

Stage VI- Final Project Report

Inception: Team Identification

Team Name: Pollution Prognosticators

Project Name: Pollution Prediction

Project URL: <http://csc415-team04.tcnj.edu/SOAP/index.php/map>

Team Members:

Name	Email	GitHub Account
Hunter Dubel	dubelh1@tcnj.edu	hdubel94
Jeremy Leon	leonj2@tcnj.edu	jbleon95
Richard Levenson	levensr1@tcnj.edu	richlvnsn
Evan Melquist	melquie1@tcnj.edu	melquiste
Zachary Nelson	nelsonz1@tcnj.edu	nelsonz1

Changes Made

- Added more info on Similar Systems in Stage II
- Milestones and issues were created and assigned in Stage II and we were not sure why we got points off
- All of the actual content is now on the Canvas Wiki Page and GitHub Wiki Page (Stage III)
- Updated the design class diagram in Stage IV
- Added error handling for latitude/longitude search bars for Stage Vb

Table of Contents

Inception: Team Identifications-----	Page 1
Changes Made-----	Page 1
Inception: Objectives and Overview-----	Page 3
Elaboration: Requirements Modeling and Analysis-----	Page 6
Elaboration: Design-----	Page 12
Construction: Implementation--- -----	Page 18
Construction: Testing-----	Page 20
Transition: Maintenance -----	Page 21
Transition: Product Hand Over-----	Page 22

Inception: Objectives and Overview

Project Goals and Aspirations:

Problem Statement- The problem that we are attempting to solve is that there is currently no way to determine the amount of pollution at a lot that has not yet been chemically tested. Chemical testing is an expensive process and it is not feasible to test every area that Habitat for Humanity is looking into for housing projects. It would be extremely beneficial for organizations like Habitat for Humanity if they were able to approximate the level of pollution at a certain lot before they spend money to test it. For example, if neighboring lots were very polluted Habitat for Humanity might decide that they should not consider that lot because of the high pollution risk. However, if it can be determined that a lot is unlikely to be polluted, it would be worth it for these organizations to invest money into further testing.

Module Objective- The completed module will accurately display polluted areas, both explicit and predicted sites of pollution. The severity and the prediction of the polluted areas will be accomplished with the help of an algorithm that will take in data about the polluted sites as an input and output the predicted polluted sites, with information on the severity of the pollution. This information will be displayed on the map module of the SOAP system, allowing users of SOAP to easily assess polluted areas and see areas that have a strong chance of being polluted and dangerous.

Why we Chose this Project:

Importance and Need for Module- The module is important because there is no way to visibly observe if a piece of land is polluted or not. A piece of land that looks green and healthy on the surface can be considered a brownfield because of the chemicals that are hidden within the soil. This makes it difficult to predict what pieces of land will be good candidates for organizations like Habitat for Humanity without using historical data of the area or data from surrounding areas. The historical data is not always available or accurate enough to provide a good prediction, so using data from surrounding areas may be the most accurate solution. This information is vital to the primary use of SOAP. Therefore, this module is an important addition to the project.

Why the Module is Innovative- The module is innovative because it uses an algorithm to predict pollution levels based on current pollution and geographical data. Current related SOAP modules include a map of the previously tested sites and a list of current pollution data, including chemicals that were found, at each site. These modules are helpful for someone that is looking for information at these specific locations, but they do not provide any information for new locations that have not yet been tested. The proposed module will solve this problem and provide a way to determine what future locations are worth testing.

What we Hoped to Learn:

Technologies and Concepts Needed- The team will need to learn the languages and systems that makeup the current SOAP system, which includes HTML5/CSS3, JavaScript, PHP5, PostgreSQL and CakePHP (since the module will not include Twitter support, Twitter Bootstrap will not be necessary to learn for our completed module). For concepts, the team will need to understand prediction algorithms and how they can be applied to accurately predict pollution. The team will have SOAP contributor Thomas Borgia as a resource for the previous work on this algorithm.

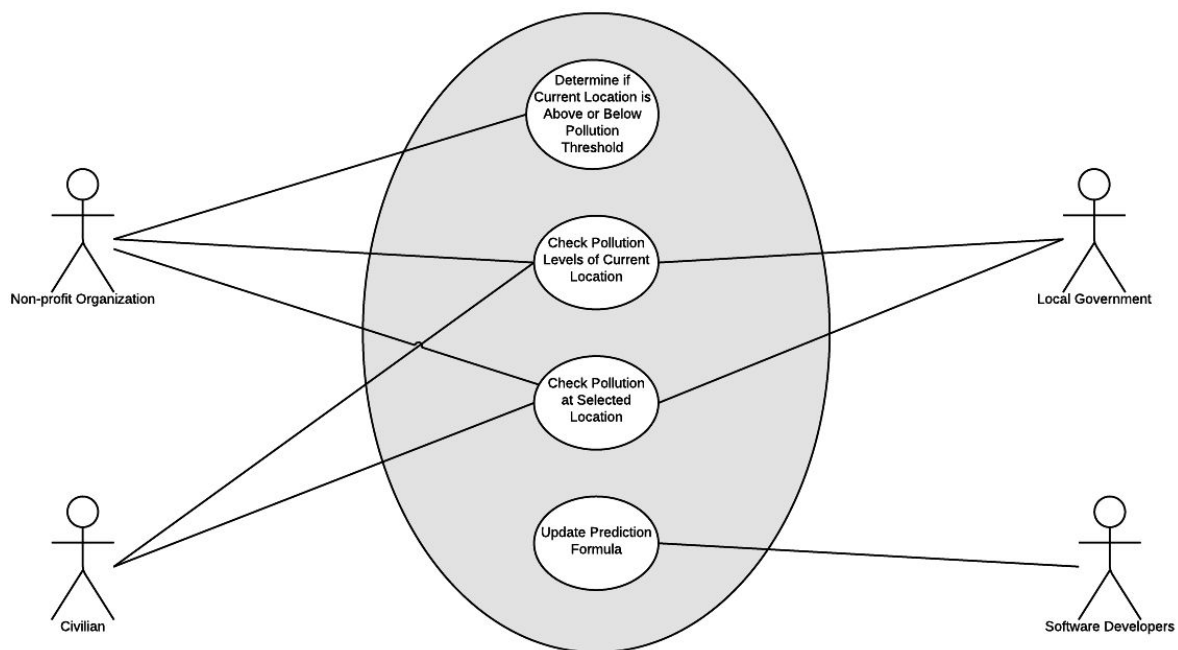
Similar Systems/Approaches- The current SOAP systems that are most similar to our module are the map and the list of sites. These systems display data about lots that have already been chemically tested. They take the existing data and directly plot it on the map or display it on a page. Other systems that are more similar to our module have been worked on in the past but have not been completed. For example, Thomas Borgia has implemented an algorithm similar to the team's end objective, yet it has not been fully implemented in the SOAP system. The focus of our project will be on continuing his previous work and not creating a new algorithm. A similar approach besides the current SOAP project is the Cleanups in My Community (CIMC) tool provided by the United States Environmental Protection Agency (EPA). Even though this system does not predict pollution levels at any location, it is similar to our project because it shows what pollution information is relevant for a given location. Another similar system is the AirNow system that estimates air quality at different locations. Our system will be dealing with ground

pollution, but the prediction algorithm that the AirNow system uses could be something that we learn from and add to Thomas Borgia's current algorithm.

