Final Report
Fusality for Stream and Field

CS4624
Multimedia, Hypertext, and Information Access

Stephen Bologna-Jill, Kevin Duong, Jason Yongjoo Ha, Jazmine Zurita, Tinsaye Sume, Ryan Smith

Virginia Tech
Blacksburg VA, 24061

Client: Dr. Nicholas Polys
Instructor: Professor Edward A. Fox

April 28, 2017
# Table of Contents

Table of Tables  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ 7  
Table of Figures  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ 7  
Executive Summary  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ 7  

2. User Manual .................................................................................. 7  
  2.1 General Information .................................................................. 7  
  2.1.1 Organization of the Manual ................................................. 7  
  2.2 System Overview ...................................................................... 7  
  2.2.1 System Configuration ........................................................... 7  
  2.2.2 User Access Levels ............................................................... 7  
  2.2.3 Contingencies ...................................................................... 7  
  2.3 Getting Started .................................................................... 8  
  2.3.1 Installation and Logging In ................................................... 8  
  2.3.2 Creating an Account ............................................................. 8  
  2.3.3 System Menu ..................................................................... 8  
  2.4 Using the System ................................................................ 9  
  2.4.1 Experiments ................................................................... 9  

3. Developer’s Manual ...................................................................... 10  
  3.1 Saving Graphs .................................................................... 11  
  3.2 Adding/Editing Graphs ........................................................... 12  
  3.3 Laying Out Graphs ................................................................. 13  
  3.4 Importing data into the database .............................................. 13  

4. Project Timeline ......................................................................... 15  

5. Lessons Learned ......................................................................... 16  

6. Acknowledgments ....................................................................... 18  

7. References ................................................................................ 18  

8. Appendix A ............................................................................... 19  

1. Executive Summary ................................................................ 25  

2. Introduction ............................................................................. 26  
  2.1 Goals ................................................................................ 26  
  2.2 Scope ................................................................................ 26  
  2.3 Major constraints ................................................................. 26  

3. Overview .................................................................................. 27  
  3.1 User Roles and Responsibilities ............................................. 27  
  3.2 Interactions with Other Systems ............................................ 28  
  3.3 Expansion of Current System ................................................ 28  

4. Functional Requirements ........................................................... 30  
  4.1 Statement of Functionality .................................................... 30  
  4.2 Usability ........................................................................... 30  
  4.3 Model Description ................................................................. 30  
  4.4 User Interface Design ......................................................... 33  

5. Database Design ....................................................................... 35  
  5.1 Database Description ............................................................ 35  
  5.2 Data Description ................................................................. 35
Table of Tables
4.1.1 Implementation Phases ................................................................. 14
Table of Figures

2.4.1.1 Experiments page with a look at how previous experiments are outlined .... 8
2.4.1.2 Graph creation page ................................................................................. 9
1.2.1 A closer look at the graph creation options ........................................... 11
3.3.1 Plotting graphs .......................................................................................... 12
1. Executive Summary

This report provides a detailed review of the work that was done to create and edit the StREAM website. This website provides users with a means to organize, graph, and analyze specific data recorded from Stroubles Creek. The website will be utilized by an Undergraduate Biological Systems Engineering class to help with their labs that deal with the health of Stroubles Creek.

Our team was designated with the task of improving a website that was created by a past Computer Science capstone team. The website we started with was barely functional and could not yet be used by the Undergraduate Biological Systems Engineering class. The website required many modifications in both the front end interface as well as the backend. Our team split up into three two-person groups based off of skill and desired learning objectives. These teams include a backend team, a front end user-interface team, and a data graphing team.

The main front-end improvements that were enacted on the website include a complete overhaul of the entire user-interface and the addition of a usable navigation bar that enables users to easily use all features of the website. Backend improvements include major changes to the tables in the MySQL database as well as PHP functions that make utilizing the database extremely easy for the data graphing team. The changes made to the database tables allowed for a more straightforward representation of the data and enabled saving graphs for a specific experiment.

Most of the improvements were on the data graphing aspect of the website. Users are now able to analyze six years of data collected from Stroubles Creek. They can analyze this data by creating either line graphs or scatter plots of whatever specific creek data they want. The graphs provide users with the ability to see trends in creek health over the course of many years.

Currently, the website is ready to be used by Undergraduate Biological Systems Engineering classes. It provides all the functionality that our client required and does so in a clean, easy-to-use manner.

Even though the website is ready for use, there are still areas that can be improved upon. These areas include more graph options, easier ways to upload new data sets, graphing large amounts of data points, and the aesthetics of the graphs.
2. User Manual

2.1 General Information
This section explains in general terms the system as well as how to use and interact with it.

2.1.1 Organization of the Manual
The user’s manual consists of four sections: General Information, System Summary, Getting Started, and Using the System.

The General Information section provides a simple definition of the system as well as its intended use. The System Overview section gives a more defined representation of the system. It examines software requirements, system configuration, user access, and system behavior. Getting Started walks through how to run the Web application. Using the System describes the system functionality.

2.2 System Overview
StREAM Lab is a web application that allows students in Biological Systems Engineering students to store and analyze data collected from Stroubles Creek. The application then saves the data onto the database. StREAM Lab is integrated to work on any web browser.

This section highlights the software requirements, system configuration and user access levels.

2.2.1 System Configuration
StREAM Lab runs on the latest versions of web browsers. As such, this application requires an Internet connection to operate fully and save graph information onto the database.

2.2.2 User Access Levels
Anyone with access to a web browser can access this site, but must log in to handle experiments and store data onto the database.

2.2.3 Contingencies
If there is no Internet connection, any data recorded will not be saved onto the database. Furthermore, if there is a power outage, any data that has been added will also not be recorded onto the database.

2.3 Getting Started
This section goes over how to fully operate and install components that enable the StREAM Lab web application to flow seamlessly on any web browser.
2.3.1 Installation and Logging In
The first step to run StREAM Lab is to become familiar with XAMPP, an application that helps the website run concurrently with the database.

Installing XAMPP
1. Install XAMPP at https://www.apachefriends.org/index.html
2. On the XAMPP control panel, find the XAMPP folder. Once in this folder, find the htdocs folder
3. Unzip the project file and place it into the htdocs folder. The name of the unzipped file is not necessarily important but ‘stream’ is recommended.
4. Now on the control panel, go to the Manage Servers tab. Turn on MySQL Database and Apache Web Server using the start button.
5. Open a web browser and go to http://localhost/phpmyadmin/
6. Click ‘new’ and create a new database, which should be named ‘hci_capstone’
7. Click the new database file, then click Import. Next click Choose File. Now find the unzipped file, open the file, then open the hci_capstone.SQL file and click ‘go’.
8. Go into the xampp folder → htdocs → stream/whatever you named the folder → app → config file. After opening the config file, change line 5 to ‘define (‘BASE_URL’, ‘http://localhost/stream/’);’ or replace stream with what you named the folder
9. Open another tab or window in the browser and go to http://localhost/stream/ or replace stream with what you named the folder

Congratulations! The website should now be running. Now, find the login button and create an account.

2.3.2 Creating an Account
To create a new account, go to the login page and hover to the bottom. Click on the link to create a new account. Now you will see a sign up page where you can fill in your desired username and password. You will also have to put your date of birth to secure the account. After completing these steps, you can click ‘join’ and start adding experiments.

2.3.3 System Menu
The menu for the website is transparent to blend well with the image on the home page and additionally remains fixed on the top left, regardless of how far down the user navigates through a page. When the user is not logged in, the menu will show the option to go to the home page or the login page. Once logged in, the user can also see the experiments page, which is where they can record the data from Troubles Creek. Additionally, once the user is logged in, the menu will show the option to log out. Once the user clicks ‘logout’ they are redirected to the main home page.
2.4 Using the System
   This section gives a more detailed look into the system functionality.

2.4.1 Experiments
   Once you are logged in and ready to record experiments, you can go to the experiments page where you will be able to visualize the data with graphs and see previous experiments.

