# Criterion A - Planning

## Description

Discovering and getting involved with school-based and community-based service opportunities for CAS and other service requirements is becoming a larger and larger challenge within our school. Many community members do not share possible service opportunities with the upper school due to the hassle of organizing and managing the service opportunities. Therefore, many experiences get ignored and students never get informed about them. Even when service opportunities do appear, there is not a proper system in place to view the service opportunities, so they end up being ignored. Solving these problems is the responsibility of the current Service Coordinator, Ms. xx. As someone who has personally struggled finding service opportunities in the past, I approached Ms. xx about these problems. She agreed that the current service system in the upper school was insufficient, and that solving the mentioned problems was necessary for future service opportunities to be recognized (see Appendix A). So, she agreed to be the client for this project.

Words: 160

## Rationale

The solution that Ms. xx and I agreed upon was a website to allow students and teachers to create, apply, and discuss possible service opportunities. The central part of the website is the student-teacher volunteer form, in which teachers would fill out a short form discussing details of their service opportunity. Then, students can access the website online to sign up for the service opportunities. I chose this solution to Ms. xx.’s problems because all upper school students are provided with a computer and email, meaning that this website can be accessed by every student as long as they have an internet connection (which is available on campus). Also, the website provides an easy method for students and organizers to connect through the forms, since connecting and discovering service opportunities was one of the main issues for students and teachers, which the website enables through the connected, always online space (Appendix A). The basic website will be made using HTML and CSS and the back-end will be managed by PHP scripts accessing an SQL database. I chose to use HTML and CSS for the web-design since they provide the necessary amount of fluidity in order to create a well-designed website without unnecessary elements that could make connecting the website to the back-end difficult.  I am not using a website building tool such as google sites or wix because they are extremely limiting when it comes to dynamic pages and back-end integration. I am using PHP to manage the data since PHP is great at parsing and managing SQL data with its in-built mysqli object and data processing functionality. PHP is also extremely easy to use with HTML and CSS as it can be embedded within the HTML code and work alongside it.

Words: 292

## Success Criteria

* Create website that can be accessed online
* Create teachers’ “create service opportunity” form that includes all necessary information
  + Date & Time
  + Frequency (If applicable) i.e. daily, weekly, monthly, etc.
  + Experience description
  + CAS strands (optional) - Used in Filters
  + Service Hours (optional) - Used in Filters
  + Learning Outcomes (optional) - Used in Filters
* Create student application form that includes all necessary information
  + Short Essay Write-up
    - “Why are you interested in this opportunity?”
    - “What are you looking to gain from this opportunity?”
* All service elements are adequately explained to users.
* Gather contact information from all users of the website (using integrated “sign-in with google” or form)
* Display available opportunities, removing opportunities that have been completed
* Different lists for Available opportunities, Active opportunities, Ongoing opportunities, and Completed opportunities.
* Website must be organized and easy to navigate
* All form data from the website is sent to an SQL database and is relatively secure.
* Teachers and students must be able to access applicant/organizer’s contact information.

# Criterion B – Solution Overview

## Record of Tasks:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task Number | Planned Action | Expected Outcome / Details | Time Estimated | Target Completion Date | Criterion |
| 1 | Find Client | Have a client with a clearly apparent software necessity | 1 hour | May 15th 2023 | A |
| 2 | Conduct Interview #1 with Client | Understand the problems and begin to develop a framework for the solution. Start brainstorming success  criteria. | 30 minutes | May 20th 2023 | A |
| 3 | Develop success criteria and write beginning of report. | Use the information gained in the first interview to develop 6-8 success criteria to propose to the client, and begin writing background information on the problem and solution  at hand in the report. | 1.5 hours | May 22nd 2023 | A |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 4 | Plan the software choice and its details. | Choose and understand software and tools to be utilized in the project as well as plan for any possible challenges to be encountered  programming-wise | 2 hours | May 22nd 2023 | B |
| 5 | Research how to integrate SQL databases with HTML and PHP forms & webpages | Learn how to connect HTML and SQL through PHP to understand web design limitations | 4 hours | January 5th 2024 | B |
| 6 | Draw & Visualize HTML/CSS format for each webpage | Create a design & layout for each webpage before programming the CSS and HTML for them in order to minimize errors. | 1 hour | January 29th 2024 | B |
| 7 | Create the phpMyAdmin Databases | After initializing an XAMPP Server, create the necessary databases in phpMyAdmin based off the webpage visualizations. | 2 hours | January 31st 2024 | B |
| 8 | Design main webpage workflow | Determine and Design how students will intuitively use & navigate the features of the service portal | 2 hours | February 2nd 2024 | B |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 9 | Program the basic HTML layout of web pages | Add the basic HTML elements for each webpage, including forms, tables, headers, etc. | 5 hours | February 10th 2024 | C |
| 10 | Program CSS Styling for webpages | Add CSS styling to each webpage, including element alignment, colors, and fonts. | 3 hours | February 10th 2024 | C |
| 11 | Program PHP connection between Sign-In and Log-In pages and Account Database | Use PHP to connect the sign-in and log-in pages to the Account Database created in phpMyAdmin. This includes account access, verification, and validation. | 5 hours | February 13th 2024 | C |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 12 | Program PHP connection between Service forms & Experience database | Use PHP to connect the teacher’s experience creation form and the student’s experience sign-up form to the SQL experiences database. | 5 hours | February 14th 2024 | C |
| 13 | Integrate Account & Experiences databases into UI of webpages | Use both the accounts and experiences databases to fill UI elements on webpages with relevant experiences, user information, etc. | 7 hours | February 17th 2024 | C |
| 14 | Design methods to test the program & its functionality | Design methods to test different features & success criteria that are important to the final product’s functionality | 2 hours | February 21st 2024 | B |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 15 | Run tests with client and verify basic functions | Show the client the finished product, and successfully complete tests to show it works. | 1 hour | February 26th 2024 | C |
| 16 | Create video evidence | Record and edit the video showing success criteria implementation. | 5 hours | March 3rd 2024 | D |
| 17 | Gather feedback from client | Get valuable feedback and possible future implementations from client after some time  using it | 1 hour | March 5th 2024 | E |
| 18 | Write Evaluation | Take any feedback and have a complete reflection | 1.5 hours | March 7th 2024 | E |

