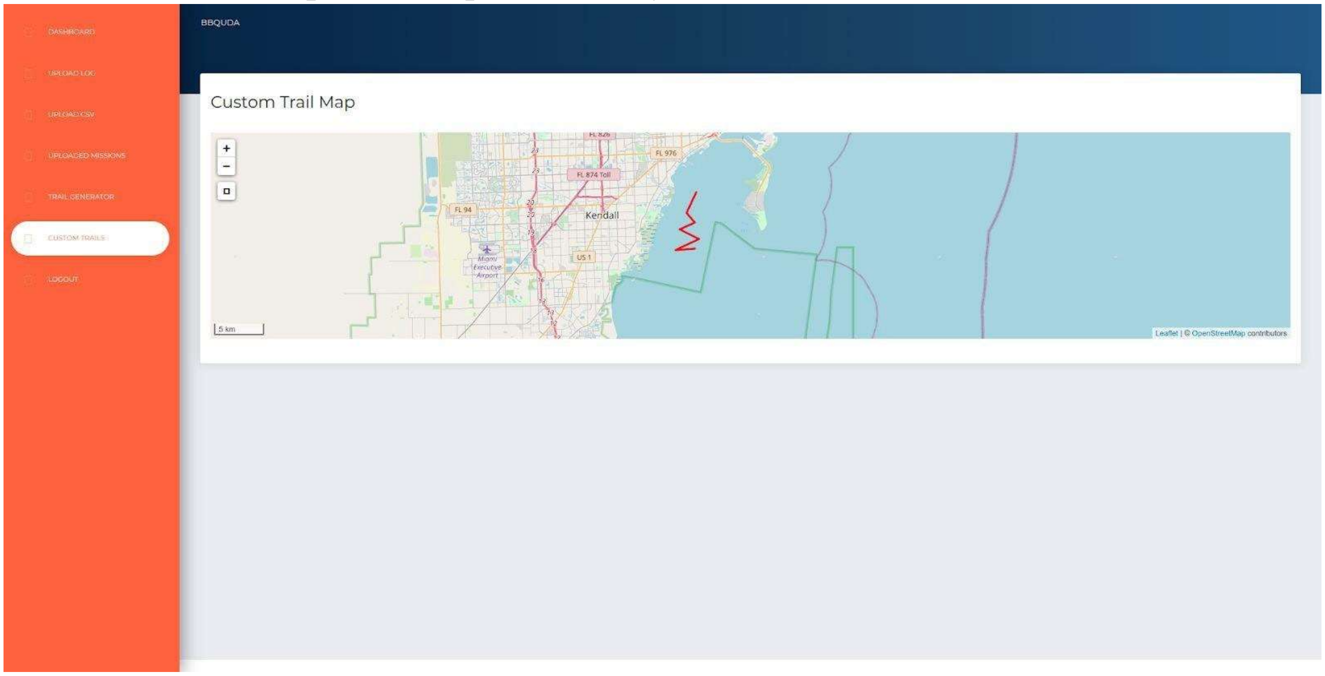


After:



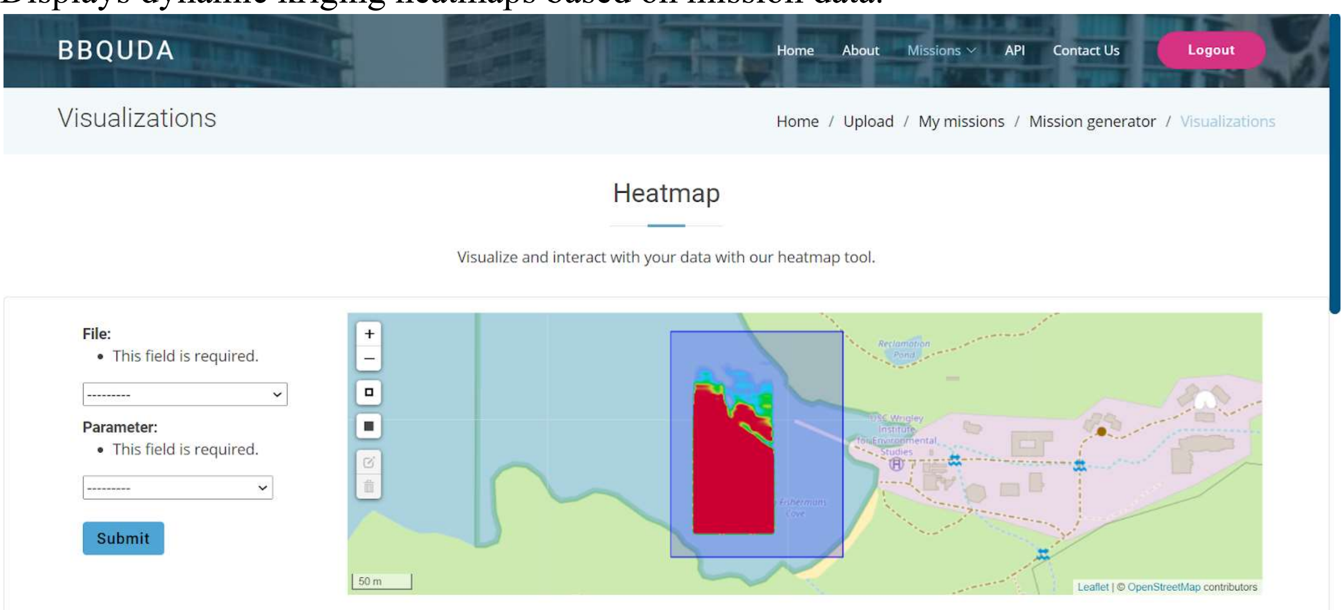
View Trail Map:

Shows the user a map with the path that they created.



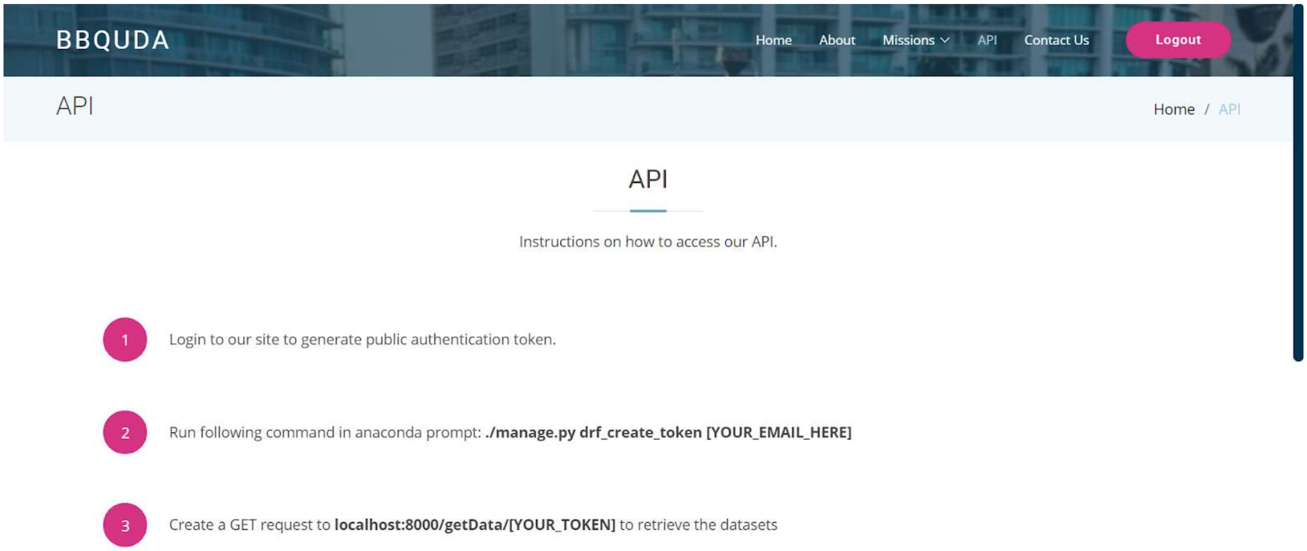
View Heatmap

Displays dynamic kriging heatmaps based on mission data.