Figure 2.4.1.1 Experiments page with a look at how previous experiments are outlined
When you first go to the experiments page, you will see ‘Previous Experiments’ at the bottom. As illustrated with Figure 1, this is where you will be able to go back and look at past experiments you recorded. These will be recorded by the date and content added. Before you begin recording graphs, the first step is to make a title and then write content regarding the experiment. As mentioned, this step is crucial in order to search and find a specific previous experiment. After clicking the submit button, you will be taken to the graph creation page of Experiments.

Figure 2.4.1.2 Graph creation page

You should now be seeing everything shown in Figure 2.4.1.2. The first thing to do is choose the dates or date at which the experiment happened. Next, you can pick what type of data you want to represent in your graph by clicking the on/off switches to the left of each data type. After selecting the data types, you can pick which graph you would like to use to show the data. Clicking the graph button will show you what the selected graph looks like given the data types selected. Lastly, if you want to add the graph onto the database, you can click the save graph button.

3. Developer’s Manual

Files we have made/changed:

- experiments.tpl - this is where all the graph creation code is as well as code from the past group

- hci_capstone.sql - this is the SQL file to import into MySQL to have the tables we need for the website
- getGraphData.php - this is the code that retrieves the saved graph data

- saveGraph.php - this is the code that saves graphs to the database

- overview.tpl – this is the experiments overview page code that shows all experiments created by the user logged in

- login.tpl – this is the login page code that lets users log in to the website

- signup.tpl – this is the signup page code that lets new users sign up an account for the website

- home.tpl – this is the home page code that contains links to the login page and the experiments overview page

### 3.1 Saving Graphs

Saving graphs are done by putting the user selected, date range, data types, and graph type into an JSON object and inserting it to the database. When a user creates a graph, they have to select a startDate, endDate, discharge option, dis oxygen option, sp condition option, turbidity option, pH option, temperature option, and graphType. All of this data is put into a JSON object and attached to the graph’s block `<div>` (see adding new graphs) when the user presses ‘Graph’. When the user hits the ‘Save Graph’ button, each block `<div>` on the page is iterated through to get its corresponding graph JSON object which is put into an array. This array is then stringified and sent to saveGraphs.php to save the graphs in the database. saveGraphs.php takes in an array of JSON in a string format and the experiment ID. It then decodes the array of JSON string into an array of JSON_objects which are our array of graphs. Next, a mysql transaction is started. This transaction first clears all graphs currently in the experiment and then iterates through the array of JSON to insert the graphs into the database. If none of the insertions or deletions failed, the transaction will commit, otherwise, it will rollback.

When the page is reloaded after having graphs saved to it, the lists of graph from that experiment is taken from the database through the getGraphData.php file and the function createGraph() is called to recreate the graphs.
### 3.2 Adding/Editing Graphs

To add new graph types, developers must first know how the graphs are being stored and transferred between functions. Given the following figure, to create a single graph, users must select a date range, data types involved within that graph, and the desired type of graph.

![Graph Creation Options](image-url)

**Figure 2.2.1 A closer look at the graph creation options**

The date ranges are stored as 2 Javascript date objects, a start_date and an end_date. Dates within this file need to be frequently switched between two formats, one for mysql to read and one for Javascript to read. For convenience to switch between these two formats, two methods are provided, toJSDate(d) in which d is the date in mysql format and Date.prototype.toMysqlFormat() in which this method can be called on a Javascript Date object.

For the types of data, each type has a 0 or 1 flag in which 1 means that the user selected it and wants to include it within their graph.

Lastly, the graph types are stored as a variable containing numbers between 0-3. 0 is the plot line graph, 1 is the double y-axis graph, and etc. If a developer wants to add another graph type, store that graphType as 4 and the next as 5, etc.

Creating graphs are done in the experiment.tpl file under /app/view. If the developer wants to make a new graph type available to the users, first create a radio button for that graph type and begin editing the onclick function for that radio set which is named #graph. Create a createGraph() call section within the onclick with the given graphType. Then go to the createGraph method and create a section for your own new function.

Your createGraph function requires a few things to keep the program fully functional. First of all, it requires the same parameters as the already given graphs in the
function. The data from the sql will be handed in through the parameters and will need to be formatted to fit your graphing library’s needs. All data within the specified date range is included. It holds the date and each of the data types as arrays. The rest of the parameters are then required for the createNewGraphContainer() function. The createNewGraphContainer() is called so that the graph can be displayed on our custom divs. Note that the createNewGraphContainer function takes in the start and end dates as a mysqlformat date. createNewGraphContainer() is necessary to call since it also attaches the graph as a JSON object onto the div to be used for saving.

3.3 Laying Out Graphs

Graphs are laid out inside <svg>’s that are within <div>’s named block + the graph’s corresponding number. This block div is also within another div container along with an x button named the container + the graph’s corresponding number in the list. Giving the graphs numbers based on where they are in the list allows for us to keep track of which graph needs to be deleted when the x button is clicked. The numbers are generated by a count initialized to 0 when the page is opened. As more graphs are being created, the count increases.

![Plotting graphs](image.png)

Figure 3.3.1 Plotting graphs

3.4 Importing data into the database

Importing data into the database as of right now requires the developer to have a program that has MySQL, like XAMPP, or MySQL directly. Then, the developer must execute these SQL commands:

To create a new table:
create table `tablename` (  
  date varchar(100),  
  discharge double,  
  stage double,  
  DO_pct double,  
  DO double,  
  pH double,  
  spCond double,  
  temp double,  
  turbidity double  
); (feel free to change the data to whatever you need in the table)

To insert data into a table from a CSV file (this inputs null data if the data is blank):  
LOAD DATA LOCAL INFILE  
'csv path'  
INTO TABLE  
`tablename`  
FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n' IGNORE 1 LINES (the above ignores the header of the CSV file, it is not necessary if it only has data points)  
  (@vone, @vtwo, @vthree, @vfour, @vfive, @vsix, @vseven, @veight, @vnine)  
SET  
  date = nullif(@vone,""),  
  discharge = nullif(@vtwo,""),  
  stage = nullif(@vthree,""),  
  DO_pct = nullif(@vfour,""),  
  DO = nullif(@vfive,""),  
  pH = nullif(@vsix,""),  
  spCond = nullif(@vseven,""),  
  temp = nullif(@veight,""),  
  turbidity = nullif(@vnine,"");  
(feel free to change the data to whatever you need in the table)

To change the date to a datetime if not that way already (do them one at a time):  
alter table `tablename` add column `name2` datetime;  
update `tablename` set `name2` = str_to_date(`name1`, '%m/%d/%Y %H:%i');  
alter table `tablename` drop column `name1`; (only necessary if you don’t need the varchar dates, however the code we have does use it, so it’s not recommend)

alter table `tablename` change `name2` `name1` datetime not null; (changes the name of a column if necessary)
## 4. Project Timeline

### Table 4.1.1 Implementation Phases

<table>
<thead>
<tr>
<th>Phase</th>
<th>Tasks</th>
<th>Start Date</th>
<th>Planned Completion</th>
<th>Status</th>
</tr>
</thead>
</table>
| 1     | a. Have a meeting with Dr. Polys to understand the outline of the project  
      b. Group meeting to come up with different types of project goals  
      c. Meeting with Dr. Polys to finalize the final goal of the project | Jan. 27 | Feb. 2 | Complete |
| 2     | a. Have multiple group meeting to figure out specific requirements of the project  
      b. Create overall design of the project  
      c. Divide work within the group  
      d. Look into the past project | Feb. 6 | Feb. 16 | Complete |
| 3     | a. Start developing the code  
      b. Provide demo to Dr. Polys and receive feedback | Feb. 20 | Mar. 15 | Completed |
| 4     | a. Depending on the feedback develop the project  
      b. Provide demo to public (other students) for feedback  
      c. Provide demo to Dr. Polys for feedback | Mar. 20 | Apr. 12 | In progress |
5. Lessons Learned

a. Problem/ Solutions

**Problem:** While working with creating graphs, we used the plotly.js library to create one of the graphs. While meeting with Dr. Polys, Dr. Polys reminded us about the licensing of the online libraries. Since the plotly.js library was based on D3.js library, which is an open source library, we thought plotly.js was also going to be open source. However, we were told that plotly.js might not be an open source library.

**Solution:** We first thought about getting rid of the Plot.ly graph from the project. However, when we did more research on Plotly.js, we found out that plotly.js is an open source library. Anyone is allowed to download, change, or update plotly.js. Since it was recommended by Dr. Polys, we also created an extra D3 graph.