Original Plan for the Project:

End Product Description- The final design of the product should successfully predict the pollution level of any location chosen on the SOAP pollution map. The map module should be interactive and take into consideration a pollution prediction module and algorithm. The algorithm will use local pollution data, such as proximity, intensity, and chemical composition, in order to successfully predict the degree of the pollution on the desired lot. The feature will work in correspondence with the other modules of SOAP in order to help upkeep the standard of usability and quality.

Use Case Diagram:



Elaboration: Requirements Modeling and Analysis

Security:

Since our project is involved with predicting the pollution at a given lot, we will need to access the database to obtain information for our prediction algorithm. One security feature that needs to be implemented is to ensure that the user cannot access the database directly and can only see pollution levels at a specific location. This is an important feature because we do not want the public to see how SOAP stores its data or be able to change the data that is currently stored in the database. The user should not be able to download all of the data to their local machine and must go through the SOAP system in order to access the data. Lastly, any code concerning the clustering algorithm for prediction pollution should not be visible to the public. We will provide these features by making sure that any modules we add for prediction do not expose any aspects of the database.

Backup and Recovery:

The database that holds the prognostication data will be relevant to backup and recovery. Handling errors of the database and the failure of the site itself will be important, displaying proper error messages and correctly updating the database if it is lost. There is also the possibility that the prediction algorithm fails if the required information is not present or if the output is incorrect. This will be handled by displaying proper error messages and recovering the page before the prediction algorithm was invoked.

Backup and recovery of development files will be handled by Git and GitHub. Backups are handled by Git commits and can additionally be pushed to the GitHub repository. Recovery of previously backed up files can be done using the Git checkout command to change to the previous commit.

Legal Issues:

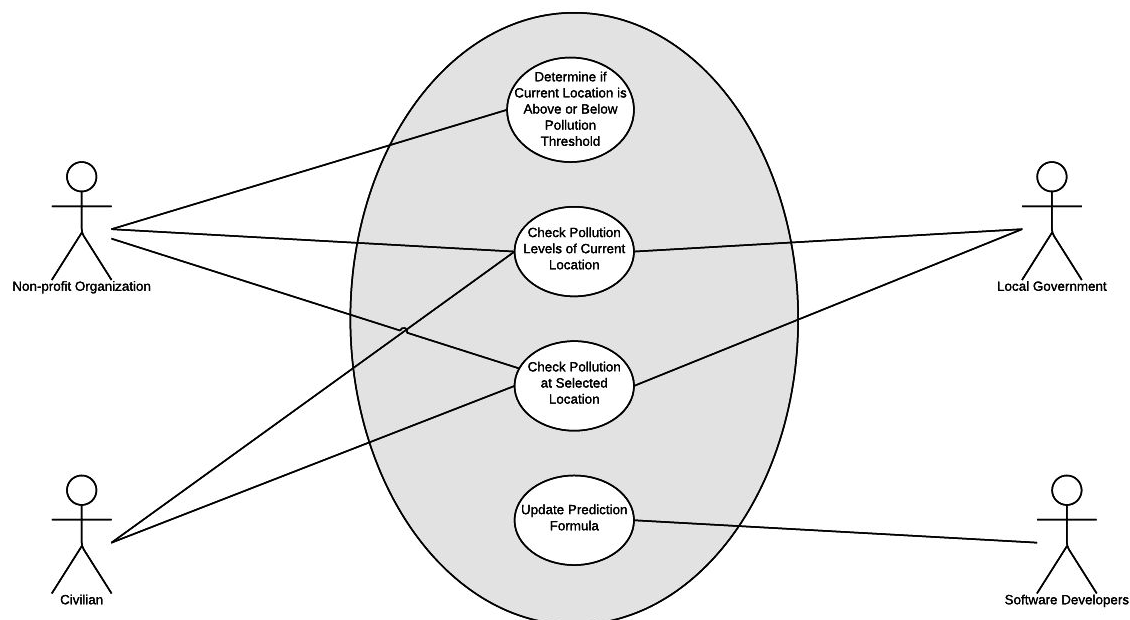
If one of our calculations that a lot is polluted turns out to be false there may be liability issues. The ideal way to avoid this would be to have the most accurate prediction algorithm as possible. We must make sure that our pollution prediction algorithm does not infringe on any copyright laws concerning pre-existing algorithms. We have to make sure that the

SOAP website explicitly states when the pollution information is actual and when it is just a prediction so that there is no ambiguity.

Possible Applications:

The pollution prediction algorithm is designed towards the specific needs of the SOAP project. Since its goal is to successfully predict or prognosticate the pollution of a designated area, this system could be slightly modified to work with various other environmental advocacies to predict the levels of pollution around the globe. Although the algorithm would not be used by the general populous, environmentalists and various other scientific professionals could utilize it to help predict future environmental concerns. Additionally, the clustering algorithm could be modified to be used for applications outside of the scope of environmental awareness. For example, it could predict the levels of available natural resources using local information.

Use Case Diagram:



Use Case Descriptions:

Use Case: Determine if Current Location is Above or Below Pollution Threshold

- Iteration: Stage II, last modification: September 24 by Dr. Pulimood
- Primary Actor: Non-Profit Organization
- Goal in Context: To determine if a selected location is above the calculated pollution threshold.

- Preconditions: System must be configured, user must be using the map feature, valid area must be selected
- Trigger: User selects a location and decides to check if pollution is over threshold
- Scenario:
 1. User enters SOAP Map
 2. User selects location to check pollution
 3. If location is a brownfield, automatically above threshold
 4. Else, clustering algorithm determines if current location is polluted above threshold
 5. Map shows user result
- Exception:
 1. Map fails to load
 2. User chooses non-valid location
 3. Not enough information at location for algorithm
- Priority: High priority, must be implemented alongside other basic functions
- When Available: Stage V
- Frequency of Use: High Frequency
- Channel to Actor: Via SOAP's Map Feature
- Secondary Actors: System Administrators, Citizens
- Channels to Secondary Actors: Via SOAP's Map Feature
- Open Issues:
 1. What error message to give if user chooses non-valid location?
 2. What to do if algorithm fails due to lack of appropriate input?
 3. What information should be shown if pollution very close to threshold?

