Group 5
Centro Web App
Final Project Document

Marco Nicolas, Alexander DiDonato, Matthew Stevens, Vitaliy Shydlonok

Team Contract:

Team members are expected to show up or to be present in some way at meetings, to work on tasks that the team assigns, to put effort into discussions, and to communicate any issues that arise. Every team member must also keep track of the time they spend on any task so that the team could know if additional resources need to be assigned to that task. If a team member is falling behind on their work, the team will schedule a meeting to change the distribution of tasks and to discuss what might be causing the issue. Decisions should be made during meetings when everyone is present and able to discuss. If the team can't agree on a decision, a majority vote will be used to resolve the disagreement.

Mission Statement:

To produce an end product that is easy for others to use as well as works efficiently. As our product is improved; our team-skills, time management, and programming knowledge will progress with it.

Preferred Project: 1) Lunar Topology 2) IBM Sample Application

Software Requirements Specification

for

Centro Bus Performance Web-App
Version 1.0 approved
Prepared by: Marco Nicolas, Alexander DiDonato, Matthew Stevens, Vitaliy Shydlonok
SUNY at Oswego
9/13/2019

1. Introduction

1.1 Purpose

Centro bus web-app version 1.0. This SRS covers the entire project scope that our team will be working on. This includes both the client-side website app and the PHP web-server that the app communicates with.

1.2 Intended Audience and Reading Suggestions

This document is intended for developers and the client.

1.3 Product Scope

The web-app will be used for the purpose of tracking the performance of CENTRO busses. The objective is to have an app which can give the users a way to filter different buses and routes to show the average performance of those busses.

2. Overall Description

2.1 Product Perspective

This website is a standalone system designed to help the user obtain information regarding the on-time performance of Centro busses. It should be more efficient than the current Centro app because of the direct connection between the user and the MySQL database hosted on the department's servers. It should also be more user-friendly than the current alternative because it is designed with this limited scope in mind.

2.2 Product Functions

This website must be able to analyze the Centro bus routes and schedules in Oswego, NY and document how often the bus reaches specific stops on-time and the consistency of it. If given a specific timeframe, it should also be able to analyze and show how often the bus was either early/on-time/late to stops during that period.

2.3 User Classes and Characteristics

It's anticipated that SUNY Oswego students and staff as well as people in the Oswego community will be benefiting from this website. All users will also have the same level of access to all data on the website.

2.4 Operating Environment

- This will operate on PC and mobile devices
- It will operate on Chrome, Firefox, Internet Explorer

2.5 Design and Implementation Constraints

The program:

- should compile and display the requested information in less than 4 seconds.
- must only collect information from the user and the SQL database.
- must be able to run on a system as limited as a smartphone.
- must be formatted to fit both mobile and computer-based browsers.

3. External Interface Requirements

3.1 User Interfaces

The UI will be made up of two main panels: one interactive panel where the user can apply various filters to the data, and a second panel which will display the data in a table. The UI must be scalable with any sized browser window. This way, both panels would be readable even on small screens such as mobile devices.

3.2 Hardware Interfaces

The web-app will be a browser based application so it must work on any device that can run a web browser. Example supported devices: personal computers (Windows/Linux/OSX) and mobile devices (Android/iOS/Windows Phone). The web-server component will be programmed using PHP which can be run on any machine that supports the Apache HTTP web server.

3.3 Software Interfaces

The web-server will connect to an SQL database to retrieve data and send it on to the client. The client web-page will send a request to the web-server every time the user wants to refresh the data. The web-server will parse the request and send another request to the SQL server for the specific data that matches the user's filters. The SQL server will then send the data back to the web-server, which will build an html web-page containing the new data. The web-server will

then send the new web-page to the client, which will display it along with the refreshed data in a browser.

3.4 Communications Interfaces

The client/browser will communicate with the web-server over HTTP. The web-server will communicate with the SQL server using a TCP connection. The client must never have direct access to the SQL server, only the web-server can communicate with or have any information about the SQL server.