**Problem:** We had a little confusion on the data files that we received. The first data file contained data sets of discharge, pH, sp condition, temperature, and turbidity for a week. However, the second data file had completely different data sets. The second data file contained three years of discharged, stage.stage_m, and stage.stage_m. Due to two completely different data files, restarting the project was a possibility.

**Solution:** When we had our meeting with Dr. Polys, we were able to figure out why two files were so different. The data sets were pulled from two different time period. It was possible that the second file we received was a wrong file. Therefore, through Dr. Polys, we contacted the BSC professor for a new data file. On 23-Apr, we received a new data file from the BSC professor.

**Problem:** When we first received the project and navigated through the website, there were a lot of features that were not working. Sometimes, it was very hard to understand the purpose and usage of some features. We had a better understanding of the website when we read the final report from the past group, but that is not going to be the case for the actual user.

**Solution:** As a group, we redesigned the website and spend more time on developing the user interface. We removed unnecessary functions and fixed all the errors from the previous project. We also conducted an informal pilot survey to make sure if our redesigned website is more effective than the original website. Adding a scatter plot is a result from the feedback.

**Problem:** It was a challenge to combine our work together. This is because all everyone’s updated code was based on the original code. Therefore, when we met together to combine the code together, the website as a whole provided multiple errors
and problems. Due to the ineffective way of combining our work, fixing errors and problems took too long.

**Solution:** Our first solution was to update often by having more meeting. However, this solution was not effective. Therefore, we used GitHub to share and update files online. Due to this change, everyone was able to begin their work from the most updated version of the project. Later, we moved our project from GitHub to Gitlab.

**Problem:** We will need to learn new tools because this project delves into topics that most members of the team have not dealt with previously. For instance, we will need to learn how to create a Python script to load the data from a CSV file, how to use MySQL to create a database that can be queried by the web page, how to use Xampp, and how to add 2D and 3D graphs on the web page.

**Solution:** We have found tutorials and YouTube videos that helped us understand the new materials. Also, some of our group members created how to document for other members to use.

**Problem:** When saving graphs, we found that sometimes all graphs would be saved, 1 graph would be saved, or none would be saved at all.

**Solution:** The way we were saving previously was to delete all graphs for the specified experiment in one PHP file and use another PHP file to insert the graphs. The insert PHP file would insert one graph per call so the number of times we called it was the same as the number of graphs that needed to be saved. We came to the conclusion that this was probably creating race conditions in the MySQL database in which inserting and deleting were happening concurrently. To fix this, we created 1 saving PHP file in which it started a transaction which would first delete, then insert one graph at a time before committing. If any of these SQL insertions failed, the whole call would rollback before anything was called and return an error.

b. Future work

Although graphs do provide much information to the users, equations and values also provide a lot of information to the users. Therefore, we thought it will be useful to provide some important values or equations to the users. We could include R², mean, standard deviation, mode, median, etc. We could also include equations such as y = mx + b. This should help students to analyze the data more effectively. When providing mean, standard deviation, mode, and median, it would be nice to provide a bell curve with the probabilistic data.

Another possibility is to provide a data table. If a user selects a time period, the website could provide a data table instead of a graph. It could be helpful if a student could look at the data table and a graph side by side.

Currently, the website cannot graph if there are too much data values. The website cannot handle more than a year and half worth of data. This is because of data overflow. A solution to this problem is to have a user select a time interval. For example, if a user selects ‘weekly’, then the website pulls a data value every week.
In order to input a data file, such as nextyear.csv, the user has to visit the database server to add a new table to the database. If a user can add and remove the data files from the website, it will be easier to manage the website.

6. Acknowledgments

We would like to thank Riley Babcock, Derek Messer, and Hanna Vess for their contributions to the 2016 StREAM website. We would also like to thank Dr. Nicholas F. Polys and Dr. Edward A. Fox for working closely with us to shape the vision of the new website.

7. References


<http://metagrid2.sv.vt.edu/~lance/proxy_test_Stroubles_Fall2016/Bridge1_SphereFeed2.html>.

<https://d3js.org/>.

<https://www.youtube.com/watch?v=xdvVKywGlc0>.

<https://sites.google.com/a/vt.edu/fusality-for-stream-and-field/>.


<https://www.youtube.com/watch?v=w428fByczE>.

<https://www.youtube.com/watch?v=ZdP0KM49IVk>.

<https://www.youtube.com/watch?v=jQ9aDyBWCXI>.

<https://www.youtube.com/watch?v=ueWpNe0PG34>.


8. Appendix A

Other than this final report, there were three reports due throughout the course of this project. The first was the Requirements and Design Report, second was the Implementation Report, and the third was the Prototyping, Refinement, and Testing Report. Each report built on the previous report. This meant that we had to make edits in each report for the previous. The report shown in this appendix is the final Prototyping, Refinement, and Testing report. It contains the previous two reports as well as edits for all three reports.
Prototype, Refinement, Testing Report
Fusality for Stream and Field
CS4624
Multimedia, Hypertext, and Information Access
Stephen Bologna-Jill, Kevin Duong, Jason Yongjoo Ha, Jazmine Zurita, Tinsaye Sume,
Ryan Smith
Virginia Tech
Blacksburg VA, 24061
Client: Dr. Nicholas Polys
Instructor: Professor Edward A. Fox
Date: March 31, 2017
Table of Contents
1. Executive Summary.................................................................25
2. Introduction.................................................................................26
  2.1 Goals .....................................................................................26
  2.2 Scope......................................................................................26
  2.3 Major constraints ....................................................................26
3. Overview......................................................................................27
  3.1 User Roles and Responsibilities ..............................................27
  3.2 Interactions with Other Systems ..............................................28
  3.3 Expansion of Current System ..................................................28
4. Functional Requirements ..............................................................30
  4.1 Statement of Functionality ......................................................30
  4.2 Usability ..................................................................................30
  4.3 Model Description ..................................................................30
  4.4 User Interface Design ..............................................................33
5. Database Design............................................................................35
  5.1 Database Description ..............................................................35
  5.2 Data Description .....................................................................35
  5.3 Possible Design Solutions ......................................................36
    5.3.1 MySQL Function .................................................................36
    5.3.2 Python Script .....................................................................36
6. Graphs...........................................................................................36
  6.1 Current graph ..........................................................................36
  6.2 Graph Preference ....................................................................37
7. Management Overview.................................................................37
  7.1 Description of Implementation ...............................................37
  7.2 Points-of-Contact ..................................................................38
  7.3 Major Tasks .............................................................................38
  7.4 Implementation Schedule .......................................................39
  7.5 Security and Privacy ...............................................................40
8. Implementation Support ...............................................................40
  8.1 Software, Facilities, and Materials ..........................................40
    8.1.1 Software ...........................................................................40
    8.1.2 Facilities ...........................................................................41
  8.2 Personnel ................................................................................41
    8.2.1 Staffing Requirements .......................................................41
9. Implementation Requirements by Site .........................................41
  9.1 Site Requirements ..................................................................42
10. Prototyping....................................................................................42
  10.1 Experiments Page ..................................................................42
  10.3 Database ...............................................................................43
11. Refinement...................................................................................43
  11.1 Navbar adjustments ..............................................................43
  11.2 Adding an image for the home page .......................................44
  11.3 Revisiting the navbar ............................................................44
  11.4 Client revision .......................................................................45
12. Testing & Refinement ................................................................. 45
  12.1 Main User Class ................................................................. 45
  12.2 Goals .................................................................................. 46
  12.3 Testing Plan ................................................................. 46
  12.4 Testing Results ................................................................. 47
  12.5 Refinement ........................................................................ 48
13. References ............................................................................ 49
Table of Tables
3.1.1 Responsibilities ................................................................. 25
5.2.1 Sample Data ........................................................................ 34
7.1.1 Implementation Phases ....................................................... 36
7.2.1 Points-of-Contact ................................................................. 37
12.7.1 Heuristics ........................................................................ 45
Table of Figures
3.3.1 Current Stream Lab Home Page ................................................................. 27
3.3.2 Current User Log In ................................................................................. 27
3.3.3 User Welcome Page ................................................................................. 27
3.3.4 Current Experiments Main Page ............................................................... 28
3.3.5 Current Experiments Creation Page ......................................................... 28
3.3.6 Current Sample Graph Template ............................................................... 28
4.3.1 Creek Data ............................................................................................... 29
4.3.2 Graph ........................................................................................................ 29
4.3.3 User .......................................................................................................... 29
4.3.4 Experiment ............................................................................................... 30
4.3.5 ER Diagram of Web Application’s Database ........................................... 31
4.4.1 Wireframe of Experiments Main Page ..................................................... 32
4.4.2 Wireframe of Experiments Creation Page .............................................. 33
10.2.1 Example Graph ....................................................................................... 41
11.1.1 Minimal Design With Hamburger Menu .............................................. 42
11.1.2 Expanded View of Hamburger Menu .................................................... 43
11.3.3 Updated Home Page ............................................................................. 44
1. Executive Summary

