**State University of New York at New Paltz**

**Emily Venuto**

**Project Type: Campus Project, “Class Key”: p-f17-15**

**“Library Usage Count Project”**

**FINAL PROJECT REPORT**

**Computer Science Projects**

**Fall 2017**

**(Prof. Hanh Pham)**

**TABLE OF CONTENTS**

**1. Problem Description**

**1.1 Business Context and Goals** ….……………………………………….…………….……. page 03

**1.2 Technical Requirements** ….……………………………………………….……….………. page 03

**1.3 Your Responsibilities** ……...……………………………………………….……….………. page 03

**2. Technologies**

**2.1 Related Technologies** ….……………………………………….……………………..…….. page 04

**2.2 Newly Learned Skills/Technologies …**………………………………..………………. page 04

**3. Design**

**3.1 System Architecture** ………………….……………………………………….………………. page 04

**4. Software/System Description** ………………………………………….……………..……..…………. page 05

**5. Test Results/Observations**

**5.1 Performance Observations** ……….…….…………………………….…..………………. page 06

**5.2 Quality Observations**……………………….……..……………………………….…………. page 06

**6. Professional and Career Benefits.**……………………………….……..….…….……..……………. page 06

**7. Conclusions** ………………………….……………………………………………...…………..………………. page 07

**8. References** ………………………….…………………..………………………………………..………………. page 07

**1. Problem Description**

**1.1 Business Context and Goals**

The Library Usage Count Project was launched to address the need for a specific web service for the Sojourner Truth Library. The library does not have a system that allows its staff to digitally log its visitors per hour; this work is currently being done manually. The goal of this project is to create a web service through which the library staff can track the date, time, and number of visitors per hour. This information will be stored in a MySQL database and will be usable in chart displays. The intended user for this software is the library staff. It will later be molded to fit into the current library web service. It requires a login to access.

The software is comprised of a login, a data log form, and a database. The login takes New Paltz library staff information, such as username and password. This gives the staff member access to the log form. The staff member is then presented with a web form that will allow him or her to enter visitor data, view the visitor data in a table, or view data in various charts. If the user chooses to log data, he or she will then press a submit button, which allows the input to be processed by a corresponding PHP file. This information is then saved in a predefined MySQL database. The PHP file then then returns the use to the main page to select another option.

**1.2 Technical Requirements**

The environment in which this service will be hosted by WordPress on a library staff PC. Therefore, it will be necessary that the user has internet access to run the service. A keyboard will be needed for user input. The user interface will be a simple web form. It is also required that this system be secure. The web form and database should not be accessible by anyone but the logged-in staff members.

**1.3 My Responsibilities**

My responsibilities in this project included creating a login system, designing a web form, initializing a MySQL database, and creating the PHP files that act as a go between for user input and MySQL database.

**2. Technologies**

**2.1 Related Technologies**

The required technologies for this project were HTML, CSS, and Bootstrap for the front end web pages, PHP for the back end server, JavaScript for the Google Charts, and MySQL for the databases.

**2.2 Newly Learned Skills/Technologies**

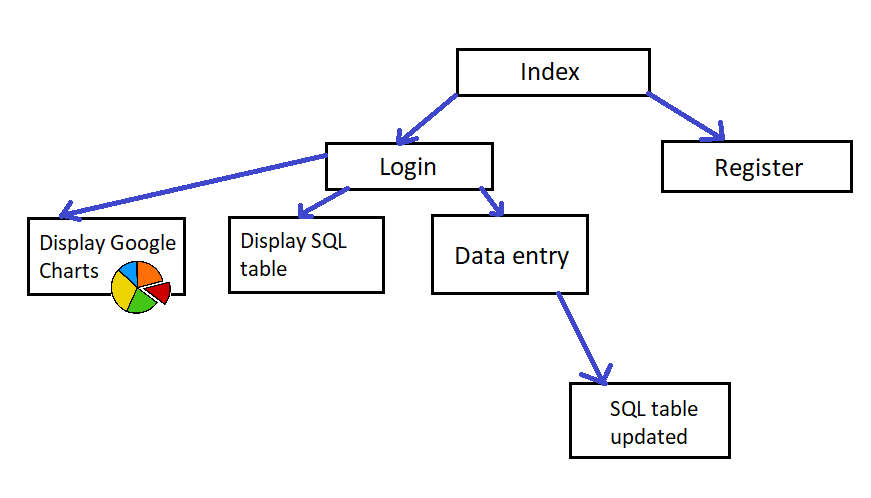
New technologies for me included PHP, JavaScript, and MySQL.