Use Case: Check Pollution Levels at Current Location

- Iteration: 2, last modification: September 24 by Dr. Pulimood
- Primary Actor: Civilian
- Goal in Context: To determine how polluted the current location is for personal safety concerns.
- Preconditions: System must be configured, user must have given location access permissions to the system from his or her device, user must be using the map feature, user must be in a valid location.
- Trigger: User wants to determine how polluted his surroundings are.
- Scenario:
 1. User enters SOAP Map
 2. User selects "current location" option
 3. If the location is a known brownfield, displays the currently known pollution information.
 4. Else, clustering algorithm determines what the likely pollution levels are.
 5. Pollution information is displayed in a pop-up window.
- Exception:
 1. Map fails to load
 2. User does not grant location access permissions
 3. User is in a non-valid location

- 4. Not enough information at location for algorithm
- Priority: High priority, must be implemented alongside other basic functions
- When Available: Stage V
- Frequency of Use: High frequency
- Channel to Actor: Via SOAP's Map Feature
- Secondary Actors: Non-profit Organizations, Local Government
- Channels to Secondary Actors: Via SOAP's Map Feature
- Open Issues:
 1. What error message to give if user is in a non-valid location?
 2. What error message to give if user has not granted location access permissions?

Use Case: Check Pollution Level at Selected Location

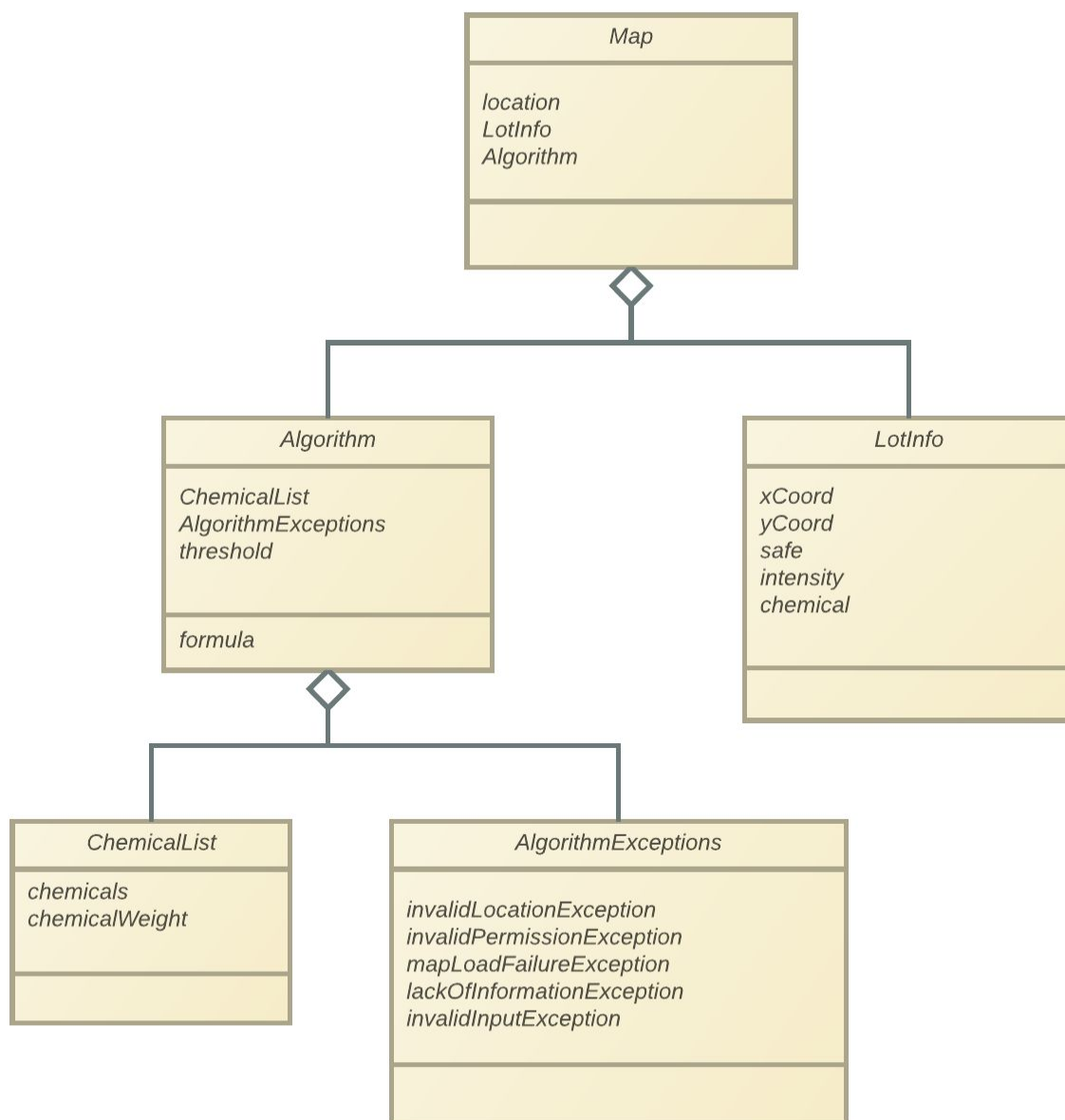
- Iteration: Stage II, last modification: September 24 by Dr. Pulimood
- Primary Actor: Civilian
- Goal in Context: To show detailed information on the pollution in a selected area, including list of chemicals, amount of each chemical, and determined pollution level
- Preconditions: System must be configured, user must be using the map feature, valid area must be selected
- Trigger: User decides to check pollution level at selected location
- Scenario:
 1. User enters SOAP Map
 2. User selects location to check pollution level
 3. Pollution information appears in pop-up window
- Exception:
 1. Map fails to load
 2. User is in a non-valid location
 3. Not enough information at location for algorithm
- Priority: High priority, must be implemented alongside other basic functions
- When Available: Stage V
- Frequency of Use: High Frequency
- Channel to Actor: Via SOAP's Map Feature
- Secondary Actors: Non-profit Organizations, Local Government
- Channels to Secondary Actors: Via SOAP's Map Feature
- Open Issues:
 1. What error message to give if user chooses non-valid location?
 2. What to do if algorithm fails due to lack of appropriate input?

Use Case: Update Prediction Formula

- Iteration: Stage II, last modification: September 24 by Dr. Pulimood
- Primary Actor: Software Developer
- Goal in Context: To update the prediction formula for more accurate predictions.
- Preconditions: User has access to clustering algorithm, user has admin privileges.
- Trigger: Developer makes significant progress on designing improvements to the formula.

- Scenario:
 1. User designs additions to the prediction formula.
 2. User enters SOAP with admin privileges
 3. User inputs formula changes to backend
 4. System accepts formula changes
- Exception: SOAP website is down
- Priority: Average
- When Available: Stage V
- Frequency of Use: Infrequent
- Channel to Actor: Via SOAP admin features
- Secondary Actors: N/A
- Channels to Secondary Actors: N/A
- Open Issues:
 1. How will we push updates?