4. System Features

4.1 Data Filter

4.1.1 Description and Priority

The data filter will be a user interface that is populated by options to filter the CENTRO bus routes on display. This will be part of the client-side webpage. High priority because it is the main part of the web-app.

4.1.2 Stimulus/Response Sequences

The user selects a start and end date by using two calendar drop-downs or default to the current day. The user selects a bus route from a drop-down list of all available bus routes or selects “all routes”. The filter interface will expand to show another drop-down for selecting multiple routes. The user can then either add another route or click on the “refresh” button. Choosing to add another route will further expand the filter interface and this functionality will repeat until there are no more bus routes to add. If the user clicks the “refresh” button, the data filter will communicate with the web-server by requesting a new set of data. The request will contain the filters that the user applied to the data.

4.1.3 Functional Requirements

1. Drop down list of all available bus routes (including “all routes” option)
2. Expanding panel to add more than one route
3. “Refresh” button that sends a data request to the web-server

4.2 Web-Server Middleman

4.2.1 Description and Priority

The web-server is the middleman between the SQL server and the client web-app. Medium priority because this can be excluded at the cost of safety.

4.2.2 Stimulus/Response Sequences

The web-server will first receive an initial web-page request from a client. Once the client is connected, the web-server will send a web-page that only contains the filter interface and an empty table panel. When the client sends a refresh request, the web-server will connect to the SQL server and will request the data that matches the filter sent by the client. When the SQL server sends the data back to the web-server, the web-server will construct a web-page that contains the filter interface panel on the top and a table panel underneath that is populated with the new data. The web-page will then be passed along by the web-server to the client to display.

4.2.3 Functional Requirements

1. Build an initial web-page with the filter interface and empty data table
2. Receive client requests for filtered data
3. Construct SQL queries based on the filters sent by the client
4. Communicate with the SQL server (send queries and receive data)
5. Build a filtered web-page with the filter interface and populated data table
6. Send the filtered web-page to the client

4.3 Data Display

4.3.1 Description and Priority

The data display will show a table of values representing the on-time performance of busses that were selected through the filter interface. High priority because it is another main part of the web-app.

4.3.2 Stimulus/Response Sequences

The client will receive a web-page that was constructed by the web-server. This web-page will contain a table panel under the filter interface panel, populated by CENTRO bus data. The client will then display this web-page to the user, which is a process handled by the browser.

4.3.3 Functional Requirements

1. A data table that shows rows of selected bus stops and columns of selected bus routes
2. Each cell contains a numerical average based on bus performance data
3. Each cell is color coded (green for on time, red for late, yellow for early)