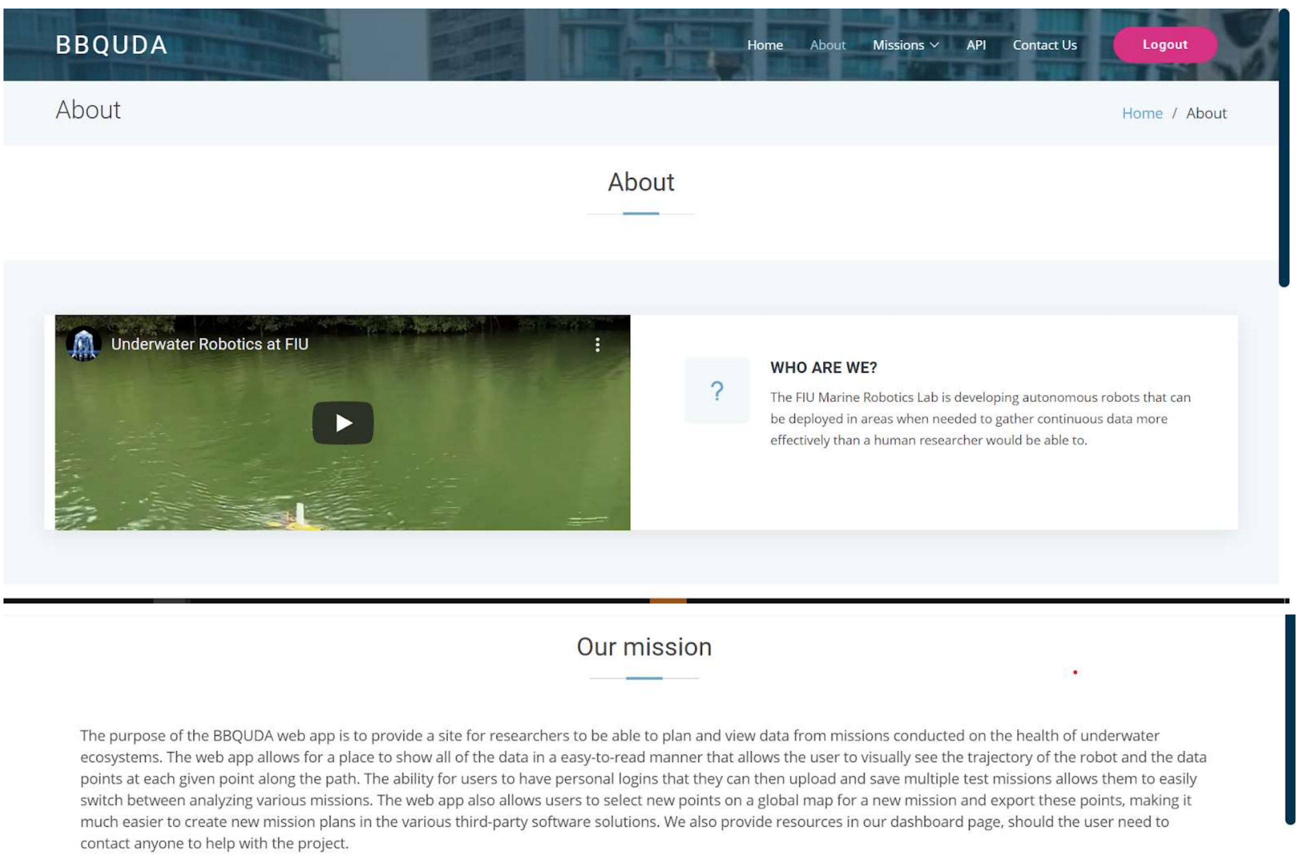
API:

Includes instructions on how to access and use the API we created.



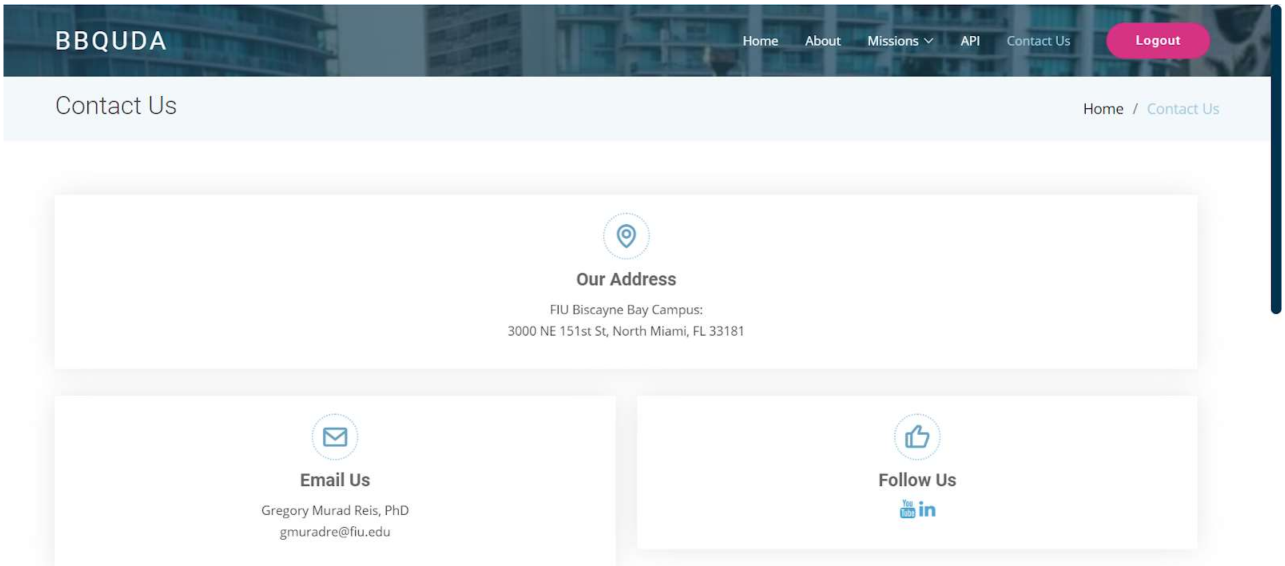
About Us:

Includes information about who we are and what our mission is.



Contact Us:

Includes information about how to get in contact with us.



Appendix C - Sprint Review Reports

- Capstone II Sprint 2 Review 1/31
  - Catherine Angelini, Meleik Hyman, Adrian Perez, John Quitto-Graham, Muddasir Riaz, Gregory Reis
  - List of Approved User Stories:
    - User Story 001
    - User Story 002
    - User Story 003
    - User Story 004
    - User Story 006
  - List of Rejected User Stories:
    - N/A

- Capstone II Sprint 2 Review 2/14
  - Catherine Angelini, Meleik Hyman, Adrian Perez, John Quitto-Graham, Muddasir Riaz, Gregory Reis
  - List of Approved User Stories:
    - User Story 005
    - User Story 007
    - User Story 008
    - User Story 010 (partially)

  - List of Rejected User Stories:
    - User Story 009 (Pushed back)
    - User Story 010 (Partially)

- Capstone II Sprint 2 Review 2/28
  - Catherine Angelini, Meleik Hyman, Adrian Perez, John Quitto-Graham, Muddasir Riaz, Gregory Reis
  - List of Approved User Stories:
    - User Story 017 (partially)
    - User Story 013 (duplicate results in later sprints)
    - User Story 012 (continue expanding in later sprints)
    - User Story 010
    - User Story 011

  - List of Rejected User Stories:
    - User Story 009 (partially)
    - User Story 017 (partially)

- Capstone II Sprint 2 Review 3/14
  - Catherine Angelini, Meleik Hyman, Adrian Perez, John Quitto-Graham, Muddasir Riaz, Gregory Reis
  - List of Approved User Stories:
    - User Story 009
    - User Story 017 (partially)
    - User Story 013 (duplicate results in later sprints)
    - User Story 012 (continue expanding in later sprints)
    - User Story 014

        - User Story 011

    - List of Rejected User Stories:
        - User Story 017 (partially)

- Capstone II Sprint 2 Review 3/28
    - Catherine Angelini, Meleik Hyman, Adrian Perez, John Quitto-Graham, Muddasir Riaz, Gregory Reis
    - List of Approved User Stories:
        - User Story 017
        - User Story 013 (duplicate results in later sprints)
        - User Story 012 (continue expanding in later sprints)
        - User Story 011 (continue expanding in later sprints)
        - User Story 015 (Partially)
        - User Story 016
        - User Story 020

    - List of Rejected User Stories:
        - User Story 015 (Partially)

- Capstone II Sprint 2 Review 04/11
    - Catherine Angelini, Meleik Hyman, Adrian Perez, John Quitto-Graham, Muddasir Riaz, Gregory Reis
    - List of Approved User Stories:
        - User Story 011
        - User Story 012
        - User Story 013
        - User Story 015
        - User Story 018
        - User Story 019
        - User Story 021

    - List of Rejected User Stories:
        - N/A

Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents

Installation, user guide and shortcomings can be found in the following videos, along with other parts of this deliverable.

- Installation Guide:https://youtu.be/vESZ1-s02cY
- User Guide: https://youtu.be/NRS31gcFNXo
- Shortcomings: https://youtu.be/RlP7X0gZF5U