This report provides a detailed review of the work that was done to create and edit the StREAM website. This website provides users with a means to organize, graph, and analyze specific data recorded from Stroubles Creek. The website will be utilized by an Undergraduate Biological Systems Engineering class in the near future to help with their labs that deal with the health of Stroubles Creek.

Our team was designated with the task of improving a website that was created by a past CS 4624 project team. The website we started with was barely functional and could not yet be used by the Undergraduate Biological Systems Engineering class. The website required many modifications in both the front end interface as well as the backend. Our team split up into three two-person groups based off of skill and desired learning objectives. These teams include a backend team, a front end user-interface team, and a data graphing team.

Our team went through a series of phases that helped us create a deliverable on time that met the client’s specifications. The first phase was a design a requirements phase where we met with our client to understand exactly what he needed from us and how we should go about getting it done on time. The second phase was an implementation and prototyping phase where we actually made the changes and improvements to the existing website. By prototyping as we made changes, we ensured that what we were creating was exactly what the client wanted. The last phase was the deliverable phase where we made final changes and polished the final website for actual user use. By following these phases, we were able to make effective use of our time and efforts which allowed us to create the final deliverable.

The main front-end improvements that were enacted on the website include a complete overhaul of the entire user-interface and the addition of a user-friendly navigation bar that enables users to easily use all features of the website.

Backend improvements include major changes to the tables in the MySQL database as well as PHP functions that make utilizing the database extremely easy for the data graphing team. The changes made to the database tables allowed for a more straightforward representation of the data and enabled the ability to save graphs for a specific experiment.

The most improvements were done on the data graphing aspect of the website. Users are now able to analyze six years of data collected from Stroubles Creek. They can analyze this data by creating either line graphs or scatter plots of whatever specific creek data they want. The graphs provide users with the ability to see trends in creek health over the course of many years.

Currently, the website is ready to be used by Undergraduate Biological Systems Engineering classes. It provides all the functionality that our client required and does so in a clean, easy-to-use manner.

Even though the website is ready for use, there are still areas that can be improved upon. These areas include more graph options, easier ways to upload new data sets, graphing large amounts of data points, and the aesthetics of the graphs.
2. Introduction

2.1 Goals

By the end of the semester, we envision having a website that will have a functional database, data visualization through graphs that present the relationships on all the samples collected, and an improved interface. We hope to respond to unresolved questions to the last group (that made this website) such as what a better workspace would look like. It is important to note that while we are trying to make this website more interactive, we are not pressed on the aesthetics of our design. Our primary focus, as discussed with Dr. Polys, is to make the database work and have visual representations of the data accumulated through the years. The current application consists of a site in which users can log into and query data within a time range of a data set to create a line graph. Our objective is to improve on this application by first migrating the data storage from a CSV file to a relational database. The intended future for the website is to have Dr. Polys install the 3D Blacksburg project onto the site and have the intended audience be able to log their own information, although this extends further than our time with StreamField.

To run the website in the way our client has expressed, we want to make full use of D3.js, since it is one of the best tools available for data visualization. Our concern here is that the intended audience gets the necessary information from these graphs. We want to make sure that the visualized data is not only interactive but also relevant to what the audience needs to know. Our database is going to be built using MySQL, and our main goal for this database is to have it structured so it is easy to make adjustments in the future. Another consideration is the fact that the website is currently running under PHP (with which our team lacks experience). We aim to have a website that can flow seamlessly with PHP, along with other languages we are more comfortable with like Python and JavaScript.

2.2 Scope

This application is mainly used to aid Biological Systems Engineering students in visualizing data gathered from a creek. Students will be able to create graphs and compare different data sets for lab experiments and lectures. Furthermore, while we discussed the future of this website with Dr. Polys, we came to an understanding that his full project would not be included in our set of tasks. Due to the scope of the project, we will only work on the general improvement of the database and user interface. The 3D Blacksburg incorporated aspect will be handled in the future.

2.3 Major constraints

The three major constraints for this project are learning new tools, understanding the implementation from the previous project, and length of time available.

We will need to learn new tools because this project delves into topics that most members of the team have not dealt with previously. For instance, we will need to learn how to create a Python script to load the data from a CSV file, how to use MySQL to create a database that can be queried by the web page, how to use xampp, and how to add
2D and 3D graphs on the web page while also allowing the user to interact with them. In order to overcome this learning curve as quickly as possible, we have found tutorials and YouTube videos that will be able to get us up to speed. The tutorials and YouTube videos we will start with are listed below.

Tutorials:
- XAMPP Tutorial: How to Use XAMPP to Run Your Own Web Server from https://blog.udemy.com/xampp-tutorial/
- D3 Tutorial from https://www.dashingd3js.com/table-of-contents

YouTube
- Manipulating data using the csv module in Python
- Connect to MySQL with PHP in XAMPP / Create a new database

Second, as previously stated, this project was started by another group, which has graduated. This means there is much code and documentation that our group is not familiar with. On top of this, not everyone in our group is experienced with web page development. Once we start understanding the PHP code we will start understanding how the web page works, how it is connected to a database, and the aims of the prior project.

The final major constraint we are anticipating is time. There are many parts to this project; getting all of the parts working separately then bringing them together for the final web page could prove to be time consuming. In addition, there are multiple deadlines and reports due throughout the lifespan of this project. To stay on top of all of this, we are setting our own deadlines before the reports are due and will meet with Dr. Polys every few weeks.

3. Overview
3.1 User Roles and Responsibilities

<table>
<thead>
<tr>
<th>First Name</th>
<th>Last Name</th>
<th>Job Title</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stephen</td>
<td>Bologna</td>
<td>Query Interface Lead</td>
<td>○ Streamline ability to draw data from database ○ Make web page and data more usable</td>
</tr>
<tr>
<td></td>
<td>Jill</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jazmine</td>
<td>Zurita</td>
<td>Query Interface Lead</td>
<td>○ Streamline ability to draw data from database ○ Make web page and data more usable</td>
</tr>
<tr>
<td>Kevin</td>
<td>Duong</td>
<td>Graph Developer Lead</td>
<td>○ Add more chart options to current website ○ Line chart, scatter plot, pie chart, pictographs, etc. ○ Compare multiple data sets through graphs ○ Compare between two graphs ○ Create one graph that contains multiple data sets ○ Understand relations between data sets and variables to visualize data</td>
</tr>
<tr>
<td>Yongjoo</td>
<td>Ha</td>
<td>User Interface Lead</td>
<td>○ Making sure graphs are user-friendly ○ Ensure it is easy to add data and remove data from the graph that was already built</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Tinsaye</th>
<th>Sume</th>
<th>Backend Database Lead</th>
</tr>
</thead>
<tbody>
<tr>
<td>○ Ensure design and setup of the graph should be clear</td>
<td></td>
<td></td>
</tr>
<tr>
<td>○ Graph creation should be easy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>○ Put 2 years of stream data into a database using MySQL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>○ Convert CSV files</td>
<td></td>
<td></td>
</tr>
<tr>
<td>○ Write a Python script to load the data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>○ Store graph information in database</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ryan</th>
<th>Smith</th>
<th>Backend Database Lead</th>
</tr>
</thead>
<tbody>
<tr>
<td>○ Put 2 years of stream data into a database using MySQL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>○ Convert CSV files</td>
<td></td>
<td></td>
</tr>
<tr>
<td>○ Write a Python script to load the data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>○ Store graph information in database</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.2 Interactions with Other Systems

This project should accomplish two things. First, it should produce a public web page that could be used by BSE 3324 students for their labs and lectures. The students should be able to create an account and access a database to create a graph that could be used to analyze the data. The second goal is to update 3D Blacksburg. 3D Blacksburg is a program that creates a 3D view, like Google Earth, of Virginia Tech and the Blacksburg community. Our goal is to provide graphs that could be used to analyze the stream that flows across Blacksburg. While navigating through the creek in 3D Blacksburg, users should be able to view the graphs that contain information about the creek. The goal of our web page is to create graphs that could be used within the 3D Blacksburg program.