5. Other Nonfunctional Requirements

5.1 Security Requirements

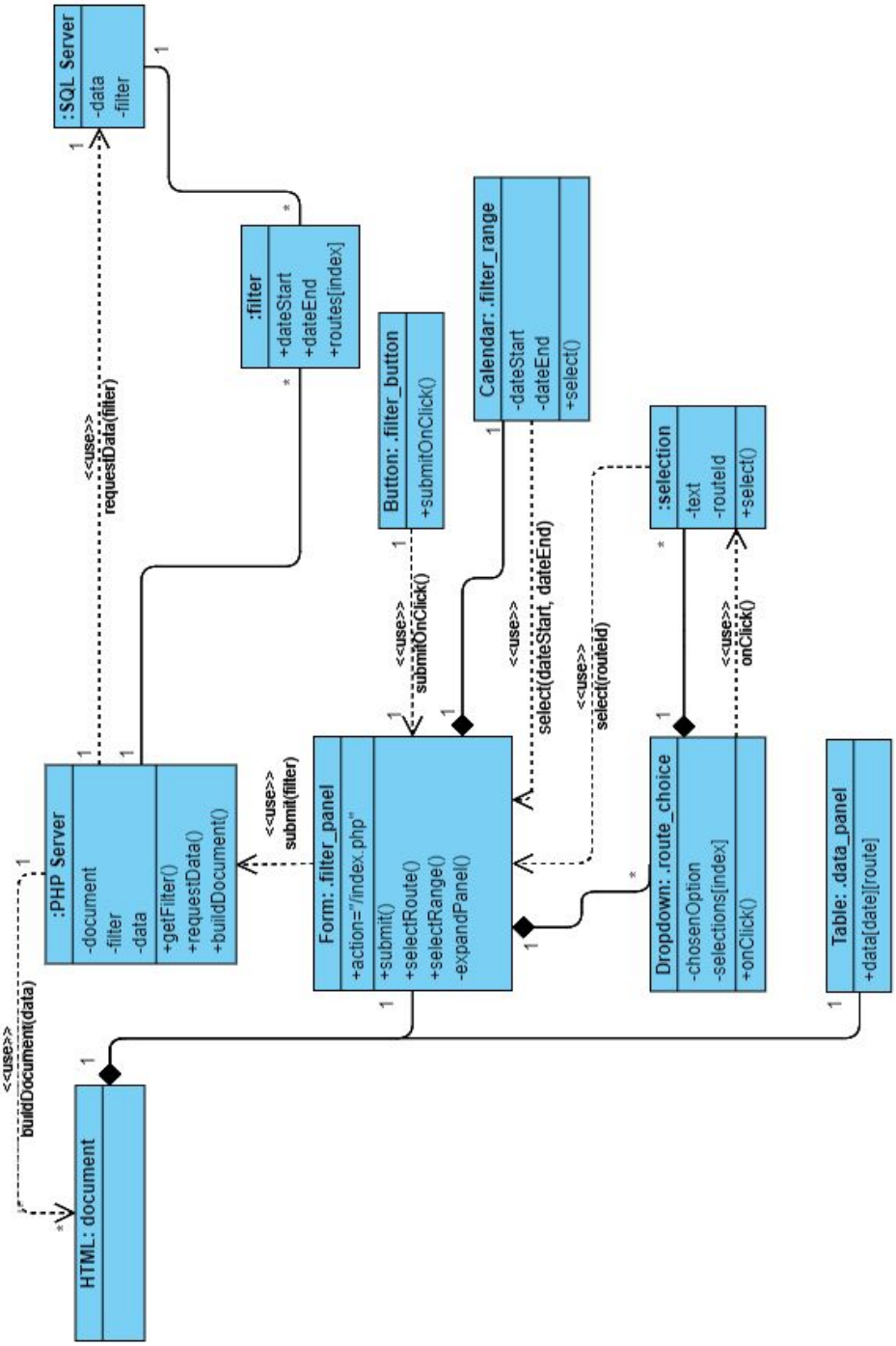
The user should not have access to the information on the SQL database directly, but rather through various filters. Administrators may access the data, however do not have permission to change it in any way. Once collected, the data can be manipulated by the program for the user.