Analysis Class Diagram:



Elaboration: Design

Detailed Design Class Diagram:

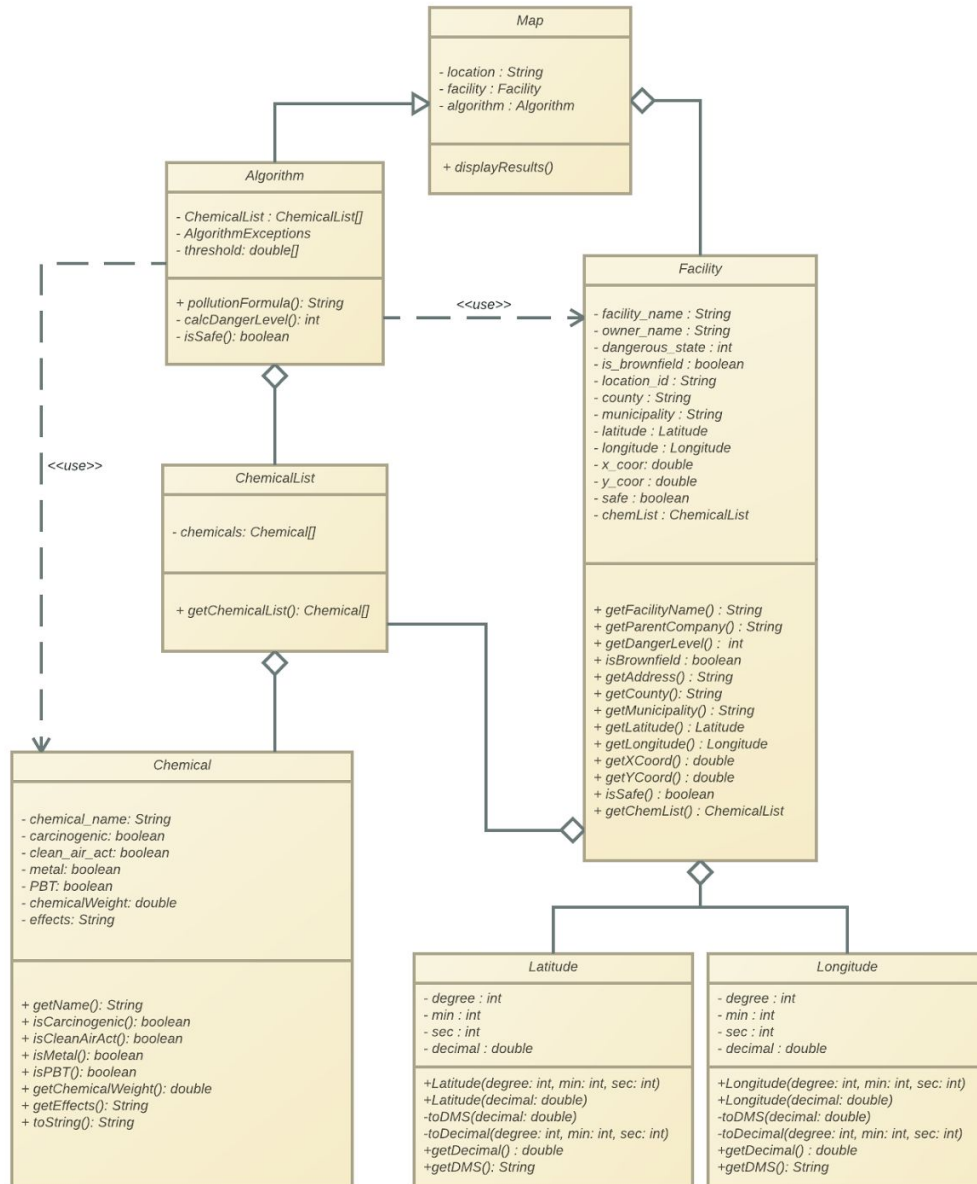


Figure 1. Detailed design class diagram for pollution prediction.

System Sequence Diagram:

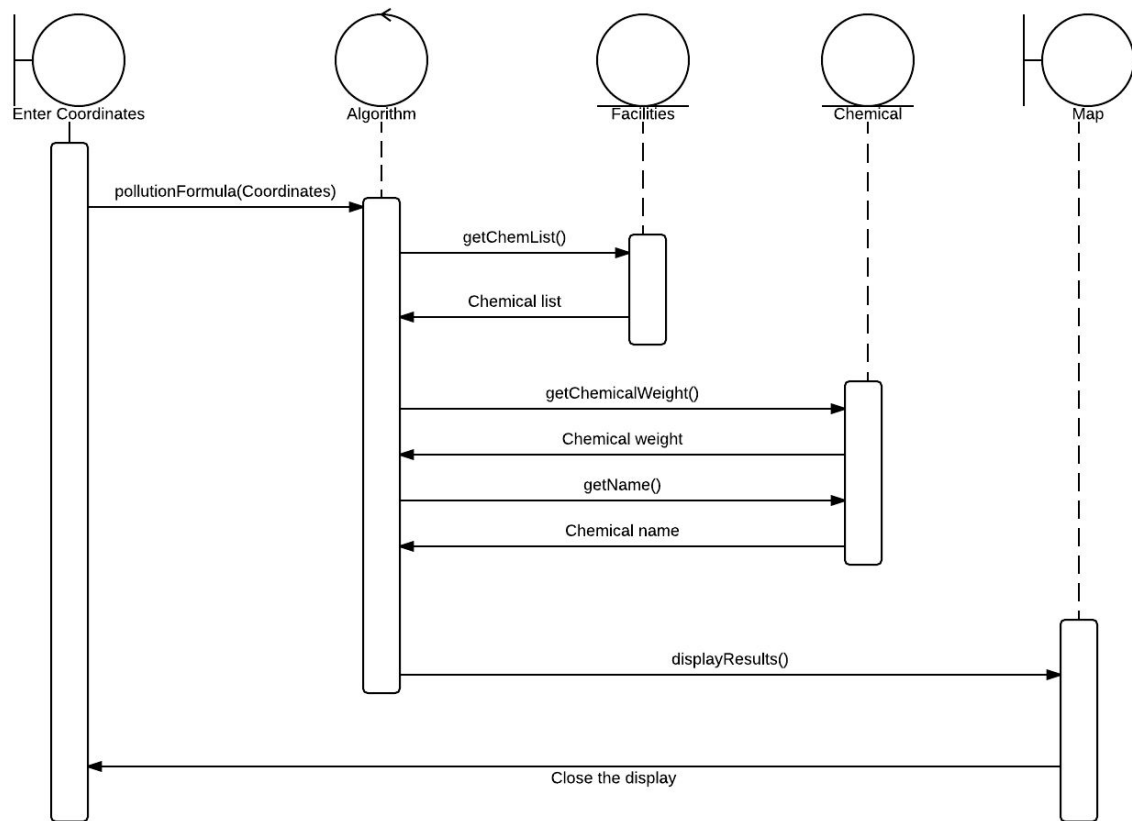


Figure 2. System sequence diagram for pollution prediction.