3.3 Expansion of Current System

As previously stated, this project has been started by a previous group. They created a web application with a limited view of the data as the graphs do not show every connection among the multiple variables collected. We will expand on these views by adding different graphs that will show various relationships within the data. These new graphs will also be able to query new data that spans over two years. In order to do this, the data will have to be inserted into a database from its current location on a CSV file.

There will be improvements on the previous web page as well. The previous web page is represented by Figures 3.3.1, 3.3.2, 3.3.3, 3.3.4, and 3.3.5. Figure 3.3.1 shows the initial screen that is displayed when the web page is loaded. Figure 3.3.2 shows the login page that is displayed when the user clicks login on the top navigation bar. Once the user
logs in, they are brought to a welcome page that is displayed in Figure 3.3.3. Figure 3.3.4 displays the page that is loaded when “experiments” in the navigation bar is clicked by the user. From this page, users can see their previous experiments or create a new one by typing in a title and content and then clicking submit. Submit then brings the user to the experiment creation page. While on this page the user can create a graph and save his or her work. The current graph representation is represented by Figure 3.3.6.

The current graph representation does not take into consideration null values of data gathered from the creek. When the data is null, the graph points down to 0, thus creating an uneven graph. The best way to handle this will be to simply not include the points when the data is null.
4. Functional Requirements

4.1 Statement of Functionality

The system will use a MySQL database to hold up to 10 years of data gathered from the creek every 10-15 minutes. A script will need to be made to transfer the current 2 years of data within the CSV file to the new MySQL database. We will also create tables to enable the save functionality which was attempted last semester to work as it was meant to, since it was discovered that currently it does not save the information which needs to be stored.

The system will be able to create an experiment which can then hold multiple graphs created from the creek data. Students will then be able to leave comments and save the graphs for the experiment to come back to.

4.2 Usability

A student should be able to create their own account. When logged into a student account, the students are able to edit their user profile and delete their user account in the profile page. When students visit the experiment page, they are able to create a new experiment or view the previous experiments. For each experiment, students are able to create, view, save, and delete graphs. The data for the graphs are already uploaded in the database. The students are allowed to access the database, but they are not allowed to upload or edit the data.

4.3 Model Description

The following tables will be updated/created in the current MySQL database for the website. This schema was decided upon so that the users will be able to save their work if they develop a graph that shows what they were desiring to see.

<table>
<thead>
<tr>
<th>Creek Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id NOT NULL PRIMARY KEY</td>
</tr>
<tr>
<td>Discharge</td>
</tr>
<tr>
<td>Dis Oxygen</td>
</tr>
<tr>
<td>SpCond</td>
</tr>
<tr>
<td>Turbulence</td>
</tr>
<tr>
<td>PH</td>
</tr>
<tr>
<td>Temperature</td>
</tr>
<tr>
<td>Date</td>
</tr>
</tbody>
</table>

Figure 4.3.1 Creek Data Entity Attributes

<table>
<thead>
<tr>
<th>Graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id NOT NULL PRIMARY KEY</td>
</tr>
<tr>
<td>Type NOT NULL</td>
</tr>
<tr>
<td>ExperimentId NOT NULL</td>
</tr>
</tbody>
</table>

Figure 4.3.2 Graph Entity Attributes
User

<table>
<thead>
<tr>
<th>Id NOT NULL PRIMARY KEY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password NOT NULL</td>
</tr>
</tbody>
</table>

Figure 4.3.3 User Entity Attributes

Experiment

<table>
<thead>
<tr>
<th>ID NOT NULL PRIMARY KEY</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserId NOT NULL</td>
</tr>
<tr>
<td>StartDate</td>
</tr>
<tr>
<td>EndDate</td>
</tr>
<tr>
<td>DateCreated</td>
</tr>
<tr>
<td>Comment</td>
</tr>
<tr>
<td>Title</td>
</tr>
</tbody>
</table>

Figure 4.3.4 Experiment Entity Attributes

Creek Data

The Creek Data table, Figure 4.3.1, represents the two years of data that has already been collected. It will be used by the Graphs the Users create in each experiment. A tuple of Creek Data will have the following attributes:

- **Id** – A unique number that will allow the tuple to be accessed.
- **Discharge** - The volume of water that passes a given location at the given time. Expressed in cubic feet per second.
- **Dis Oxygen** – The concentration of dissolved oxygen in the creek at the given time. Expressed in milligrams per liter.
- **SpCond** – This stands for the specific conductivity, or how conductive, the water is. It is measured in milli-siemens per centimeter (mS/cm).
- **Turbulence** - is a flow regime characterized by chaotic changes in pressure and flow velocity. Expressed in milligrams per liter.
- **PH** - a measure of the relative acidity or alkalinity of water. Water with a pH of 7 is neutral; lower pH levels indicate increasing acidity, while pH levels higher than 7 indicate increasingly basic solutions.
- **Temperature** – Measured in Celsius.
- **Date** – The time in which all measurements were taken for this given tuple.

Graph

A Graph, Figure 4.3.2, will be created by a user in an experiment. There can be multiple Graphs in one experiment. The type of Graph that is chosen by the user will be stored. A tuple of Graph will have the following attributes:

- **Id** – A unique number that will allow the tuple to be accessed.
- **Type** – The type of graph that is displayed, for instance a bar or line graph.
- **ExperimentId** – The Id of the Experiment the graph was created in.
An Experiment, Figure 4.3.3, will be created by a user. An Experiment will store the Id of the user that created the Experiment so that it can be saved and accessed again. The user will specify the range of dates to be accessed for all graphs in the Experiment. A tuple of Experiment will have the following attributes:

- **Id** – A unique number that will allow the tuple to be accessed.
- **UserId** – The Id of the user that created the experiment and that can access this experiment.
- **StartDate** – The beginning of the Creek Data to be displayed.
- **EndDate** – The end of the Creek Data to be displayed.
- **DateCreated** – The date the user created the graph.
- **Comment** – The user will be able to store comments for the saved experiment
- **Title** – The user will be able to give a title to the experiment

**User**

A User, see Figure 4.3.4, will be able to create and access Experiments associated his or her Id. When signing in the user will be required to type in the password that is the same as the one corresponding to their Id. A tuple of User will have the following attributes:

- **Id** – Unique name the user has created.
- **Password** – Required for the user to login to his or her account.

The ER Diagram displayed in Figure 4.3.5 shows the relationships between all of these entities, as well as their attributes.

![Figure 4.3.5 ER Diagram of Web Application’s Database](image-url)
4.4 User Interface Design

There are a few issues with the current user interface that we hope to address. The first is the overall design and usability of the web page. In an effort to improve the aesthetics of this web page, we will experiment with more complimenting colors as the current colors are not very easy on the eyes and make adjustments to the CSS since the current aligning for the text boxes and buttons are not centered and the styling is very plain. Another way we are going to improve the aesthetics is by adding more interactive features to the web page. The current user interface does not support the functionality we are planning to have and as such we will be adding buttons and text fields to correspond to the various additional functions.

We will also be redesigning the menu bar on the top of the screen. One of the biggest changes we will make is with the ‘Experiments’ tab, seen in Figure 4.4.1. Currently when this tab is hovered over, nothing appears. Our plan is to design the website so that when a user hovers over this tab, a drop down menu will appear with two options, create new and previous, the latter of which will show the previous experiments and their data. This facilitates the process for users who know they want to create a new experiment so as to make one less click while still allowing users who are not sure what they want, to just go to the main ‘Experiments’ page. We hope to add functionality to the experiments creation page that allows the user to pick data and graph that data in different ways. Our goal for this page is shown in Figure 4.4.2. We also plan to improve ease of use by following guidance from Bootstrap (http://getbootstrap.com/).