5.2 Software Quality Attributes

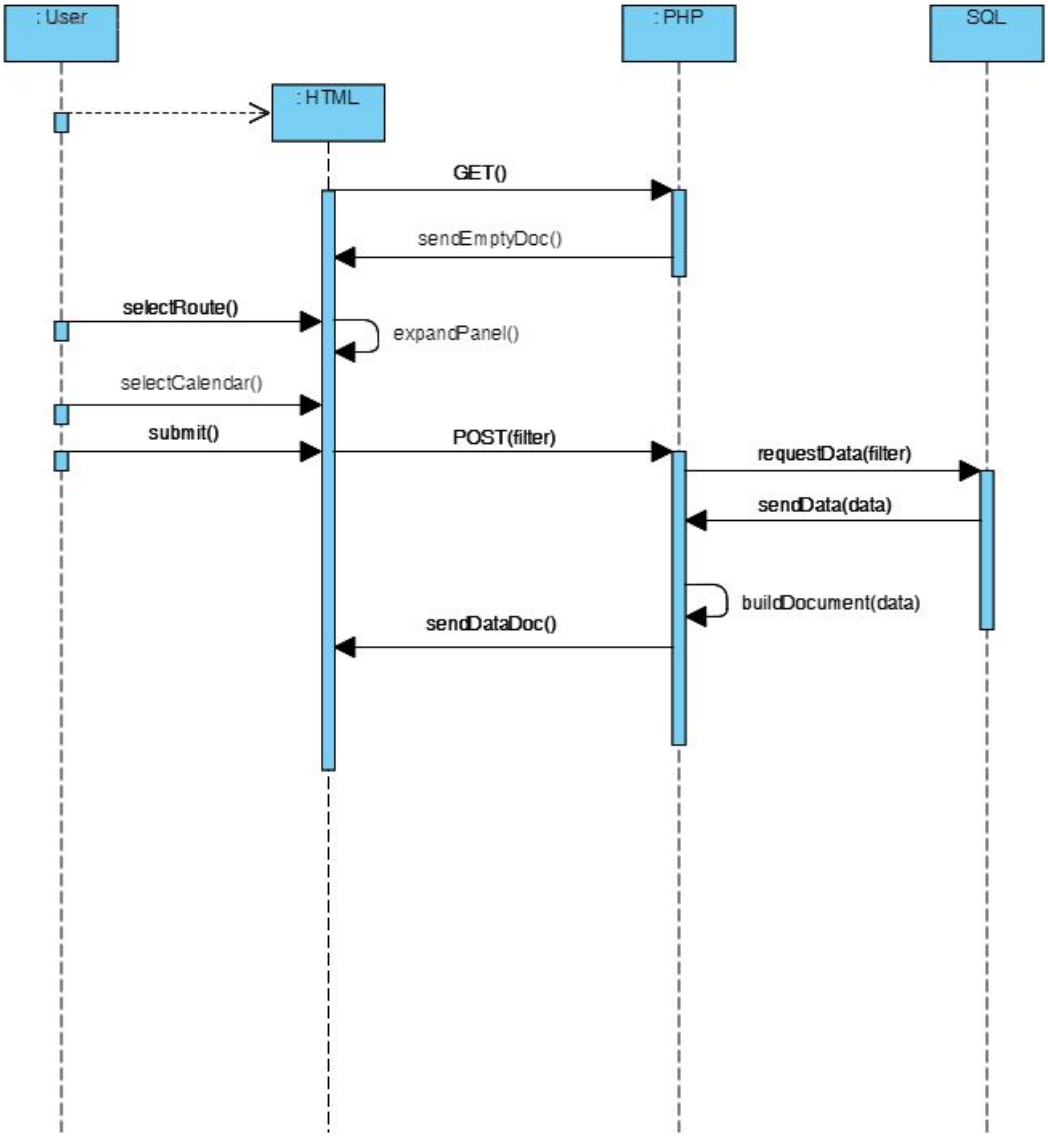
The main quality characteristics for the product are:

- Availability on a wide range of browsers and operating systems.
- Reliability of the data collected and it’s transformation to the user.
- Flexibility of the various filters available and their compatibility with each other.
- Usability across many devices with the performance of the average smartphone and above.
- Performance. The program should run quickly and get the required information to the user efficiently in repeated succession.