## Design Overview:

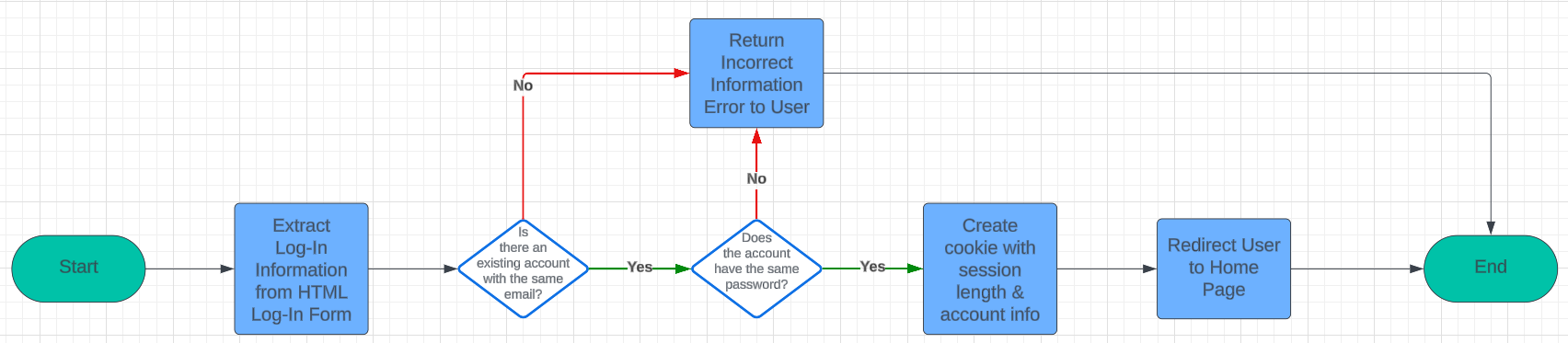
### Algorithm Flowcharts

**Account Sign-Up Process**

A diagram of a flowchart

Description automatically generated

**Account Log-In Process**

****

**Student Experience Sign-Up Process**

**A diagram with blue squares and black text

Description automatically generated**

**Student Experience Display Process**

A diagram of a diagram

Description automatically generated with medium confidence

## Test Plan

|  |  |
| --- | --- |
| Action to Test | Method of testing |
| Website can be accessed online | Have client access the site from their school computer while running the website from my computer |
| Access a form to create service opportunities with all necessary information (see success criteria) | Have client create a service activity on the website based off of one of their completed service activities to see if allotted inputs allow for essential information input |
| Sign up for a service activity and provide all necessary information | Have the client sign up for an existing service activity using the website and ensure that all information is provided by the client. |
| Navigate & understand the different webpages & functions easily | Have the client perform the other tasks with little guidance on how to access the necessary elements. |
| Gather the user’s contact information by creating an account for the website | Have the client create an account on their own when first asked to log onto the website. Ensure that the client cannot log-on without creating and validating their account details. |
| Display available opportunities, removing opportunities that have been completed. | The user’s navigation and accessing of the available sample opportunities and inability to access the sample completed experiences will demonstrate this success criteria |
| Different lists for Available opportunities, Active opportunities, Ongoing opportunities, and Completed opportunities. | Create multiple sample activities using the create activity form with start and end dates that vary from before the current date to after the current date, seeing if the database and website can correctly categorize each opportunity. |
| Send all form data from the website to an SQL database and keep it relatively secure | The connection between the client’s created activities, the sample activities, and the information displayed on the webpage demonstrates the connection to an SQL database. Security of the data comes from the use of third normal form when designing the tables. |
| Access participant contact information & responses | Have the client view their created activity after I have signed up for it using the website with all necessary information included. |

# Criterion C - Development

## Development of the Product

**Techniques Used to Develop the Solution**:

* **phpMyAdmin SQL Database**
  + Normalized Database Structures (3NF)
    - accounts
    - activity\_participants
    - service\_activities
  + Database Connection
  + Database Queries
    - SELECT
      * WHERE
      * LEFT JOIN
      * INNER JOIN
      * ORDER BY
    - INSERT
* **HTML**
  + <div>
  + <table>
  + <form>
  + <a>
* **CSS**
  + Classless CSS
  + Id-specific styling
  + Element styling
  + Page Centering
* **PHP**
  + Procedural Style MySQLi objects
  + Sessions
  + Regular Experession
  + Conditional Statements
    - If-Else-Elseif
    - If-Nested HTML
  + echo
  + Loop Handling - While Loops
* **Form Validation**
  + HTML Input Client-Side Validation
  + Custom PHP Server-Side Validation
  + Error Handling
  + Bcrypt Password Hashing

**Technique 1: phpMyAdmin SQL Database**

1. Normalized Database Structures

I created a database called “alcuin\_service\_db” that contains three tables in third normal form:

A screenshot of a computer

Description automatically generated

accounts:

A screenshot of a computer

Description automatically generated

activity\_participants:

A screenshot of a computer

Description automatically generated

service\_activities:

A screenshot of a computer

Description automatically generated

Normalization limits insertion, update, and deletion errors in the database and is important for the data’s integrity.

1. Database Connection

#Connect to SQL Database

    $mysqli = mysqli\_connect("localhost", "root", "", "alcuin\_service\_db");

This statement is used to establish a connection with the SQL database via the root login. This is done using a procedural style mySQLi object and its function mysqli\_connect();.

1. Database Queries

SELECT - Selects database data based on provided constraints. Used to display activity data on the site or to validate information.

WHERE – Filter data by specific value. Used to determine if data exists in database.

$sql = "SELECT \* FROM accounts

            WHERE email = '$email'";

LEFT JOIN – Used in main page to combine service\_activities and activity\_participants to find all service activities not signed up for by user. NULL results from a LEFT JOIN empty result lets us find the necessary activities.

$sql = "SELECT service\_activities.name, service\_activities.activity\_id FROM service\_activities

                            LEFT JOIN activity\_participants

                            ON service\_activities.activity\_id = activity\_participants.activity\_id

                            AND activity\_participants.account\_id = {$acc\_data["account\_id"]}

                            WHERE account\_id IS NULL AND organizer\_id != {$acc\_data["account\_id"]} AND start\_date < CURRENT\_TIMESTAMP AND end\_date >= CURRENT\_TIMESTAMP

                            ORDER BY start\_date DESC";

INNER JOIN – Used in main page to only display activities that user has signed up for. All activities without sign ups are not displayed in result due to INNER JOIN being used instead of LEFT JOIN.

$sql = "SELECT service\_activities.activity\_id, service\_activities.name FROM service\_activities

                            INNER JOIN activity\_participants

                            ON service\_activities.activity\_id = activity\_participants.activity\_id

                            WHERE account\_id = {$acc\_data["account\_id"]} AND organizer\_id != {$acc\_data["account\_id"]} AND start\_date >= CURRENT\_TIMESTAMP";

ORDER BY – Used to order activity entries by date on the main page. Used for an easier organization method for users to understand and to help sort the activities into Future, Ongoing and Completed Experiences based on their start times (descending order).

$sql = "SELECT service\_activities.name, service\_activities.activity\_id FROM service\_activities

                            LEFT JOIN activity\_participants

                            ON service\_activities.activity\_id = activity\_participants.activity\_id

                            AND activity\_participants.account\_id = {$acc\_data["account\_id"]}

                            WHERE account\_id IS NULL AND organizer\_id != {$acc\_data["account\_id"]} AND start\_date >= CURRENT\_TIMESTAMP

                            ORDER BY start\_date DESC";

INSERT – Adds a new row of data to the table. Used during sign up and activity creation to add new accounts/activities.

$sql = "INSERT INTO accounts (email, first\_name, last\_name, password\_hash) VALUES ('$email', '$fname', '$lname', '$password\_hash')";

**Technique 2: HTML**

The HTML code defines the base elements that create the foundation of the website. These are a few of the complex and unique elements used within the program:

1. <div>

The div element was used to group and visually organize elements together. The manner in which they were organized was determined by the element’s id (discussed further in Element styling)

<div id="login-box">

    …

</div>

1. <table>

Used to list elements retrieved from the SQL database (activities, participants, etc.). Typically used within echo statements.