![Figure 4.4.1 Wireframe of Experiments Main Page](image-url)
Figure 4.4.2 Wireframe of Experiments Creation Page
5. Database Design

We have access to roughly two years of stream data. Altogether, this data adds up to more than 3 gigabytes of information. The system needs to have a resilient back end database that can not only handle large amounts of data, but also support rapid access to this data.

5.1 Database Description

This system already implements a MySQL database on the back end. The current system uses one table for all of the data which will be unable to handle the large amount of data we plan to add to the system. To solve this problem, we plan to add more tables to the existing database that will properly separate the data.

5.2 Data Description

The data has multiple different properties; see examples in Table 5.2.1. Columns represent variables, while the values are measurements of these variables for collected samples.

<table>
<thead>
<tr>
<th>date</th>
<th>discharge</th>
<th>stage</th>
<th>DO_pct</th>
<th>DO</th>
<th>pH</th>
<th>spCond</th>
<th>temp.</th>
<th>turbidity</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/23/2013 2:30</td>
<td>0.120017</td>
<td>0.156</td>
<td>77.3</td>
<td>9.77</td>
<td>7.61</td>
<td>0.719</td>
<td>5.34</td>
<td>8.2</td>
</tr>
<tr>
<td>3/23/2013 2:40</td>
<td>0.118492</td>
<td>0.155</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/23/2013 2:45</td>
<td></td>
<td></td>
<td>77.4</td>
<td>9.78</td>
<td>7.61</td>
<td>0.719</td>
<td>5.36</td>
<td>9.2</td>
</tr>
<tr>
<td>3/23/2013 2:50</td>
<td>0.116976</td>
<td>0.154</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/23/2013 3:00</td>
<td>0.120017</td>
<td>0.156</td>
<td>77.6</td>
<td>9.78</td>
<td>7.61</td>
<td>0.719</td>
<td>5.39</td>
<td>8.8</td>
</tr>
<tr>
<td>3/23/2013 3:10</td>
<td>0.118492</td>
<td>0.155</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/23/2013 3:15</td>
<td></td>
<td></td>
<td>77.6</td>
<td>9.79</td>
<td>7.61</td>
<td>0.719</td>
<td>5.4</td>
<td>8.2</td>
</tr>
</tbody>
</table>

As shown in Figure 5.2.1, some data records do not have a value for every single property, which may be due to the rate at which the data was being gathered. This is another factor that could affect how we should manage these incomplete records when
we implement the database. We could solve this issue by merging an incomplete row with the row before or after it, since it seems to be an issue of timing.

5.3 Possible Design Solutions

5.3.1 MySQL Function

Since the data we currently have is in CSV format, it would be ideal if we could easily import all of it into a MySQL table. One of the team members has done exactly this in the past with a built-in SQL command. This command allows us to format the table just like it is formatted in the CSV, then directly inject all the data into a new or pre-existing MySQL table. If we use this and it is successful, it will solve the issue. For normalization, we will look into splitting the data into several tables.

5.3.2 Python Script

We would like to automate the process of importing data of this size, since doing it one by one would be very time consuming. Making a Python or similar scripting language script can help us save time, while making sure we get all the data in the table. The data will only need to be added to the database during the setup of the web application. Consequently, our MySQL ingest routine will only need to be run once. But during this run Python script will need to result in the creation of Creek Data entities with attributes filled with the properly parse the CSV file.

6. Graphs

6.1 Current graph

Currently, our web page only supports plain line graphs. When date range and type of graph is selected, a line graph pops up on the screen. It is possible to open multiple graphs, but it is not clear how graphs are saved. It is also possible to comment. In order to determine whether or not these features are proper requirements for our system, we developed user scenarios:

Scenario 1:

Emma, a Biological Systems Engineering PhD student, is conducting research on Troubles Creek. She wants to determine if there is some sort of relationship between the pH and turbulence of the water. Emma starts by selecting the date range she wants to examine, then a type of graph to display the data. She doesn’t see any relationship. However, there are multiple ways she can examine the data so she continues to tinker with the display. Finally, Emma finds the perfect display to see the relationship between pH and turbulence. She wants to share her findings with her supervising professor so she saves her graph and logs out of the system. Scenario 2:
Timmy, a Biological Systems Engineering undergraduate student, has a few minutes between classes to work on his homework. His homework asks him to find a direct relationship between two of the measurements that were recorded. Timmy finds the proper relationship just before his next class starts. He doesn’t have time to submit so he makes a comment on the graph then saves it so he can come back to it later.

These scenarios were based off of the projected use of this web page provided to us by our client. After some discussion over these scenarios, we came to the conclusion that these were proper requirements for the graphs. Further, we discovered from our scenarios that it would be very useful to show multiple graphs on the experiment page. Consequently, we have added this feature to our list of requirements.

6.2 Graph Preference

We’re planning to create at least 3 more basic chart types for the application: bar chart, combo chart, and scatter plot. To create these charts we will use our main guide for this project, as mentioned before, D3.js (https://d3js.org/), since it is a helpful data visualization tool.

7. Management Overview

7.1 Description of Implementation

Our first job for this project is to study and understand the past work done by the last capstone group. Since we had a group member who took CS 3744 Intro GUI Programming/Graphic last semester, we were able to figure out how to run the past project on our local machine. We took the following steps to run the past project.

1. Download and unzip the file provided from Dr. Polys
2. Download and install xampp
3. Run XAMPP and start Apache and MySQL
4. Open http://localhost/phpmyadmin/ and import SQL file
5. Open http://localhost/NameOfWebSite/ to view the project

To improve our project we will have our weekly meetings on Monday. Also, we will have 5 major phases as explained in Table 7.1.1.

<table>
<thead>
<tr>
<th>Phase</th>
<th>Tasks</th>
<th>Start Date</th>
<th>Planned Completion</th>
<th>Status</th>
</tr>
</thead>
</table>
| 1     | a. Have a meeting with Dr. Polys to understand the outline of the project  
      |       | Jan. 27    | Feb. 2             | Complete |
|       | b. Group meeting to come up with different types of project goals  
      |       |            |                    |         |
|       | c. Meeting with Dr. Polys to finalize the |            |                    |         |
## 7.2 Points-of-Contact

Table 7.2.1 describes supporting personnel for this project, with contact information.

<table>
<thead>
<tr>
<th>Role</th>
<th>Name</th>
<th>Contact</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>Dr. Polys</td>
<td><a href="mailto:npolys@vt.edu">npolys@vt.edu</a></td>
</tr>
<tr>
<td>Sponsor</td>
<td>Dr. Fox</td>
<td><a href="mailto:fox@vt.edu">fox@vt.edu</a></td>
</tr>
</tbody>
</table>

### 7.3 Major Tasks

Task include addressing possible changes needed following system implementation, e.g., personnel and technology equipment alignment and contractor support.

- Reviews
  - Dr. Polys
    - For every phase, we hope for a review and feedback from Dr. Polys.
    - We will visit him as a group and get a feedback on each section of the project.
- This will take place in the conference room in Visionarium.

  - Group Review
    - Group members who took CS 3724 Human-Computer Interaction, have experience conducting analytic evaluations.
    - We will utilize this experience and conduct a heuristic evaluation.

- User Interface
  - We plan to update the overall look of the website
    - We will do this by utilizing the add-ons that Bootstrap supports like using toggles instead of regular buttons.
  - We will improve the layout of the web page
    - We will adjust the color scheme using the “A Guide to Color, UX, and Conversion Rates” article written by Hannah Alvarez as a reference.
    - We will adjust the button layout to make the interface more user friendly. We will measure this by conducting a heuristic evaluation that utilizes heuristics based around our goals.

- Database
  - We will input the new data needed in the database.
  - We will fix any issues with the data already in the database.
  - We will investigate other ways to organize the data to be more efficient.