Detailed Statechart:

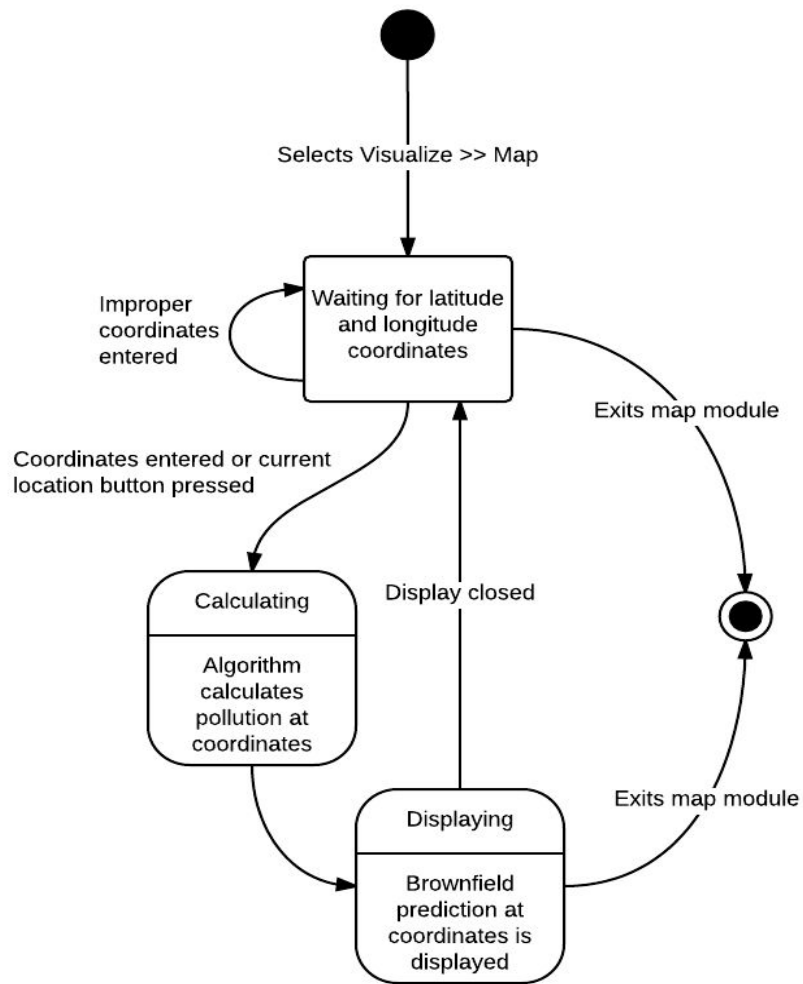


Figure 3. Detailed statechart for predicting pollution at a location.

User Interface Design:

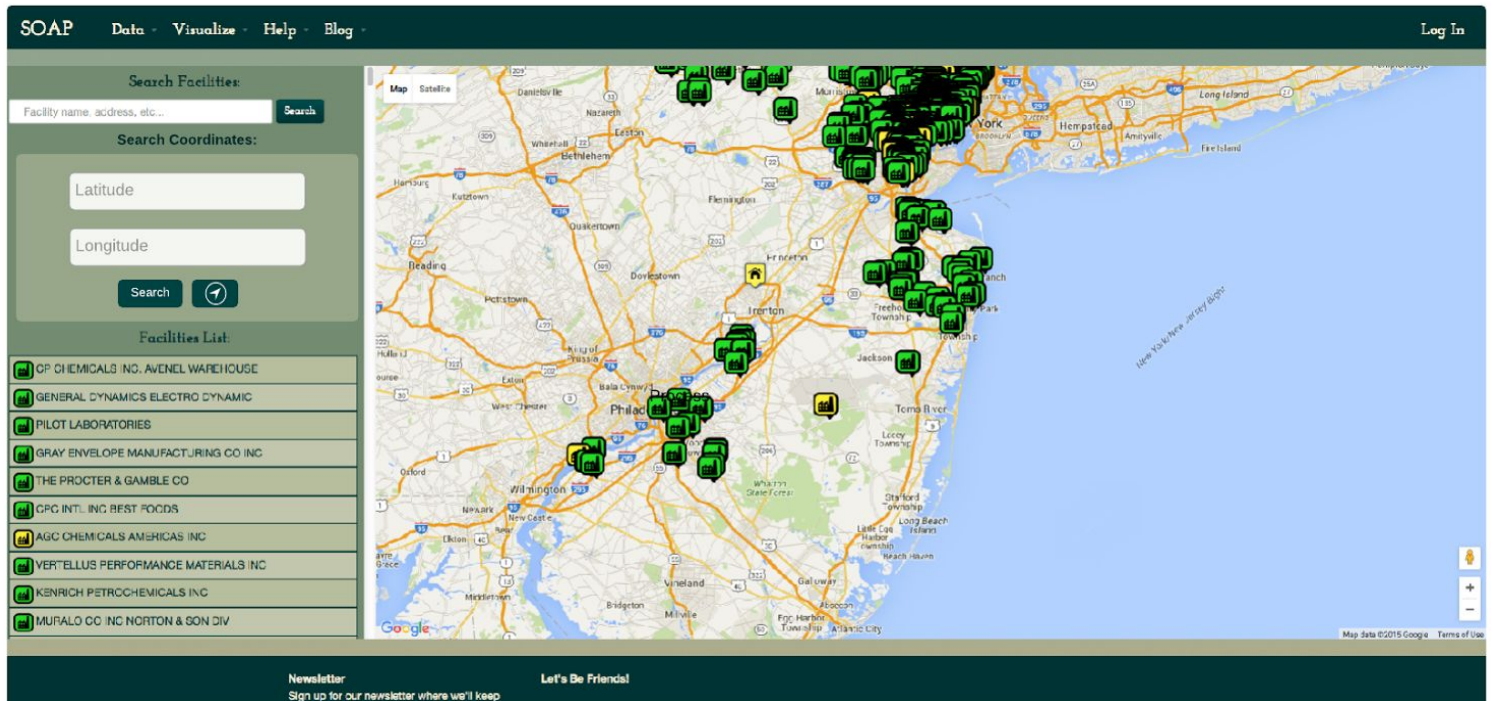


Figure 4. Map module with the ability to search by coordinate.