<table style="width:100%;">

…

echo "<tr><td>$activity[name]<a href=\"service.php?id=$activity[activity\_id]\"><button style=\"float: right;\" onclick=\"\">View</button></a></td></tr>";

if($is\_empty) {echo "<tr><td>Nothing here :(</td></tr>";}

…

</table>

1. <form>

Stores label, input, and submit elements for forms. Has the post method attribute to store form data securely upon submission for use with the PHP code and SQL database.

if ($\_SERVER["REQUEST\_METHOD"] === "POST")

{

    …

}

…

<form method="post">

                <label for="email">Email:</label>

                <br>

                <input type="email" id="email" name="email" size="35px" style="margin-bottom: 15px;">

                <br>

                <label for="password">Password:</label>

                <br>

                <input type="password" id="password" name="password" size="35px" style="margin-bottom: 20px;">

                <br>

                <input type="submit" label="Submit" style="margin: 5px;">

            </form>

1. <a>

Used to create interactable links for webpages on the website. Often fixed in HTML code, but occasionally used in echo statements for dynamic links.

<a href="main.php"><img src="./img/home-button.svg" height="40px"></a>

**Technique 3: CSS**

CSS code was used to stylize the website and organize elements on the webpage.

1. Classless CSS

I used a pre-made classless CSS style from github to create the base visual style for my website. This saved a lot of time on stylization of HTML elements that was spent creating the rest of the product.

/\* GitHub Link: https://github.com/raj457036/attriCSS \*/

/\*# sourceMappingURL=brightlight-green.css.map \*/

1. Id-Specific Styling

Commonly used with the <div> element to apply more specific styling to all elements. Uses specificity rules to apply styling to appropriate elements.

#experiences {

    width:20%;

    float:left;

    margin:40px;

    padding:50px;

    padding-top:20px;

    border: 1px solid darkgray;

}

1. Element Styling

Most specific styling format used. Typically used for special padding, margin, or text color of a single element.

<div id="login-box">

            <h4 style="margin:0px; margin-bottom: 35px;">Log In</h4>

1. Page Centering

Special method used to center elements both horizontally and vertically. Useful in emphasizing groups of elements on a page no matter the size or shape.

login-box {

    border: 1px solid darkgray;

    text-align: center;

    display: inline-block;

    padding: 70px;

    padding-top: 35px;

    padding-bottom: 10px;

    position: fixed;

    top: 50%;

    left: 50%;

    -webkit-transform: translate(-50%, -50%);

    transform: translate(-50%, -50%);

}

#activity-box {

    text-align: center;

    display: inline-block;

    position: fixed;

    left:50%;

    -webkit-transform: translate(-50%, 0%);

    transform: translate(-50%, 0%);

}

**Technique 4: PHP**

1. Procedural Style MySQLi Objects

The in-built MySQLi object was used to connect between the PHP code and the SQL database, interacting with the data. I chose the procedural style for this object, making every subclass and function a separate function of the object.

    #Connect to SQL Database

    $mysqli = mysqli\_connect("localhost", "root", "", "alcuin\_service\_db");

$sql = "SELECT \* FROM accounts

            WHERE email = '$email'";

    $result = mysqli\_query($mysqli, $sql);

    $acc\_data = mysqli\_fetch\_assoc($result);

1. Sessions

Unlike cookies, sessions can only be accessed on the server-side, making them much more secure. Therefore, I used sessions to store important user data like their ID, letting the user stay logged in as they navigate the site and even as they leave the site.

if($acc\_data && password\_verify($password, $acc\_data["password\_hash"]))

    {

        session\_start();

        session\_regenerate\_id();

        $\_SESSION["user\_id"]=$acc\_data["account\_id"];

        mysqli\_close($mysqli);

        header("Location: main.php");

        exit;

1. Regular Expressions

I used regular expressions to validate the password information entered on the signup page through the in-built preg\_match(); function.

elseif(!preg\_match("/[a-z]/i", $password))

    {

        $signuperr = "Password must contain at least one letter";

    }

    elseif(!preg\_match("/[0-9]/", $password))

    {

        $signuperr = "Password must contain at least one number";

    }

1. Conditional Statements

Conditional statements were used frequently to conditionally execute PHP code or display HTML elements.

If-Elseif-Else Statements – These were mostly used to perform multiple validation tests before executing some PHP code. These were used instead of a switch-case statement because the validation tests often involved multiple variables and different checks such as empty();, strlength();, and preg\_match();.

if (!filter\_var($email,FILTER\_VALIDATE\_EMAIL))

    {

        $signuperr = "Valid Alcuin email is required";

    }

    elseif (empty($fname))

    {

        $signuperr = "First Name is required";

    }

    …

    else

    {

…

}

If-Nested HTML – When HTML code needed to be conditionally shown to the user, the elements were nested between separate PHP if and endif statements.

<?php if($created): ?>

            <em style="color:#11FF11; margin:auto; margin-left: 25%; width:50%; text-align:center; display:inline-block;">Experience Created!</em>

        <?php endif; ?>

        <?php if($signup): ?>

            <em style="color:#11FF11; margin:auto; margin-left: 25%; width:50%; text-align:center; display:inline-block;">Sign Up Successful!</em>

        <?php endif; ?>

1. echo

Echo statements were used frequently to add HTML elements to the webpages through PHP code. This statement allowed the formatting and addition of various elements to be looped through, added conditionally, or modified based on some parameters.

while($activity = mysqli\_fetch\_assoc($result))

                    {

                        echo "<tr><td>$activity[name]<a href=\"service.php?id=$activity[activity\_id]\"><button style=\"float: right;\" onclick=\"\">View</button></a></td></tr>";

                        $is\_empty = false;

                    }

They were also used to define the content of HTML elements based on PHP variables.

<input type="submit" value="Sign Up" style="margin:5px;" <?php if($disabled){echo "disabled";}?>>

1. Loop Handling – While Loops

Loops were mainly used to run through query results from the database. Since there is no definitive length to the number of results from a query, while loops were exclusively used.

while($activity\_data = mysqli\_fetch\_assoc($result))

        {

            …

                while($participant\_id = mysqli\_fetch\_assoc($result))

                {

                    …

                }

                break;

            …

        }

**Technique 5: Form Validation**

1. Client-Side HTML Validation

Although I did code a Server-side validation through PHP, I kept some of the built-in HTML validation as a redundant backup to minimize the error handling that the PHP code needs to perform.

<input type="email" id="email-validation" name="email" size="35px" style="margin-bottom: 15px;">

1. Server-Side PHP Validation

The server-side PHP validation is used to cover most basic user errors and security issues with the form data. This includes requiring every input to be filled out and ensuring the password is adequately secure. This prevents SQL querying errors later due to missing or invalid form data.