- Graphs
  - Line graphs
    - We will create simple line graph (time vs. others).
    - We will add a cumulative line graph (time vs. temp, pH, etc.).
  - Scatter Plot
    - Comparing temp, pH, etc. to each other at given time
  - Bar Graph
    - It might not be useful if we are only comparing stream data only. However, if we compare stream data with other data (such as water, acid, rain, etc.), this graph might be useful.

### 7.4 Implementation Schedule

1. Weekly meeting
   a. We will meet every Monday (10 am)
b. Location for meeting will be library or computer science lounge

c. If needed, we will have other meetings throughout the week

2. Major meeting and due date

   a. Jan 27, First meeting with Dr. Polys
   b. Feb 2, Second meeting with Dr. Polys
   c. Feb 19, Requirements + Design Report Due
   d. Feb 22, Get past project running in local machine
   e. Mar 13, Developing the website
   f. Mar 15, Third meeting with Dr. Polys
   g. Mar 17, Implementation Report
   h. Apr 9, Developing Project
   i. Apr 10, Feedback from students
   j. Apr 10, Fourth meeting with Dr. Polys
   k. Apr 11, Prototyping, Refinement, and Testing Report
   l. Apr 23, Finalizing the project
   m. Apr 24, Fifth meeting with Dr. Polys
   n. Apr 25, Final Report
   o. Apr 27, Presentation

7.5 Security and Privacy

   There is no need for strong security or privacy. Therefore, the only security we will have for this web page is a user log in. A student must log into the website in order to view their work. Each student is only allowed to view experiments that he or she created. We will recommend to each student that they should avoid simple passwords.

8. Implementation Support

8.1 Software, Facilities, and Materials

8.1.1 Software

   The software required to support the implementation of our system includes MySQL, XAMPP, d3.v3.min.js, nv.d3.min.js, and Bootstrap.
MySQL v5.7.17 is a Relational Database Management System that is needed to store creek information, user information, the experiments, and the graphs. https://www.mysql.com/ can be used as a reference.

XAMPP v5.6.30 will be the PHP development environment that we will be using to run the website locally. https://www.apachefriends.org/index.html can be used as a reference.

d3.v3.min.js and nv.d3.min.js are JavaScript libraries that will be used to create the graphs needed for this project. These graphs include line graphs and scatter plots. https://d3js.org/ can be used as a reference.

Bootstrap v3.3.6 is a JavaScript framework that will be used for the UI of the website. http://getbootstrap.com/ can be used as a reference.

8.1.2 Facilities

Our team has many different facilities at our disposal to aid with the development of our project. Our main facility that we use is the Visionarium and the side rooms associated with that. Dr. Polys has given us access to these rooms because they are not only close to him, but they also have large screens that we can use, free workstations, and a place with no outside interruptions during team meetings or client interactions.

8.2 Personnel

8.2.1 Staffing Requirements

Tinsaye and Ryan have been selected to work on the Database portion of the project. Their responsibilities include creating tables, editing tables, migrating CSV Creek Data to MySQL, and queries required to create the graphs.

Kevin and Yongjoo have been selected to work on the graphs portion of the project which includes improving the kinds of graphs that will be available and the ways the datasets can be compared and visualized. They will set up a scatter plot for comparing data sets against each other, and they will also set up a better line graph so that many trends can be seen on one graph.

Stephen and Jazmine have been selected to work on the frontend UI portion of the project. Their responsibilities are to mainly utilize the capabilities of Bootstrap in order to improve the overall look and feel of the website. For example a lot of the buttons and text are not formatted in any specific way. Jazmine and Stephen will change use Bootstrap to change this. They will also improve on the way querying will be made to generate the graphs.

9. Implementation Requirements by Site
9.1 Site Requirements  
We will use XAMPP (https://www.apachefriends.org/index.html) because it provides the necessary tools, Apache and MySQL, to develop a website that can be connected to a database. To develop the front end, we will use Bootstrap (http://getbootstrap.com/). For the graphs, we will use d3.v3.min.js and nv.d3.min.js. The data requirements are to have the provided data from the client into the database. There are no facilities requirements.

10. Prototyping

10.1 Experiments Page  
The main changes from the very first prototype we had were made in the actual business code that ran the website. The “Experiments” page received the most changes because this page contained a lot of the features and functionality that the client wanted to have finished. The Experiments page, as shown in Figure 10.1.2, now allows users to graph different datasets on one graph. This allows a user to analyze multiple different aspects of stream data all at once.

The graphs allow for quick and effective analysis of the data sets. The users are also able to select a specific date range. This enables users to only view the data they are actually interested in. These graphs were created through D3.js and another graphing library called Plotly (https://plot.ly/).
10.3 Database

The previous system pulled information directly from the CSV file every time a query for the data was made. This had a negative effect on the usability of the web page because this type of data gathering was very time consuming. To solve this issue, we created a table in the database called Creek Data. We then used the MySQL import feature to put the data in the table from the CSV file. The CSV file we used was also different from the original because it has 6 years’ worth of data while the previous CSV file only had a week’s worth of data. This new CSV file did have some blank data points which would have caused errors when trying to display it. To solve, this issue we set the blank data to NULL so that it would not appear in the graphs.

Another big change from the previous system is that the graphs can be properly saved. The previous system had a table in the database for saving graphs but never actually implemented it. This table did not have the proper attributes so we created a new one that contained startDate, endDate, discharge, DO, pH, spCond, temp, turbidity, graphType, experimentID. Now every time we save a graph, the database saves these data points of the graph and then pulls that data and makes the same graph again when the page is loaded.

11. Refinement

11.1 Navbar adjustments

After prototyping the website, we realized the navbar was less useful than expected. We had focused more on the layout and overall user experience of the pages themselves as well as functionality and usability. Some pages like the settings page had the options to edit or delete user information but the links sent Error 404s. Additionally, the profile page only really had the username and name, the first of which showed the path directory up to the current username and the latter of which showed an error. The profile page did not add anything of weight to the website, so we removed it. For the purpose of testing, the settings page was taken out as well. It is our intention to bring back a version of the settings page, but at this stage we are not as focused on maintaining this in comparison to the experiments page. Revising the navbar, we were now left with three main links: the home page, the experiments page, and the login page. With only three options when the user was logged in, and two when the user was not logged in, we decided to try a navbar that was minimalistic. We chose a hamburger menu, similar to what most mobile versions use for their websites, where the user can click on the hamburger and the links open as a result. We considered this a great design because it saved more space on both the login page and home page, as shown in Figure 11.1.1 and 11.1.2 respectively, on each of the pages and gave the website a better aesthetic.
11.2 Adding an image for the home page
After changing the navbar, we decided to add one image of Stroubles Creek as seen in Figure 11.1.2 because it gave the home page a better visual of the location and also happened to be the best picture from the three pictures available on the VT site for the project. This process turned out to take a little longer than we had expected because of how the website was connected to XAMPP. We had to store the image in a different folder, outside of the project folder we were in, to have it displayed on the home page.

11.3 Revisiting the navbar
The hamburger menu was a great simplification to the original navbar, however upon discussing it further with our team a point was brought up that the hamburger menu was probably better for a website that had more than three links. One person brought up that there was no need to keep these links hidden since there were not enough to necessarily hide. The argument was that it was one extra click that the user had to make for a small action. We also discussed more at length on whether or not the hamburger would confuse users because of how much less common it was to find one on a website versus a mobile version. For these reasons we decided to just have a menu that would show options without the hamburger. It was agreed to be transparent to blend enough with the image on the home page while still standing out at the right amount. As is shown in Figure 11.3.3, the web page now shows all the links on the side with a transparent background to adjust better with the image. Notably, this does not affect the text written on the pages. Foreseeing this issue, the text layouts were tightened on the right and left margins so as to not clash with the new navbar when the user scrolled up and down the page.
11.4 Client revision
After pinpointing changes to the prototype, we met with our client. He was pleased with the new look of the website, but advised that we adjust our graphs to support more open source tools like D3js graphs. At that point, we were using a D3 graph and a Plot.ly graph to show the results from the weekly data we had received. Because we were using student licenses, we were not aware that most of the Plot.ly sources required $30/month to maintain on a website when they were not used on student accounts. Because the ultimate goal of the website is to have it running on the Virginia Tech server, this would be expensive. Another feature we were reminded of was the save function in the experiments page, as it can save the graph locally, but not on the database. His two main concerns for this new website were saving graphs and using more of D3’s capabilities instead of Plot.ly. We followed his suggestions and looked for graphs that were in line with D3. We also had to revise how we thought of saving graphs as the old version of our website had saved graphs locally but not to the database. Discussing this issue with Professor Polys, we concluded that the graphs were to be saved and stored into the database. This way, all of the data can be stored and retrieved with ease as well as be contained in the same location.