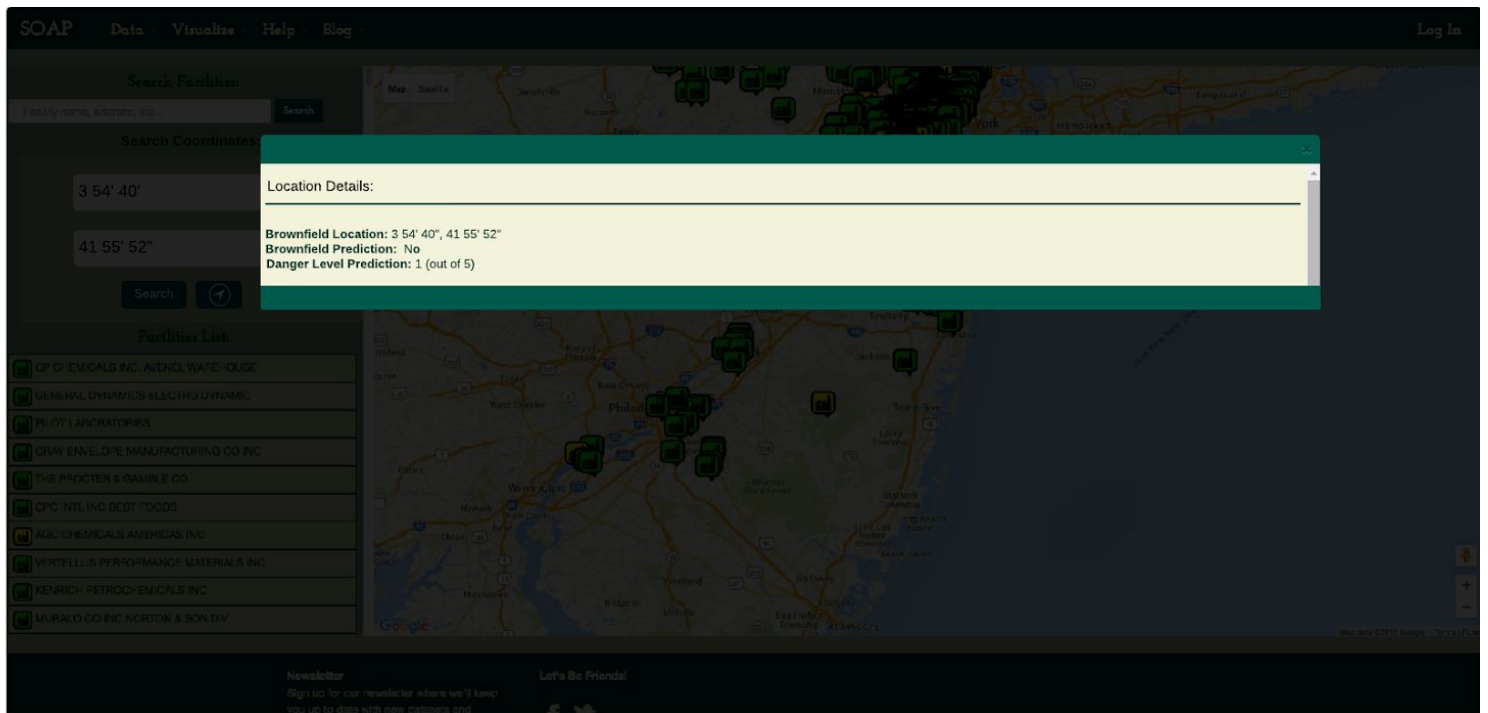


Figure 5. Pollution prediction pop-up window that opens when location is entered.

Strive for Consistency- The first way the user interface strives for consistency is that it maintains a similar color scheme to the current SOAP website. The latitude and longitude text fields used to search for a coordinate also have the same look as the text field used to search for a facility. The new search button and “current location” button also look the same as the other search button on the page.

Enable Frequent Users to Use Shortcuts- Using the current location feature allows the user to quickly determine pollution at their location. This is helpful for a user who needs to estimate the pollution at their location and doesn’t want to keep entering in latitude and longitude values.

Offer Informative Feedback- All entered and processed information will be displayed to the user in real-time when either the search or current location button is pressed. When the pollution prediction window opens it includes the potential brownfield’s location again as informative feedback to the user.

Design Dialogs to Yield Closure- The search functionality will alert the user that the calculation is taking place and present the results once they have been calculated. Therefore, opening the pollution prediction pop-up window lets the user know the calculation is done and serves as dialog to yield closure.

Offer Simple Error Handling- Incorrect coordinates will prompt for another set of correct coordinates. Also, the user will be alerted if the pollution level cannot be predicted at the coordinates specified.

Permit Easy Reversal of Actions- Nothing is being permanently changed when the prediction functionality is used. The lot or coordinates searched will be the only action capable of reversal which will be done by simply entering new coordinates or using the browser’s undo feature to remove the coordinates. The user can also easily close out of the pop-up window that displays the prediction information.

Support Internal Locus of Control- The user feels like they are in control because they are the ones controlling the prediction feature since they have to enter the coordinates or allow the user to use their current location.

Reduce Short-Term Memory Load- This is supported because there is minimal page traversal to be able to use new features. The user only needs to get to the map page to use the functionality. Once the user gets to the map they have two simple choices, enter specific coordinates or use their current location to find the estimated pollution level at that location.

Construction: Implementation

All of the working source code is on the team's GitHub repository and a summary of our work is shown below.

- Learned the Clustering Algorithm under SOAP/app/View/Results/scr. This code was developed by Thomas Borgia as part of the Mentored Undergraduate Research Experience (MUSE). In order to implement the prediction functionality in SOAP, it was critical to understand how the existing algorithm worked. Unfortunately, there were very few comments in this code which made it extremely difficult to use. We decided to spend a large portion of our time commenting this code so that future groups would find it more useful. We tried to comment every line of code in the following files:
 - Commented Chemical.cpp
 - Commented Chemical.h
 - Commented Clustering.cpp
 - Commented Clustering.h
 - Commented Facility.cpp
 - Commented Facility.h
 - Commented PointAnalysis.cpp
 - Commented UpdateClusters.cpp
 - Compiled the C++ files and corrected errors that originally prevented us from doing this.
- Updated search bar on map page
 - We added a latitude and longitude search bar to the map page.
 - Two buttons called "Search" and "Use Current Location" were added below the search bars.
 - These changes were made under SOAP/app/View/Map/index.ctp that
- Made a window open when the search button was pressed
 - A function was added in SOAP/app/webroot/js/map.js that opens a window that is supposed to display the pollution prediction information at valid

longitude and latitude values. An error message appears if invalid values are entered.

- `app/View/Map/coordTest.php`
 - Modified to include function to call `PointAnalysis.exe` file for point prediction
- The SOAP repository that was given to us had issues where the map page was corrupted. The map would not appear and the facilities list was in the wrong location.
 - We believe this was a merge issue because there were syntax errors in `SOAP/app/webroot/js/map.js` that indicated so.
 - We tried to fix these errors for SOAP and were able to successfully get the map to appear along with the facility pins. We also fixed the facility list on this page so that it appeared on the correct part of the page.
 - We were also able to fix an error that was causing the pop-up not to work when clicking on a facility in the list or on the map.
- When the “Use Current Location” button is pressed, the browser asks if you want to share your location.
 - The map also moves over your current location.
 - The latitude and longitude for the current location are also filled in to the appropriate text fields.
 - Code was added to `SOAP/app/webroot/js/map.js`
- When the user goes to the map page and permits location sharing, the map automatically goes to the current location.
 - The latitude and longitude for the current location are also filled in to the appropriate text fields.

Construction: Testing

- The individual testing reports by other students can be found on Canvas

Module Overview:

Our team was able to successfully edit and work on the map page to add new functionality, with room to improve and grow these features in future projects. First, the team needed to fix multiple features of the map page that had previously been broken such as loading the map itself and loading facility information when clicked. These features had been broken by merge conflicts and errors accessing the SOAP databases. Currently, the team has added support for finding the user's current location, appearing when the page is first loaded or when requested via a button. This works as long as the user allows the site to access such info. The latitude and longitude of the current location also appear in text boxes added for the requisite info. Additionally, the team added in basic error handling. If a request for a non numerical latitude or longitude is entered, an error dialog will appear.

Transition: Maintenance

All the source code has been documented with comments that explain the code and states work that needs to be completed for the future.