#Server-Side Info Validation

    if (!filter\_var($email,FILTER\_VALIDATE\_EMAIL))

    {

        $signuperr = "Valid Alcuin email is required";

    }

    elseif (empty($fname))

    {

        $signuperr = "First Name is required";

    }

    elseif (empty($lname))

    {

        $signuperr = "Last Name is required";

    }

    elseif (strlen($password) < 8)

    {

        $signuperr = "Password must be at least 8 characters long";

    }

    elseif(!preg\_match("/[a-z]/i", $password))

    {

        $signuperr = "Password must contain at least one letter";

    }

    elseif(!preg\_match("/[0-9]/", $password))

    {

        $signuperr = "Password must contain at least one number";

    }

    elseif($password !== $password\_valid)

    {

        $signuperr = "Passwords must match";

    }

1. Error Handling

After identifying the errors through Client-Side and Server-Side Validation, the errors are displayed to the user through the webpages using a conditionally displayed <em> tag.

<em style="color:#FF1111;"> <?php echo $signuperr; ?> </em>

<?php if($is\_invalid): ?>

                <em style="color:#FF1111;">Invalid Login</em>

            <?php endif; ?>

1. Bcrypt Password Hashing

The user’s password is stored securely in the database by using Bcrypt Hashing. Since the process is irreversible, the password cannot be stolen through the hash.

#Hashing Password

        $password\_hash = password\_hash($password, PASSWORD\_DEFAULT);

As such, in order to verify inputted passwords during login, they must hashed before comparing the hashes against each other using the password\_verify(); function.

if($acc\_data && password\_verify($password, $acc\_data["password\_hash"]))

Words: 984

Programming Resources:

Classless CSS: <https://github.com/raj457036/attriCSS>

XAMPP: <https://www.apachefriends.org>

PHP Documentation: <https://www.php.net/docs.php>

# Criterion E - Evaluation

## Evaluation of the Product

The product I created for my client Ms. xx was evaluated against the success criteria and her expectations. Overall, the product met almost all of the success criteria, with the only slight problem in my product with respect to my success criteria being that I didn’t explicitly include fields for the optional CAS Strands, Learning Outcomes, and Frequency inputs in the create activity form. This issue was also highlighted during my Evaluation interview seen in Appendix C, as even Ms. xx was looking for those fields, but ended up having to put the information in the description. The other features, including the online functionality (amongst the network), aesthetics, and security were all well implemented. However, above all these features, the client stressed the importance of the website’s ease of use, since her initial problem was with people being unable to intuitively connect and share service experiences. Without an intuitive and easy-to-use system, the program itself would become obsolete. As seen in our interview, Ms. xx found that the product met her expectations in this area, with the clear descriptions and UI helping to guide users through the process. She did agree that some areas needed to be clearer, like what information students and teachers were supposed to provide in their descriptions, but the overall product was very intuitive and useful.

Words: 220

## Recommendations for improving the product

During our interview transcribed in Appendix C, my client Ms. xx and I came up with a few ways in which the product could be improved. First, I have yet to implement a way to edit or update account information, so if someone’s email or name changes, they are unable to change the account’s information to match, which is important to add in the future as the user’s contact information is central to the website’s functionality. Similarly, update and deletion features for experiences should also be added, as canceled experiences or incorrect information being passed off as active ones in the system could cause confusion for organizers and applicants. Also, in order to resolve the success criteria flaws, the optional information listed in the success criteria for creating an activity such as frequency, CAS strands, and learning outcomes should not in separate fields. Rather, they should be added to the description like I did for my sample experience if they are to be specified. This ensures that there won’t be any empty cells within the SQL database, and that there will not be any confusion if organizers decide against adding them. However, to indicate that the description is the right place for those features, the description will be pre-filled with some general headers, which will include the optional information, indicating their suggested location within the form. On a service level, I want to expand the website across multiple networks rather than limiting it to just the school network. This ensures that applicants and organizers can continue to communicate and use the website regardless of location or network, even across long holidays such as summer or winter break. This also makes implementing the email notification system easier, as I will be able to implement the API system to work at all times with every account. Hopefully, these improvements will make the system even more useful to students across our school.

Words: 319

# **Appendix A**

**Ms. xx Initial Interview Transcript:**

Me: Alright, I’ve already explained to you my project idea over text, but just to restate it, I am interested in making a student service volunteer form where teachers can put their service opportunities and the details of the activities and then students can apply for the opportunities. Since you are the service coordinator for StuCo, I thought that you would be the most suitable client for this project. What are your thoughts?

Ms. xx: Yeah, I mean I think this would be really helpful because a lot of students are looking for service opportunities for the presidential awards or for CAS. I think giving them access to those opportunities would be really nice, both for inside and outside the school

Me: So to clarify, would you prefer that the opportunities be inside and outside the school community?

Ms. xx: Yeah, I don't want to limit this project to just the school. I am not expecting many opportunities from outside though.

Me: Okay, so I just want to get a grasp of what are your expectations for this project, like are there any important features or aspects of the product that you see as necessary for the functionality of it?

Ms. xx: I think that the most important feature would probably be being able to easily access the site or be notified of new opportunities. As it is, very few students frequently check the StuCo website even though the Tech Coordinator consistently updates it, so instead of putting it there maybe you could send out an email or notification through an app. I don’t know how you are going to make this program so as long as there is a way to notify students thats fine.

Me: From the suggestions you gave I think email might be the easiest and most effective method, because I could probably give the student and teacher emails in the form/site for ease of access.

Ms. xx: Yeah that’s perfect, it just needs to be used by people because otherwise there is no reason to have it. Also in the past teachers with service opportunities have been emailing all of upper school about each and every service opportunity which can clog our inboxes and get e

Me: Ok, now in terms of the form itself, what information do you think teachers should be required to add? I am not sure what information is necessary for service or CAS experiences.

Ms. xx: Well other than the basic scheduling stuff you told me about by text, I think all the teacher should need to input is the details of the experience, so a description of the experience, what type of service it is, and the number of participants as well.

Me: What do you mean by the type of service?

Ms. xx: I mean like helping kids learn, or cleaning up a park, or helping a teacher prepare their class in the morning.

Me: So different filters almost that we can categorize opportunities by?

Ms. xx: Yeah something like that.

Me: Okay, one last thing. I was thinking about ways to integrate this form as a way to generate CAS experiences, like teachers would pick learning objectives and the strands and fill out the information about the experience for the student.

Ms. xx: That would be nice, but I don't know how much IB and CAS the Elementary teachers know, since they are a part of the Montessori system. If you could though that would be a nice addition.

Me: Ok, now for students. Students will have to respond to the teachers’ service requests. What should they put in their, I guess applications of sorts?

Ms. xx: I don’t want students to have to do much, because that will discourage them from participating. Maybe just a short response that asks like “Why are you interested in this?” or, “What do you want to gain from this experience?”. Just something to make them think about the process more and provide information to the teachers.

Me: Ok. Well I believe that is all I needed from you today, I will keep in touch with you about my progress on the project! Thank you.

Ms. xx: You too! I am looking forward to it.