**3. Design**

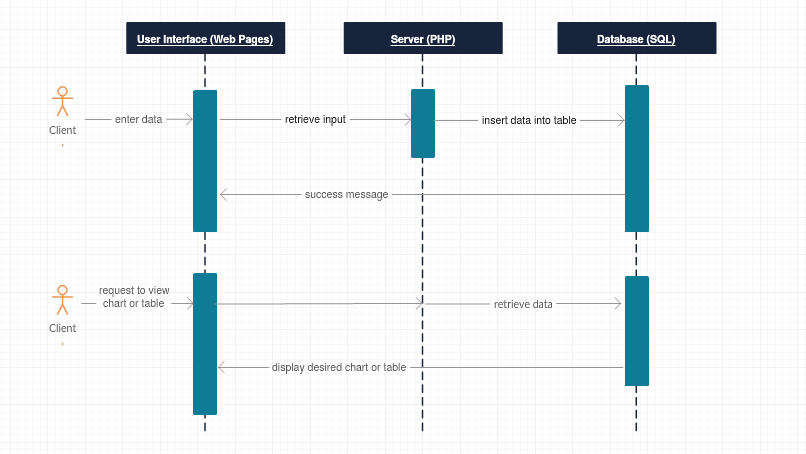
**3.1 System Architecture/Components**

The user starts by accessing the homepage, index.html. On this page, the user can either register a new staff member or login. When the user logs in, he or she can choose to enter data, display the current database table, or look at a chart representation of the current data. If the user chooses to add an entry, the input is saved to the Visit SQL table and saved. This update translates to the table and chart views mentioned previously. Everything is updated dynamically.

Architecture Flow:



System Protocol:

****

**4. Software/System Description**

There are several requirements I defined for this software system:

* Software has to be HTTPS compatible
* Coordinates HTML, PHP, and SQL
* Web application is built with HTML, CSS, Bootstrap, and JavaScript.

There are three important pieces of this project to understand: the SQL Visit table, the data entry system, and the chart display system.

1. SQL Visit table

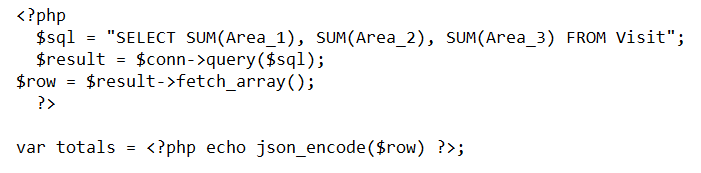
* The table that holds the visitor information
* Has four columns: Time (string 🡪 military format time), the three library floors (each entry holds the number of people on that floor at the given time)
* This table is instantiated ahead of time, the web application is used to update it. This is also where the chart display is drawn from

2. Data entry:

* Entry.htm takes input using a Bootstrap web form. The form posts to the PFP file insertData.php.
* insertData.php defines a variable for each input field and uses the $\_POST['data’] function to get the input. An SQL query is then used to insert the time of day and the floor counts into the table. An error is returned if this is unsuccessful.

3. Chart display:

* First, chartDisplay.php loads the Google AJAX API
* To create a Google chart, I loaded the appropriate package (ex: for a pie chart you’d use google.load('visualization', '1', {'packages':['piechart']});
* The Google chart I wanted to create was to show how many people had been on each floor cumulatively through the day. Therefore, I defined my createChart() function based on this idea.
* The createChart() function is used to bring in data and format it to be used in a chart. Within this function, I created a table to hold the information from the SQL table, adding the appropriate columns and rows. I used PHP to query the SQL database to collect the sums of each floor for that day. I used json\_encode() on each PHP array value to convert them to JSON. I then had to parse each JSON value to an integer so it had numerical meaning in the chart. This code snippet is displayed below:



* I instantiated a chart object, call the draw() function on it, passing the table that had been created as a parameter. From here, the chart will be displayed.

**5. Test Results/Observations**

**5.1 Performance Observations**

There are no issues with speed for this application. Everything runs quickly and smoothly.

**5.2 Quality Observations**

The main quality issue exists for displayChart.php. There is a mixed content error. Some of the JavaScript is blocked when the page is converted to HTTPS compatibility.

**6. Professional and Career Benefits**

Much of today’s computer science work is going to those who are knowledgeable in web service production. At the beginning of the semester, I had just begun to delve into Full Stack Web Development. This project aided me in learning new technologies that bolstered this interest. I spent many hours poring over PHP and MySQL, coordinating my web pages, and understanding how JavaScript played a part in Google Charts. I know that these will be important skills to have as I pursue my student and professional career.

**7. Conclusions**

Working project is located at <https://cs.newpaltz.edu/p/f17-03/>. Pieces of this project will be implemented for both the staff and for data visualization purposes on the new library website. This project provided a plat form for me to advance my web development skills and output a working, clean product.

**8. References**

[**https://www.w3schools.com/**](https://www.w3schools.com/)(for learning SQL and PHP)

[**https://www.tutorialspoint.com/php/php\_mysql\_login.htm**](https://www.tutorialspoint.com/php/php_mysql_login.htm)

[**http://getbootstrap.com/docs/4.0/getting-started/introduction/**](http://getbootstrap.com/docs/4.0/getting-started/introduction/)

[**https://developers.google.com/chart/interactive/docs/quick\_start**](https://developers.google.com/chart/interactive/docs/quick_start)(for learning Google Charts)