12. Testing & Refinement

Once our team completed the implementation plan, we finally had a prototype that was ready to be tested. We then had to come up with a testing plan that would maximize the information we could obtain. To make sure the plan that we created was complete and thorough, we had to consider our users, and the goals of our testing.

12.1 Main User Class
We determined that our main user class is biological systems engineering students that will be new to the system. We know this because our system will be used in advanced biological systems courses. Having the main user class defined allows us to ensure the results from our testing are actually useful.
12.2 Goals

We determined that we have three main goals for our testing. First is the ease of use for this system. This is vital to the success of our work because we don’t want our users wasting time thinking about navigating and understanding our system; we want them focusing on the data and their work. This leads us to our next goal of ease of finding correlations in data. We want to see if we improved this aspect of the system in addition to adding different ways to display the stream data. Our final main goal of testing is initial user satisfaction, to make sure that the interface changes and overall functionality changes have positive effects.

12.3 Testing Plan

With our main user class and goals defined, we were ready to create our testing plan. We decided the best way to accomplish our testing goals would be to conduct an analytic evaluation, more specifically a heuristic evaluation. We started creating this heuristic evaluation by listing the essential tasks our system should support. The essential tasks are listed below and were gathered from our user scenarios that were previously discussed in section 6.1.

A. From the home screen, have the user log into the system with ID: Durk, and with password: password.
B. From the home screen have the user create a graph named test and then get to the graph creation page.
C. From the graph creation page have the user choose a date range, graph type, data type, and then save the graph.
D. From the graph creation page have the user find a correlation in the data by using the graph creation feature.

To come up with our heuristics, we used Nielsen’s heuristics (https://www.nngroup.com/articles/ten-usability-heuristics/) as a base line and then identified ones most relevant to our goals. These heuristics did not address analyzing data, which is a very important aspect of our system, as well as we wanted. Consequently, we decided to add the “Ease of analyzation” heuristic. See Table 12.7.1.

Table 12.7.1 Heuristics (https://www.nngroup.com/articles/ten-usability-heuristics/)

<table>
<thead>
<tr>
<th>Heuristic</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visibility of system status</td>
<td>The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.</td>
</tr>
<tr>
<td>Match between system and the real world</td>
<td>The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.</td>
</tr>
<tr>
<td>User control and freedom</td>
<td>Users often choose system functions by mistake and will need a clearly marked &quot;emergency exit&quot; to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.</td>
</tr>
<tr>
<td>Ease of analyzation</td>
<td>The user should be able to easily analyze the graphs and chose which data types they are viewing.</td>
</tr>
<tr>
<td>Error prevention</td>
<td>Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.</td>
</tr>
<tr>
<td>Recognition rather than recall</td>
<td>Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.</td>
</tr>
</tbody>
</table>
Aesthetic and minimalist design | Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Efficiency of use | Users should be able to conduct tasks in a reasonable amount of time.

Help users recognize, diagnose, and recover from errors | Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Help and documentation | Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

With the testing plan complete, we began conducting the evaluation as a group. We felt this was a proper way to conduct this testing plan because multiple members of the group have had experience conducting heuristic evaluations in the past for a previous course, specifically CS 3724 Human-Computer Interaction. As a group we performed tasks A-D. While performing these tasks, we analyzed the system against our heuristics and recorded data regarding room for improvement. The data recorded included the issue found, together with relevant details including the task attempted, where they encountered the problem, and why it is a problem.

12.4 Testing Results

The results of the analytic evaluation are shown below. The tasks are referenced by their corresponding number in Section 12.3. This was only conducted on the new system.

A. From the home screen, have the user log into the system with ID: Durk, and with password: password (time how long this takes and count errors).
   a. Home page: Violates Visibility of system status/error prevention – for the home/login buttons when you hover over the box, the box gets highlighted but when you click the empty space in the block nothing happens. You are only taken to the desired link if you click on the word “Login”.
   b. Login page: Violates Help and Documentation - If the user forgets password there is no help or redirection page for them.

B. From the home screen have the user create a graph named test and then get to the graph creation page (time how long this takes and count errors).
   a. Home page: Violates recognition rather than recall – When desiring to create a graph, users might not know that in order to do that they have to go to experiments. There is nothing that indicates that except that they are pigeon holed between the “Home” and “Logout” buttons.
   b. Experiments page: Violates user control and freedom – The “Logout” button does not log the user out. This leaves the user no chance to go back or go to a different profile at this point.
   c. Experiments page: Violates visibility of system status – Under the “Create New Experiment” title there is a button on the side of a text entry box that says “Submit”. This does not inform the user that if they click that button they will go to the graph creation page.

C. From the graph creation page have the user choose a date range, graph type, data type, and then save the graph (time how long this takes and count errors).
   a. Graph Creation Page: Violates Aesthetic and minimalist design – This page is very minimal but not very aesthetically pleasing. Could better utilize color to help user understand different options of the page.
   b. Graph Creation Page: Violates Help users recognize, diagnose, and recover from errors – If a user does not enter all of the required information into the page they are not able to
create a graph if they click “Graph” or “Save Graph”. They will receive no notification of why the page is not performing how they expected.

c. Graph Creation Page: Violates Match between system and the real world – There are two titles for “Select a Type of Graph”. The one further down on the page is indicative of what it titles. But the one on the top of the page titles different types of data that could be graphed.

d. Graph Creation Page: Violates visibility of system status – When “Post!” button is clicked under the “Comments” title nothing happens. The text in the adjacent box remains and there is no notification of what happened by clicking that button.

D. From the graph creation page have the user find a correlation in the data by using the graph creation feature (time how long this takes and if the correlation is correct).

a. Graph Creation Page: Violates visibility of system status – There are no axis labels on the graphs. This makes the user unsure of what data is currently being shown on the graph.

b. Graph Creation Page: Aesthetic and minimalist design – When adjusting the screen size the navbar overlaps the page text.

12.5 Refinement

After conducting our testing plan, we were able to compile a list of changes that we could potentially implement to our website. These changes are not required to make the website more functional, but they are required to enhance usability. Most of these changes are simple user-interface fixes that we plan to address before the final deliverable.

The first change that could be made is to make the buttons clickable on any part of the button. Currently, when a user clicks a button, they have to click directly in the center of the button. If they click on the edge of a button, nothing happens.

The second change that could be made is to add a password recovery feature. The website now has no way to help a user if they forget their password. The user must create a new account and lose all of their saved information. By adding a password recovery feature, we save the users time and effort by allowing them to recover their lost accounts.

The third change that could be made is to add some way of telling a user that the Experiments page is where they create their graphs from the home page. By adding this simple information, we can enhance a user’s experience of the website. This will make using the website much easier and decrease the chance of errors by the user.

The fourth change that could be made is that when a user creates an experiment, they have to press the “Submit” button. However, this submit button does not convey any information to the user that they are about to be redirected. By adding a simple note that a user will be redirected to a new page, we will make the website much more clear to a user and hopefully increase user satisfaction with the website.

The fifth change that could be made is to make sure that our naming choices help guide the user and help them understand how to use the website and provide error messages. Currently, the website does not help the user in alerting what is happening when the user does a task on the website. Renaming buttons and adding pop ups of warnings will help the user not be lost or confused about what they are doing with the website. For example, in the graph page, we have the same title for two different functions. This causes confusion to the user and should be addressed. Also, we currently don’t have comments implemented; when the user tries to create a comment, nothing happens. Instead, we should have a pop up that alerts the user that it is still being worked
on. Also, when it is done, we would like a pop up that says that the comment has been posted.

Once we complete these changes we plan on obtaining IRB approval to conduct user based pilot testing. This will allow us to get feedback straight from the users about how well our system accomplishes our testing goals.

13. References