# **Appendix B**

**Website Code:**

login.php:

<?php

$is\_invalid = false;

if(isset($\_GET['signup']))

{

    $signup = $\_GET['signup'];

}

if ($\_SERVER["REQUEST\_METHOD"] === "POST")

{

    #Storing form data

    $email = $\_POST['email'];

    $password = $\_POST['password'];

    #Connect to SQL Database

    $mysqli = mysqli\_connect("localhost", "root", "", "alcuin\_service\_db");

    $sql = "SELECT \* FROM accounts

            WHERE email = '$email'";

    $result = mysqli\_query($mysqli, $sql);

    $acc\_data = mysqli\_fetch\_assoc($result);

    if($acc\_data && password\_verify($password, $acc\_data["password\_hash"]))

    {

        session\_start();

        session\_regenerate\_id();

        $\_SESSION["user\_id"]=$acc\_data["account\_id"];

        mysqli\_close($mysqli);

        header("Location: main.php");

        exit;

    }

    $is\_invalid = true;

}

?>

<!DOCTYPE html>

<html lang="en">

    <head>

        <meta charset="UTF-8">

        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <link rel="stylesheet" type="text/css" href="format.css" />

        <title>AS Student Service Portal</title>

    </head>

    <header>

        <h1 id="title" style="text-align:center;">Log-In to Alcuin Student Service Portal</h1>

    </header>

    <body>

        <div id="login-box">

            <h4 style="margin:0px; margin-bottom: 35px;">Log In</h4>

            <?php if($is\_invalid): ?>

                <em style="color:#FF1111;">Invalid Login</em>

            <?php endif; ?>

            <?php if(isset($signup) && !$is\_invalid): ?>

                <em style="color:#11FF11;">Sign-Up Successful!</em>

            <?php endif; ?>

            <form method="post">

                <label for="email">Email:</label>

                <br>

                <input type="email" id="email" name="email" size="35px" style="margin-bottom: 15px;">

                <br>

                <label for="password">Password:</label>

                <br>

                <input type="password" id="password" name="password" size="35px" style="margin-bottom: 20px;">

                <br>

                <input type="submit" label="Submit" style="margin: 5px;">

            </form>

            <a href="signup.php" style="font-size: 10; margin-top: 60px;">Dont have an account?</a>

        </div>

    </body>

</html>

signup.php:

<?php

$signuperr = "";

if ($\_SERVER["REQUEST\_METHOD"] === "POST")

{

    #Storing form data

    $email = $\_POST['email'];

    $fname = $\_POST['fname'];

    $lname = $\_POST['lname'];

    $password = $\_POST['password'];

    $password\_valid = $\_POST['password-validation'];

    #Server-Side Info Validation

    if (!filter\_var($email,FILTER\_VALIDATE\_EMAIL))

    {

        $signuperr = "Valid Alcuin email is required";

    }

    elseif (empty($fname))

    {

        $signuperr = "First Name is required";

    }

    elseif (empty($lname))

    {

        $signuperr = "Last Name is required";

    }

    elseif (strlen($password) < 8)

    {

        $signuperr = "Password must be at least 8 characters long";

    }

    elseif(!preg\_match("/[a-z]/i", $password))

    {

        $signuperr = "Password must contain at least one letter";

    }

    elseif(!preg\_match("/[0-9]/", $password))

    {

        $signuperr = "Password must contain at least one number";

    }

    elseif($password !== $password\_valid)

    {

        $signuperr = "Passwords must match";

    }

    else

    {

        #Hashing Password

        $password\_hash = password\_hash($password, PASSWORD\_DEFAULT);

        #Connect to SQL Database

        $mysqli = mysqli\_connect("localhost", "root", "", "alcuin\_service\_db");

        #Check Database for Account

        $sqlvalid = "SELECT \* FROM accounts

                WHERE email = '$email'";

        $result = mysqli\_query($mysqli, $sqlvalid);

        $acc\_data = mysqli\_fetch\_assoc($result);

        if($acc\_data)

        {

            $signuperr = "Account already exists. <a href=\"login.php\" style=\"font-size: 10; margin-top: 60px;\">Try Logging In</a>";

            mysqli\_close($mysqli);

        }

        else

        {

            #Add Account to Database

            $sql = "INSERT INTO accounts (email, first\_name, last\_name, password\_hash) VALUES ('$email', '$fname', '$lname', '$password\_hash')";

            mysqli\_query($mysqli, $sql);

            header("Location: login.php?signup=true");

            mysqli\_close($mysqli);

            exit;

        }

    }

}

?>

<!DOCTYPE html>

<html>

    <head>

        <link rel="stylesheet" type="text/css" href="format.css" />

        <title>AS Service Sign-Up</title>

    </head>

    <header>

        <h1 id="title" style="text-align:center;">Sign-Up for the Alcuin Student Service Portal</h1>

    </header>

    <body>

        <div id="login-box">

            <h4 style="margin:0px; margin-bottom: 35px;">Sign Up</h4>

            <em style="color:#FF1111;"> <?php echo $signuperr; ?> </em>

            <form method="post">

                <label for="email-validation">Email:</label>

                <br>

                <input type="email" id="email-validation" name="email" size="35px" style="margin-bottom: 15px;">

                <br>

                <label for="first-name">First Name:</label>

                <br>

                <input type="text" id="first-name" name="fname" size="35px" style="margin-bottom: 15px;">

                <br>

                <label for="last-name">Last Name:</label>

                <br>

                <input type="text" id="last-name" name="lname" size="35px" style="margin-bottom: 15px;">

                <br>

                <label for="password">Password:</label>

                <br>

                <input type="password" id="password" name="password" size="35px" style="margin-bottom: 20px;">

                <br>

                <label for="password-validation">Confirm Password:</label>

                <br>

                <input type="password" id="password-validation" name="password-validation" size="35px" style="margin-bottom: 20px;">

                <br>

                <input type="submit" label="Sign-Up" style="margin: 5px;">

            </form>

            <a href="login.php" style="font-size: 10; margin-top: 60px;">Already have an account?</a>

        </div>

    </body>

</html>

main.php:

<?php

    #Session Check

    session\_start();

    if(isset($\_SESSION["user\_id"]))

    {

        $mysqli = mysqli\_connect("localhost", "root", "", "alcuin\_service\_db");

        $sql = "SELECT \* FROM accounts

                WHERE account\_id = {$\_SESSION["user\_id"]}";

        $result = mysqli\_query($mysqli, $sql);

        $acc\_data = mysqli\_fetch\_assoc($result);

    }

    else

    {

        header("Location: login.php");

    }

    #Logout function

    if(isset($\_GET["logout"]))

    {

        session\_destroy();

        header("Location: login.php");

    }

    #Special messages

    $created = false;

    if(isset($\_GET['created']))

    {

        $created = $\_GET['created'];

    }

    $signup = false;

    if(isset($\_GET['signup']))

    {

        $signup = $\_GET['signup'];

    }

?>

<!DOCTYPE html>

<html lang="en">

    <head>

        <meta charset="UTF-8">

        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <link rel="stylesheet" type="text/css" href="format.css" />

        <title>AS Student Service Portal</title>

    </head>

    <header>

      <a href="main.php"><img src="./img/home-button.svg" height="40px"></a>

        <h1 id="title" style="margin:auto; margin-left: 23%; width:50%; text-align:center; display:inline-block;">Alcuin Student Service Portal</h1>

        <div id="acc\_info" style="float:right; margin-right: 5px;">

        <p style="margin-top: 0px; margin-bottom: 0px; text-align:right;">Logged in as: <?php echo $acc\_data["first\_name"]; ?> <?php echo $acc\_data["last\_name"]; ?></p>