GitHub Repository:

<https://github.com/TCNJSEteam4/SOAP>

Transition: Product Hand Over

Successful Test Report:

Functionality Tested	Input	Expected Results	Actual Results
Current Location	Initial Page Load (With permission)	Centers map on geographical location in the real world, coordinate inputs filled with lat/long	As expected
Current Location	Clicking "Current Location" button	Centers map on geographical location in the real world, coordinate inputs filled with lat/long	As expected
Current Location	Clicking "Current Location" button (Without permission)	Nothing	As expected
Custom Long/Lat Coords	Lat/Long coords in respective fields	Popup with loading circle	As expected
Custom Coords	Non numerical characters	Dialog box popup requesting valid input	As expected
Map Page	Clicking the "Map" tab	Map page opens with all of the facilities and search bars shown	As expected
Map Page	Clicking on an icon on the map or from the facilities list	Popup window opens with more information about that facility	As expected

Manual:

- Navigate to the Maps page of SOAP
- Upon the page loading you may enable sharing your location to input your current geographical coordinates quickly.
- Alternatively, you can enter your own longitude and latitude to search
 - (use xx.xxxx yy.yyyy, do not use 59°19' 46" N 18°4' 7" E)
- When the search button is pressed, a screen appears that will later hold pollution prediction information.

Presentation Slides: Attached.

SOAP Final Presentation

Hunter Dubel, Richard Levenson, Jeremy Leon,
Evan Melquist, and Zachary Nelson

Table of Contents

- I. Introduction and Rationale for Choosing Module
- II. Map Page Before and After
- III. Module UI and Code Additions
- IV. Clustering Algorithm Comments
- V. coordTest.php: Calling the c++ executable
- VI. Challenges Faced
- VII. Future Work
- VIII. Insights Gained
- IX. Questions

Objectives and Rationale for Choosing Module

- **Goal:** Predict pollution level at a given location (using latitude and longitude)
 - Also include the option to use your current location
- Pollution level was approximated using a clustering algorithm
 - Algorithm was originally created by Thomas Borgia during the Mentored Undergraduate Student Experience (MUSE) at TCNJ
 - Algorithm was written in C++ and was not implemented with SOAP
- We did not want Thomas's work on the algorithm to go to waste
- This was one of the main features that Dr. Caruso wanted implemented

Map Page (Before)



Map Page (After)

[SOAP](#) [Data](#) [Visualize](#) [Help](#) [Blog](#) [Log In](#)

Welcome to SOAP's Map Feature

Please search for properties by name or location in the search bar on the left. Click on a coordinate data point located on the map for specific information about the site, including contaminants and location. View demographic information, such as local minority percentage and income level, by clicking the button above the map on the left.

Go to Address:

Search Coordinates:

Search Facilities:

Filter by County:

Filter by Danger Level:
☒ 1 ☒ 2 ☒ 3 ☒ 4 ☒ 5

Facilities List:


	ECOLAB INC
	CONTINENTAL PLASTICS CO CHEMICALS INC
	CP CHEMICALS INC. AVENEL WAREHOUSE
	GENERAL DYNAMICS ELECTRO DYNAMIC
	PILOT LABORATORIES
	PQ CORP. RAHWAY

5

Latitude and Longitude Search Bars

Search Coordinates:

40.2720413 

-74.7780525 

Search

Use Current Location

Error Handling

Search Coordinates:



Search



Use Current Location

The page at csc415-team04.tcnj.edu says: ×

Please enter a valid longitude and latitude.

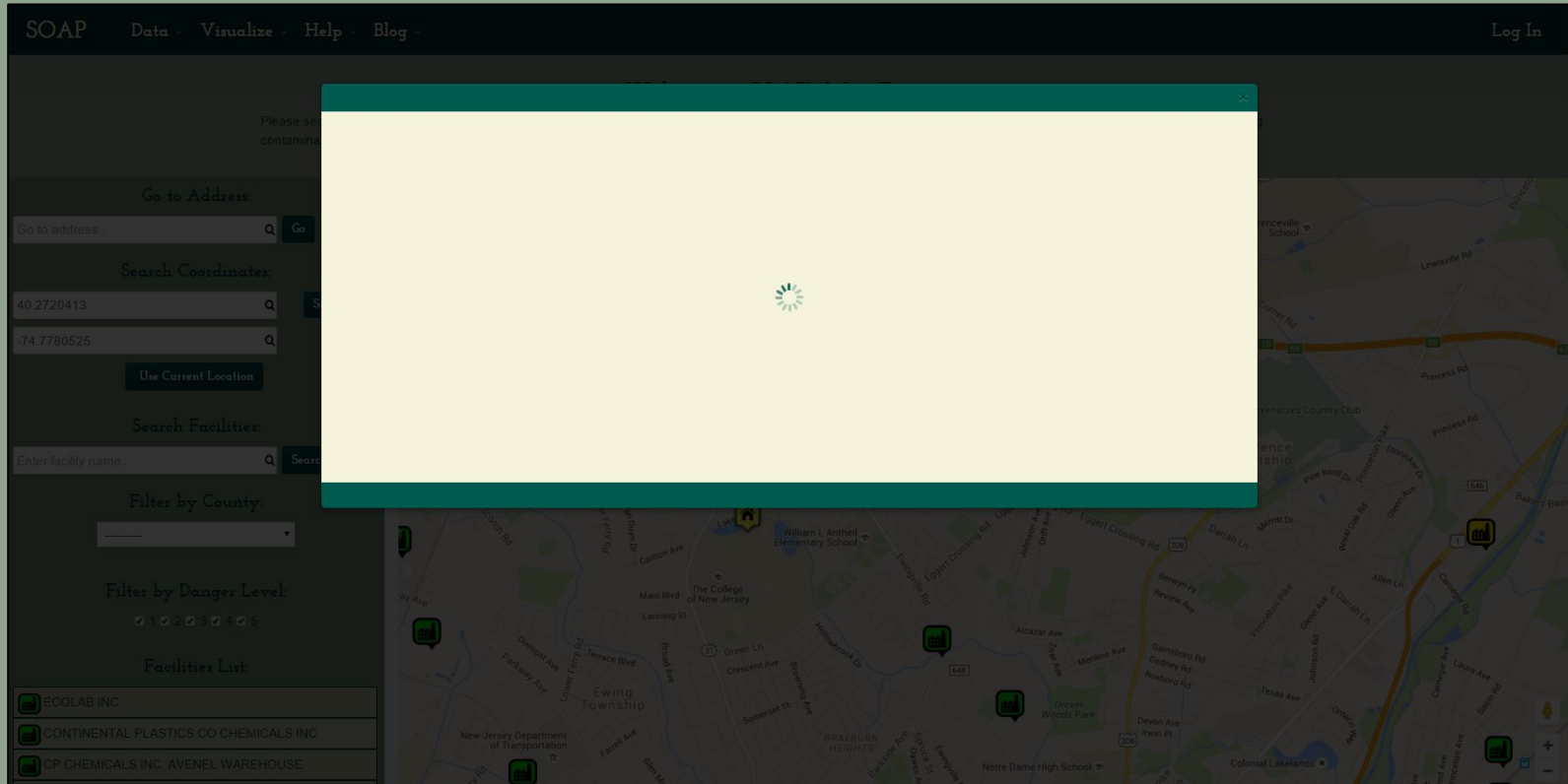
OK

map.js: nonSitePredictor()