UML Diagram

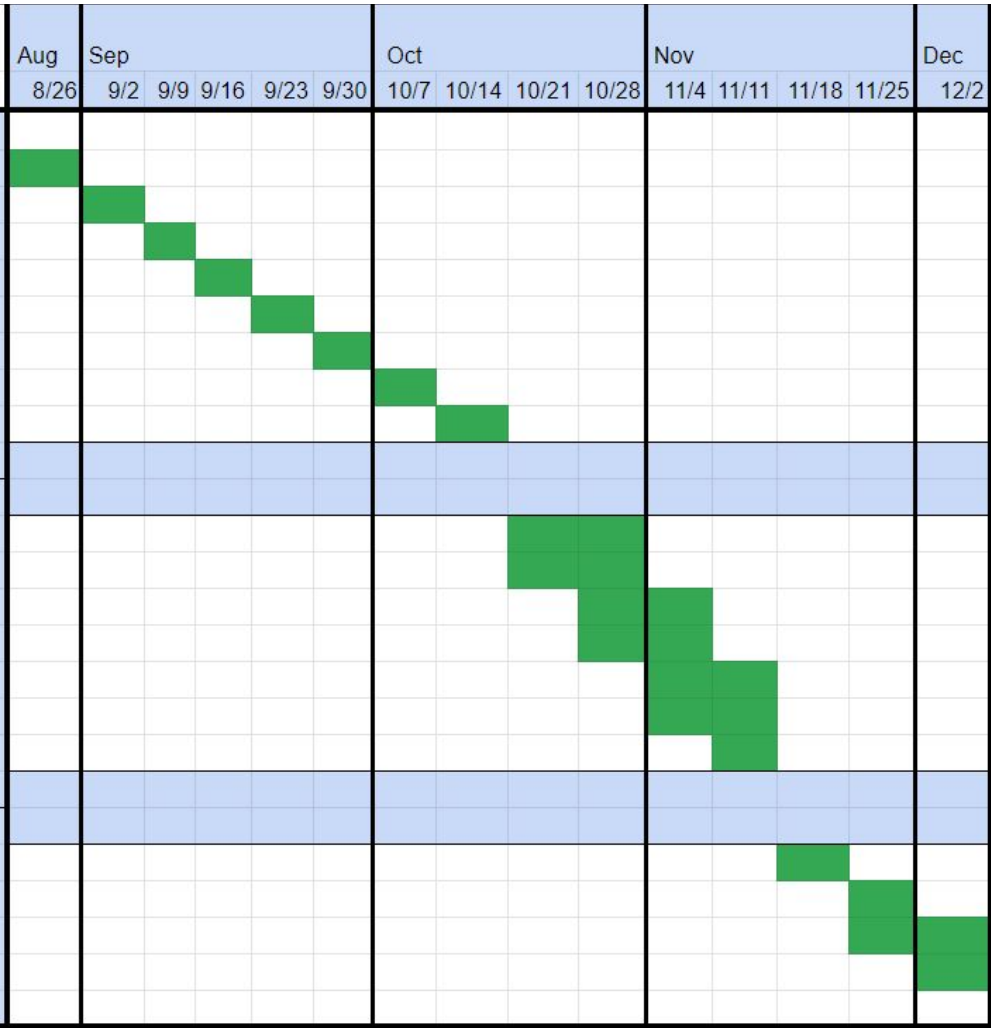

:SQL Server
-data
-filter
:filter
+dateStart
+dateEnd
+routes[index]
Button: .filter_button
+submitOnClick()
Calendar: .filter_range
-dateStart
-dateEnd
+select()
:selection
-text
-routeId
+select()
:PHP Server
-document
-filter
-data
+getFilter()
+requestData()
+buildDocument()
Form: .filter_panel
+action="/index.php"
+submit()
+selectRoute()
+selectRange()
-expandPanel()
Dropdown: .route_choice
-chosenOption
-selections[index]
+onClick()
Table: .data_panel
+data[date][route]
HTML: document
<<use>>
requestData(filter)
<<use>>
submit(filter)
<<use>>
submitOnClick()
<<use>>
select(dateStart, dateEnd)
<<use>>
select(routeId)
<<use>>
onClick()
<<use>>
buildDocument(data)