        <a style="float:right;"href="main.php?logout=true">Log Out</a>

        </div>

    </header>

    <body>

        <!--Special message body-->

        <?php if($created): ?>

            <em style="color:#11FF11; margin:auto; margin-left: 25%; width:50%; text-align:center; display:inline-block;">Experience Created!</em>

        <?php endif; ?>

        <?php if($signup): ?>

            <em style="color:#11FF11; margin:auto; margin-left: 25%; width:50%; text-align:center; display:inline-block;">Sign Up Successful!</em>

        <?php endif; ?>

        <br>

        <div id="experiences" style="margin-left:7.5%;">

            <h4 style="text-align: center; margin-bottom:0px;">Available Service Experiences</h4>

            <p style="text-align: center; margin-bottom:0px; margin-top:0px;">You are not signed up for these experiences yet. You can learn more about each experience and sign up for them by clicking View.</p>

            <table style="width:100%;">

                <?php

                    $sql = "SELECT service\_activities.name, service\_activities.activity\_id FROM service\_activities

                            LEFT JOIN activity\_participants

                            ON service\_activities.activity\_id = activity\_participants.activity\_id

                            AND activity\_participants.account\_id = {$acc\_data["account\_id"]}

                            WHERE account\_id IS NULL AND organizer\_id != {$acc\_data["account\_id"]} AND start\_date >= CURRENT\_TIMESTAMP

                            ORDER BY start\_date DESC";

                    $result = mysqli\_query($mysqli, $sql);

                    $is\_empty = true;

                    echo "<tr><td style=\"text-align:center;\"><h5 style=\"margin-bottom:0px;\">Future Experiences</h5>These Experiences have not started yet and you can sign up for them.</td></tr>";

                    while($activity = mysqli\_fetch\_assoc($result))

                    {

                        echo "<tr><td>$activity[name]<a href=\"service.php?id=$activity[activity\_id]\"><button style=\"float: right;\" onclick=\"\">View</button></a></td></tr>";

                        $is\_empty = false;

                    }

                    if($is\_empty) {echo "<tr><td>Nothing here :(</td></tr>";}

                    $sql = "SELECT service\_activities.name, service\_activities.activity\_id FROM service\_activities

                            LEFT JOIN activity\_participants

                            ON service\_activities.activity\_id = activity\_participants.activity\_id

                            AND activity\_participants.account\_id = {$acc\_data["account\_id"]}

                            WHERE account\_id IS NULL AND organizer\_id != {$acc\_data["account\_id"]} AND start\_date < CURRENT\_TIMESTAMP AND end\_date >= CURRENT\_TIMESTAMP

                            ORDER BY start\_date DESC";

                    $result = mysqli\_query($mysqli, $sql);

                    $is\_empty = true;

                    echo "<tr><td style=\"text-align:center;\"><h5 style=\"margin-bottom:0px;\">Ongoing Experiences</h5> These experiences have already started, but you can still sign up for them.</td></tr>";

                    while($activity = mysqli\_fetch\_assoc($result))

                    {

                        echo "<tr><td>$activity[name]<a href=\"service.php?id=$activity[activity\_id]\"><button style=\"float: right;\" onclick=\"\">View</button></a></td></tr>";

                        $is\_empty = false;

                    }

                    if($is\_empty) {echo "<tr><td>Nothing here :(</td></tr>";}

                    $sql = "SELECT service\_activities.name, service\_activities.activity\_id FROM service\_activities

                            LEFT JOIN activity\_participants

                            ON service\_activities.activity\_id = activity\_participants.activity\_id

                            AND activity\_participants.account\_id = {$acc\_data["account\_id"]}

                            WHERE account\_id IS NULL AND organizer\_id != {$acc\_data["account\_id"]} AND end\_date < CURRENT\_TIMESTAMP

                            ORDER BY start\_date DESC";

                    $result = mysqli\_query($mysqli, $sql);

                    $is\_empty = true;

                    echo "<tr><td style=\"text-align:center;\"><h5 style=\"margin-bottom:0px;\">Complete Experiences</h5> These experiences have already ended and you can no longer sign up for them.</td></tr>";

                    while($activity = mysqli\_fetch\_assoc($result))

                    {

                        echo "<tr><td>$activity[name]<a href=\"service.php?id=$activity[activity\_id]\"><button style=\"float: right;\" onclick=\"\">View</button></a></td></tr>";

                        $is\_empty = false;

                    }

                    if($is\_empty) {echo "<tr><td>Nothing here :(</td></tr>";}

                ?>

            </table>

        </div>

        <div id="experiences">

            <h4 style="text-align:center; margin-bottom:0px;">Active Service Experiences</h4>

            <p style="text-align: center; margin-bottom:0px; margin-top:0px;">You have signed up for these experiences. To see more information about them, click the View button.

            <table style ="width:100%;">

            <?php

                    $sql = "SELECT service\_activities.activity\_id, service\_activities.name FROM service\_activities

                            INNER JOIN activity\_participants

                            ON service\_activities.activity\_id = activity\_participants.activity\_id

                            WHERE account\_id = {$acc\_data["account\_id"]} AND organizer\_id != {$acc\_data["account\_id"]} AND start\_date >= CURRENT\_TIMESTAMP";

                    $result = mysqli\_query($mysqli, $sql);

                    $is\_empty = true;

                    echo "<tr><td style=\"text-align:center;\"><h5 style=\"margin-bottom:0px;\">Future Experiences</h5>These Experiences have not started yet.</td></tr>";

                    while($activity = mysqli\_fetch\_assoc($result))

                    {

                        echo "<tr><td>$activity[name]<a href=\"service.php?id=$activity[activity\_id]\"><button style=\"float: right;\" onclick=\"\">View</button></a></td>";

                        $is\_empty = false;

                    }

                    if($is\_empty) {echo "<tr><td>Nothing here :(</td></tr>";}

                    $sql = "SELECT service\_activities.activity\_id, service\_activities.name FROM service\_activities

                            INNER JOIN activity\_participants

                            ON service\_activities.activity\_id = activity\_participants.activity\_id

                            WHERE account\_id = {$acc\_data["account\_id"]} AND organizer\_id != {$acc\_data["account\_id"]} AND start\_date < CURRENT\_TIMESTAMP AND end\_date >= CURRENT\_TIMESTAMP";

                    $result = mysqli\_query($mysqli, $sql);

                    $is\_empty = true;

                    echo "<tr><td style=\"text-align:center;\"><h5 style=\"margin-bottom:0px;\">Ongoing Experiences</h5>These Experiences have begun.</td></tr>";

                    while($activity = mysqli\_fetch\_assoc($result))

                    {

                        echo "<tr><td>$activity[name]<a href=\"service.php?id=$activity[activity\_id]\"><button style=\"float: right;\" onclick=\"\">View</button></a></td>";

                        $is\_empty = false;

                    }

                    if($is\_empty) {echo "<tr><td>Nothing here :(</td></tr>";}

                    $sql = "SELECT service\_activities.activity\_id, service\_activities.name FROM service\_activities