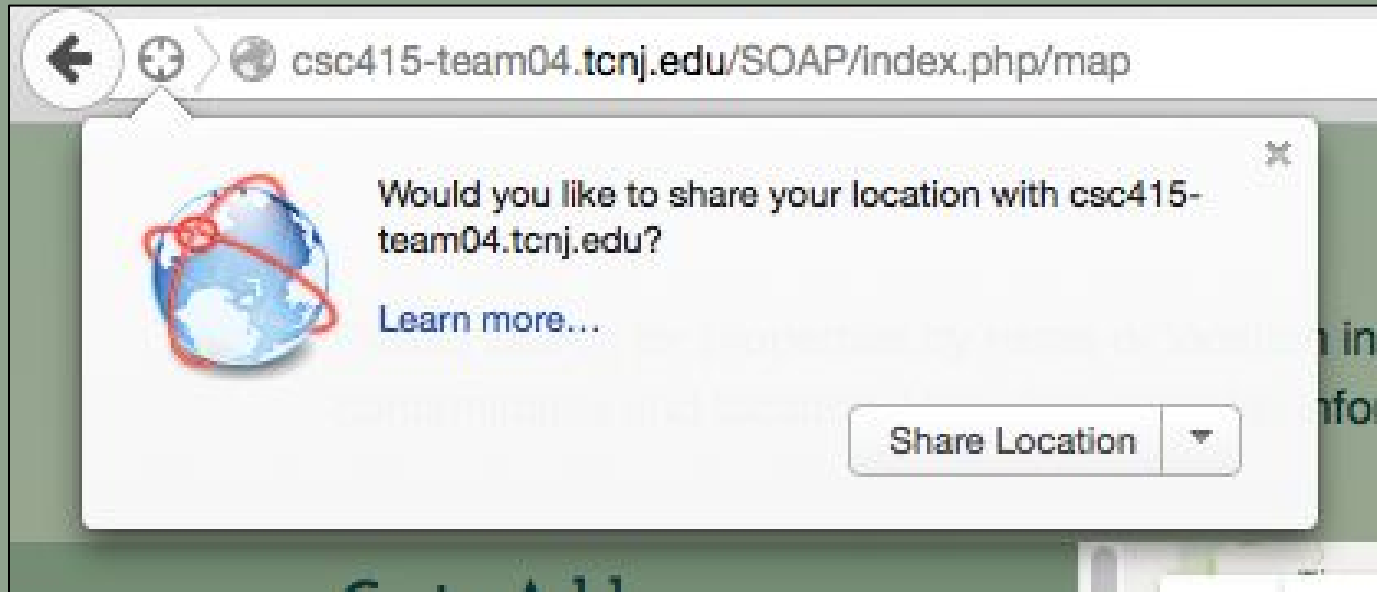
- Functions for handling button clicks (with error handling)

```
231 //Predicts the pollution at a location that is not a known facility
232 //Added Evan Melquist, Jeremy Leon, and Richard Levenson
233 //Modified by Hunter Dubel and Jeremy Leon
234 function nonSitePredictor() {
235     var latitude = document.getElementById("latitudeSearchBar").value;
236     var longitude = document.getElementById("longitudeSearchBar").value;
237
238     if(latitude !== "" && longitude !== "" && typeof parseInt(latitude) === 'number' && parseInt(longitude) === 'number') {
239         $('#mapModal').modal('show');
240
241         $.ajax({
242             type: 'get',
243             url: location.origin + '/SOAP/app/webroot/index.php/map/prediction/' + latitude + longitude,
244             beforeSend: function() {
245                 $("#div#mapModal div.modal-body").empty();
246                 $("#div#mapModal div.modal-body").addClass("loading");
247             },
248             success: function(response) {
249                 $("#div#mapModal div.modal-body").removeClass("loading");
250                 $("#div#mapModal div.modal-body").append(response);
251             }
252         });
253     } else {
254         alert("Please enter a valid longitude and latitude.");
255     }
256 }
```

Pop-up Window



Use Current Location



map.js: goToCurrLoc()

- Functionality for finding current location

```
66 //Centers the map on the user's current location.
67 //If nothing is entered, zooms out and centers on initial position (Trenton, NJ)
68 //SE Fall 2015
69 //Added by Zach Nelson & Hunter Dubel
70 //Modified by Richard Levenson.
71 function goToCurrLoc(position) {
72     mapOptions.initialPosition = new google.maps.LatLng(position.coords.latitude, position.coords.longitude);
73     setInitialPosition(mapOptions);
74     map.setCenter(mapOptions.initialPosition);
75     map.setZoom(15);
76     var latbox = document.getElementById('latitudeSearchBar');
77     var lonbox = document.getElementById('longitudeSearchBar');
78     latbox.value = position.coords.latitude;
79     lonbox.value = position.coords.longitude;
80 }
```

Clustering Algorithm Comments

- Helped us understand algorithm and will help future groups
- Chemical.cpp
 - Object that included chemical name, chemical ID, and total amount of chemical
- Facility.cpp
 - Object that included chemical count, facility ID, facility name, latitude, longitude, and if visited
- Clustering.cpp
 - Creates a vector of clusters and handles writing the information to files for each cluster.
- UpdateClusters.cpp
 - Read data in from facilities.csv and contains.csv and call clustering.cpp on this data
- PointAnalysis.cpp
 - Takes a latitude and longitude point and finds what cluster it belongs to
 - If it finds a cluster, all the chemicals for that cluster are accessed and pollution information is displayed

coordTest.php: Calling the c++ executable

- Includes function to call PointAnalysis executable file

```
17 lines (14 sloc) | 510 Bytes
Raw Blame History
1 <!--
2     Created by Evan Melquist.
3 -->
4 <!--
5     Modified by: Richard Levenson and Hunter Dubel to include function to call PointAnalysis.exe file for point prediction.
6 -->
7
8
9 <?php
10     // Runs the PointAnalysis executable file to find the cluster info of the point
11     // Takes the latitude and longitude strings as input parameters and returns the cluster information.
12     function runPointAnalysis($lat, $long){
13         exec("SOAP/app/View/Results/src/PointAnalysis $lat $long" , $clusterInfo);
14         return $clusterInfo;
15     }
16 ?>
```

Challenges Faced

- Issues setting up the virtual machine
- Past merge issues in map.js and map.css files
- Managing the interactions between various languages
 - php, javascript, c++
- Lack of available documentation

Future Work

- Fixing custom pop-up window
- Fully linking the front end to the C++ algorithm
- Improving the algorithm

Insights Gained

- Deadlines are unpredictable
- Good communication facilitates efficient team development
- Easy things can seem difficult and vice-versa
- Documentation is important for current and future developers

Questions