Sequence Diagram

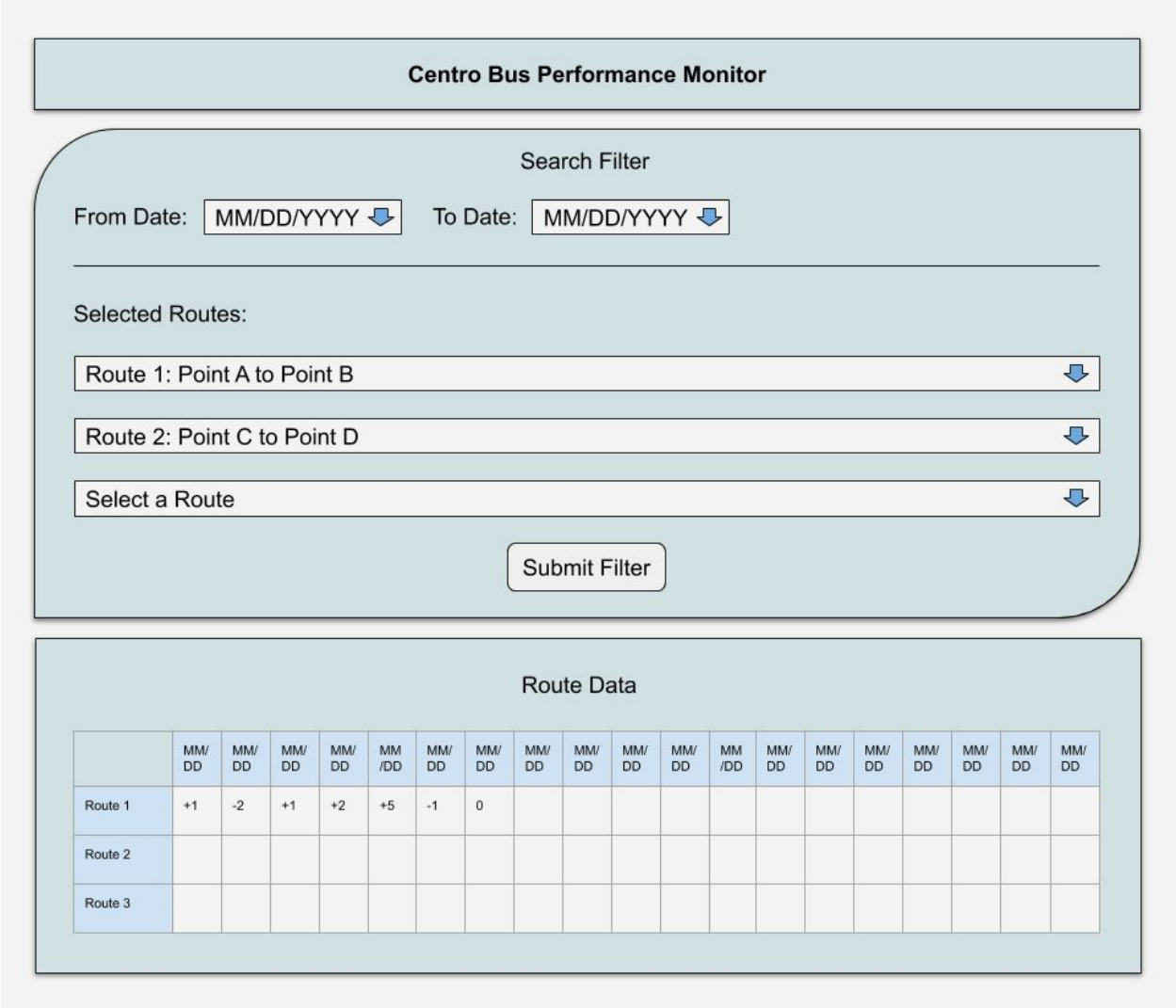


Centro Bus Web-Application:

Task	Duration	Leader
Design:		
Skill Survey	1 W	Team
Team Contract	1 W	Team
Requirements Document Draft	1 W	Team
Requirements Document Final	1 W	Team
UML Diagrams Draft	1 W	Team
UML Diagrams Final	1 W	Team
GANTT Chart	1 W	Team
GUI Design	1 W	Team
Implementation:		
Client HTML/CSS GUI	2 W	Vitaliy
Research communication with SQL	2 W	Marco
Client filter selection JS	2 W	Alex
Client communication with PHP Server	2 W	Matt
PHP Server parsing filter selection	2 W	Alex
PHP Server sending and receiving SQL	2 W	Matt
PHP Server building and sending new document	1 W	Vitaliy
Testing and QA:		
Unit Testing	1 W	Alex
Integration Testing	1 W	Marco
System Testing	2 W	Team
Acceptance Testing	1 W	Team



GUI Design


Centro Bus Performance Monitor
Search Filter
From Date:
MM/DD/YYYY
To Date:
MM/DD/YYYY
Selected Routes:
Route 1: Point A to Point B
Route 2: Point C to Point D
Select a Route
Submit Filter
Route Data
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
MM/DD
Route 1
+1
-2
+1
+2
+5
-1
0
Route 2
Route 3

Individual Reflections:

Vitaliy Shydlonok

I learned how to design a project before writing any code. Previously, I would write the code and then figure out how to organize it as I went along. That method was bad because I would have to constantly rework how different pieces interact. This time, we made sure to plan out the structure of the project using UML diagrams and sequence diagrams. We also created a requirements document that allowed us to know what things to focus on during development and what features we wanted to include. One thing I would have done differently is to back up the database regularly to prevent data loss. I would have also given us more time for the development and less time for unit testing because the testing didn't take very long but the development went over the allotted time.

Marco Nicolas

This course challenged me and forced me to try to learn things in certain areas that I was weak at, but it also taught me a lot about the software engineering world that I would've knew. I learned all the tedious things engineers have to take into consideration when building something for a client such as making a plan and sticking to it, making sure to tend to the customers needs, and trying to avoid mistakes to help keep things affordable. This course definitely changed my approach to software development because I had no idea there were so many factors to just developing software. I used to be a Software Engineer major before I switched to Information Science and I would've never thought it was this difficult and tedious for developers to create a working program that's actually user friendly and has little to no defects. If I was able to do something differently it would be to have more experience with coding specifically php. That was a huge weakness for me and it made it difficult to contribute to that part of the project, but I still tried to understand as much as possible to try to be of some help to my group members.

Alexander DiDonato

Prior to this course I had no experience with php and very little with html. However, with the help of my teammates I have a fairly good understanding of the syntax and how it works. The same can also be said with entering things into a database, both through a program and manually. The way I approach software assignments has also changed. I found that creating different charts and diagrams before I start programming is very useful and speeds up the whole process. It also makes testing much easier because the end result is visualized and can be compared to the actual output. If I was to do something differently it would be to create a backup for the database so the data is safe. Another change would be to get communication arranged at the beginning, which would have fixed all the dysfunction that we had early on.

Matthew Stevens

Over the duration of this course I've learned many great things about the software development process. The most important of those being planning and organization. It truly surprised me how much time we spent before any coding was done. I think the time spent learning the different types of UML's, Gantt charts, and requirements documents saved lots and lots of time in the long run. I had almost no knowledge of these planning and organization techniques before this class so I am happy I had the chance to learn them as they will no doubt be used in my future projects. Planning and organization aside, I also learned a lot about different languages and programs like datagrip, mySQL, php, html, and css. I spent a lot of time researching for this project, something I found interesting as I've never walked into a project knowing almost nothing about its functionality. Something I would have done differently from the beginning would have making backups of the data. A loss like the one we had was costly and set us back several days. I also would have coded more frequently as we became slightly crunched for time at the end. Everything worked out but it could have been less stressful.

Source Appendix:

common.css : CSS containing styling for the GUI
func.js : Javascript for the front-end functionality
index.php : Main source containing the php and HTML