                            INNER JOIN activity\_participants

                            ON service\_activities.activity\_id = activity\_participants.activity\_id

                            WHERE account\_id = {$acc\_data["account\_id"]} AND organizer\_id != {$acc\_data["account\_id"]} AND end\_date < CURRENT\_TIMESTAMP";

                    $result = mysqli\_query($mysqli, $sql);

                    $is\_empty = true;

                    echo "<tr><td style=\"text-align:center;\"><h5 style=\"margin-bottom:0px;\">Completed Experiences</h5>These Experiences have ended.</td></tr>";

                    while($activity = mysqli\_fetch\_assoc($result))

                    {

                        echo "<tr><td>$activity[name]<a href=\"service.php?id=$activity[activity\_id]\"><button style=\"float: right;\" onclick=\"\">View</button></a></td>";

                        $is\_empty = false;

                    }

                    if($is\_empty) {echo "<tr><td>Nothing here :(</td></tr>";}

                ?>

            </table>

        </div>

        <div id="experiences">

            <h4 style="text-align:center; margin-bottom: 0px;">My Service Experiences</h4>

            <p style="text-align: center; margin-bottom:0px; margin-top:0px;">These are the experiences you have created. To create a new experience, click the Create Experience button. To view the participants in your experiences, click the View button.

            <table style ="width:100%;">

            <?php

                    $sql = "SELECT service\_activities.activity\_id, service\_activities.name FROM service\_activities

                            WHERE organizer\_id = {$acc\_data["account\_id"]}";

                    $result = mysqli\_query($mysqli, $sql);

                    $is\_empty = true;

                    while($activity = mysqli\_fetch\_assoc($result))

                    {

                        echo "<tr><td>$activity[name]<a href=\"service.php?id=$activity[activity\_id]\"><button style=\"float: right;\" onclick=\"\">View</button></a></td>";

                        $is\_empty = false;

                    }

                ?>

            </table>

            <td><tr><a href="create\_activity.php"><button style="width:100%;" onclick="">Create Experience</button></a></tr></td>

        </div>

    </body>

</html>

create\_activity.php:

<?php

    session\_start();

    if(isset($\_SESSION["user\_id"]))

    {

        $mysqli = mysqli\_connect("localhost", "root", "", "alcuin\_service\_db");

        $sql = "SELECT \* FROM accounts

                WHERE account\_id = {$\_SESSION["user\_id"]}";

        $result = mysqli\_query($mysqli, $sql);

        $acc\_data = mysqli\_fetch\_assoc($result);

    }

    else

    {

        header("Location: login.php");

    }

    if ($\_SERVER["REQUEST\_METHOD"] === "POST")

    {

        $name = $\_POST['activity\_name'];

        $desc = $\_POST['activity\_description'];

        $start\_dt = $\_POST['start\_date'];

        $end\_dt = $\_POST['end\_date'];

        $service\_hrs = $\_POST['service\_hours'];

        #Connect to SQL Database

        $mysqli = mysqli\_connect("localhost", "root", "", "alcuin\_service\_db");

        $sql = "INSERT INTO service\_activities (organizer\_id, name, description, hours, start\_date, end\_date) VALUES ('$\_SESSION[user\_id]', '$name', '$desc', '$service\_hrs', '$start\_dt', '$end\_dt')";

        mysqli\_query($mysqli, $sql);

        header("Location: main.php?created=true");

        mysqli\_close($mysqli);

        exit;

    }

?>

<!DOCTYPE html>

<html lang="en">

    <head>

        <meta charset="UTF-8">

        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <link rel="stylesheet" type="text/css" href="format.css">

        <title>AS Student Service Portal</title>

    </head>

    <header>

        <a href="main.php"><img src="./img/home-button.svg" height="40px"></a>

        <h1 id="title" style="margin:auto; margin-left: 23%; width:50%; text-align:center; display:inline-block;">Alcuin Student Service Portal</h1>

        <div id="acc\_info" style="float:right; margin-right: 5px;">

        <p style="margin-top: 0px; margin-bottom: 0px; text-align:right;">Logged in as: <?php echo $acc\_data["first\_name"]; ?> <?php echo $acc\_data["last\_name"]; ?></p>

        <a style="float:right;"href="main.php?logout=true">Log Out</a>

        </div>

    </header>

    <body>

        <br>

        <h2 id="experience-name" style="margin:auto; width:50%; text-align:center;">Create Experience</h2>

        <br>

        <div id="activity-box">

            <form method="post">

                <label for="activity\_name">Activity Name:</label>

                <br>

                <input type="text" id="activity\_name" name="activity\_name" size="35px" style="margin-bottom: 15px;" required>

                <br>

                <label for="activity\_description">Activity Description (include details not listed elsewhere):</label>

                <br>

                <textarea rows="5" cols="40" name="activity\_description" style="margin-bottom: 15px;" required></textarea>

                <br>

                <label for="start\_date">Start Date & Time:</label>

                <br>

                <input type="datetime-local" id="start\_date" name="start\_date" style="margin-bottom: 15px;" required>

                <br>

                <label for="end\_date">End Date & Time:</label>

                <br>

                <input type="datetime-local" id="end\_date" name="end\_date" style="margin-bottom: 15px;" required>

                <br>

                <label for="service\_hours">Total Service Hours:</label>

                <br>

                <input type="number" min="0" id="service\_hours" name="service\_hours" style="margin-bottom: 15px;" required>

                <br>

                <input type="submit" value="Create Experience" style="margin:5px;">

            </form>

        </div>

    </body>

service.php:

<?php

    session\_start();

    if(isset($\_SESSION["user\_id"]))

    {

        $mysqli = mysqli\_connect("localhost", "root", "", "alcuin\_service\_db");

        $sql = "SELECT \* FROM accounts

                WHERE account\_id = {$\_SESSION["user\_id"]}";

        $result = mysqli\_query($mysqli, $sql);

        $acc\_data = mysqli\_fetch\_assoc($result);

    }

    else

    {

        header("Location: login.php");

    }

    $valid\_id = false;

    $disabled = false;

    $is\_organizer = false;

    if(isset($\_GET['id']))

    {

        $activity\_id = $\_GET['id'];

        $sql = "SELECT \* FROM service\_activities

                INNER JOIN accounts

                ON service\_activities.organizer\_id = accounts.account\_id";

        $result = mysqli\_query($mysqli, $sql);

        while($activity\_data = mysqli\_fetch\_assoc($result))

        {

            if($activity\_id == $activity\_data['activity\_id'])

            {

                $valid\_id = true;

                if($acc\_data['account\_id'] == $activity\_data['organizer\_id'])

                {

                    $is\_organizer = true;

                    break;

                }

                $sql = "SELECT account\_id FROM activity\_participants

                        WHERE activity\_participants.activity\_id = $activity\_id";

                $result = mysqli\_query($mysqli, $sql);

                while($participant\_id = mysqli\_fetch\_assoc($result))

                {

                    if($acc\_data['account\_id'] == $participant\_id['account\_id'] || strtotime($activity\_data['end\_date']) < time())

                    {

                        $disabled = true;

                        break;

                    }

                }

                break;

            }

        }

    }

    if(!$valid\_id) {header("Location: main.php");}

    if ($\_SERVER["REQUEST\_METHOD"] === "POST")

    {

        $extra\_info = $\_POST['extra\_info'];

        $mysqli = mysqli\_connect("localhost", "root", "", "alcuin\_service\_db");

        $sql = "INSERT INTO activity\_participants (activity\_id, account\_id, extra\_info) VALUES ('$activity\_id', '$\_SESSION[user\_id]', '$extra\_info')";

        mysqli\_query($mysqli, $sql);

        header("Location: main.php?signup=true");

        mysqli\_close($mysqli);

        exit;

    }

?>

<!DOCTYPE html>

<html lang="en">

    <head>

        <meta charset="UTF-8">

        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <link rel="stylesheet" type="text/css" href="format.css" />

        <title>AS Student Service Portal</title>

    </head>

    <header>

        <a href="main.php"><img src="./img/home-button.svg" height="40px"></a>

        <h1 id="title" style="margin:auto; margin-left: 23%; width:50%; text-align:center; display:inline-block;">Alcuin Student Service Portal</h1>

        <div id="acc\_info" style="float:right; margin-right: 5px;">

        <p style="margin-top: 0px; margin-bottom: 0px; text-align:right;">Logged in as: <?php echo $acc\_data["first\_name"]; ?> <?php echo $acc\_data["last\_name"]; ?></p>

        <a style="float:right;"href="main.php?logout=true">Log Out</a>

        </div>

    </header>

    <body>

        <br>

        <h2 id="experience-name" style="margin:auto; width:50%; text-align:center;"><?php echo $activity\_data['name'];?></h2>

        <p style="margin:auto;margin-bottom:20px; text-align:center;"><?php echo $activity\_data['description'];?></p>

        <p style="margin:auto; text-align:center;">&ensp;Organized by: <?php echo $activity\_data['first\_name'];?> <?php echo $activity\_data['last\_name'];?>&ensp; Email: <?php echo $activity\_data['email'];?></p>

        <p style="margin:auto; text-align:center;"> Service Hours Awarded: <?php echo $activity\_data['hours'];?> Hours</p>

        <p style="margin:auto; margin-bottom: 20px; text-align:center;">Activity Duration: <?php echo date("m/d/Y", strtotime($activity\_data['start\_date']));?> - <?php echo date("m/d/Y", strtotime($activity\_data['end\_date']));?></p>

        <hr>

        <?php if($is\_organizer):?>

            <h4 style="text-align:center;">Activity Participants:</h4>

            <div id="activity-box" style="height: 500px; overflow:auto;">

                <table>

                    <thead style="text-align:center;">

                        <tr><td>Name</td><td>Information</td><td>Email</td></tr>

                    </thead>

                    <?php

                        $sql = "SELECT \* FROM activity\_participants

                                INNER JOIN accounts

                                ON accounts.account\_id = activity\_participants.account\_id

                                WHERE activity\_participants.activity\_id = $activity\_id";

                        $result = mysqli\_query($mysqli, $sql);

                        while($participant = mysqli\_fetch\_assoc($result))

                        {

                            echo "<tr><td>$participant[first\_name] $participant[last\_name]</td><td>$participant[extra\_info]</td><td style=\"text-align:right;\">Email: $participant[email]</td></tr>";

                        }

                    ?>

                </table>

            </div>

        <?php else: ?>

            <div id="activity-box">

                <h3>Sign-Up</h3>

                <form method="post">

                    <label for="extra\_info">Share any information with the activity organizer:</label>

                    <br>

                    <textarea rows="5" cols="40" name="extra\_info" style="margin-bottom: 15px;"></textarea>

                    <input type="submit" value="Sign Up" style="margin:5px;" <?php if($disabled){echo "disabled";}?>>

                </form>

            </div>

        <?php endif;?>

    </body>

</html>

style.css:

@import url("https://fonts.googleapis.com/css?family=Nunito");html,body,div,span,applet,object,iframe,h1,h2,h3,h4,h5,h6,p,blockquote,pre,a,abbr,acronym,address,big,cite,code,del,dfn,em,img,ins,kbd,q,s,samp,small,strike,strong,sub,sup,tt,var,b,u,i,center,textarea,input,select,dl,dt,dd,ol,ul,li,fieldset,form,label,legend,table,caption,tbody,tfoot,thead,tr,th,td,article,aside,canvas,details,embed,figure,figcaption,footer,header,hgroup,menu,nav,output,ruby,section,summary,time,mark,audio,video{font-family:"Nunito",sans-serif;color: #fff;}body,html{padding:0;margin:0;overflow-x:hidden;background-color:#37474F}nav{font-family:"Nunito",sans-serif;background-color:#212121;color:#fff;display:-webkit-box;display:-ms-flexbox;display:flex}nav header{padding:8px;display:inline}nav header a{text-decoration:none;color:#fff}nav header a:hover{color:#00c853}nav header+input:checked+div{display:block}nav div{display:none;margin-left:auto}nav div ul{list-style:none;display:-webkit-box;display:-ms-flexbox;display:flex;margin:0 10px}nav div ul li a{display:inline-block;padding:8px;color:#fff;text-decoration:none}nav div ul li:hover{background-color:#00c853}nav div ul li ul{right:0;position:relative;background:pink}@media (max-width: 630px){nav{display:-webkit-box;display:-ms-flexbox;display:flex;-webkit-box-orient:vertical;-webkit-box-direction:normal;-ms-flex-direction:column;flex-direction:column}nav header{text-align:center}nav div{margin:auto}nav div ul{padding:0}nav div ul li{border-top-left-radius:5px;border-top-right-radius:5px}nav div ul li[active],nav div ul li.active{border:1px solid #00c853;border-bottom:none}blockquote{border-left:4px solid #00c853;padding:8px 5px;margin:0}blockquote p{font-size:0.4rem}}@media (min-width: 630px){nav div ul li.active,nav div ul li[active]{border:1px solid #00c853;border-bottom:none;border-top:none}nav header label{display:none}}section[container]{max-width:38em;margin:auto;padding:5px}h1{font-size:2.35em}h2{font-size:2em}h3{font-size:1.75em}h4{font-size:1.5em}h5{font-size:1.25em}h6{font-size:1em}a{color:#00c853}a:hover{color:#212121}mark{background-color:#00c853}code{font-family:monospace;background-color:#bdbdbd;padding-left:5px;padding-right:5px}blockquote{border-left:4px solid #00c853;padding:8px 10px;width:100%}blockquote p{font-style:italic;font-size:1.1rem}blockquote footer::before{content:"\2014 \00A0"}blockquote footer cite{font-style:italic;color:#bdbdbd}pre{background:#eee;overflow-x:auto;text-align:left;padding:5px}pre code{display:block;padding:0 10px;background:transparent}table{display:table;padding:5px;border-collapse:collapse}table thead,table tbody{text-align:left}table tr th,table tr td{padding:5px 10px;border-bottom:1px solid #00c853}div[overflow]{overflow-x:auto;max-width:100vw}div[overflow] ::-webkit-scrollbar{height:0}img{max-width:100%;border-radius:5px}form div{display:-webkit-box;display:-ms-flexbox;display:flex;-webkit-box-orient:vertical;-webkit-box-direction:normal;-ms-flex-direction:column;flex-direction:column;margin:10px 5px}form div p{margin:0px}form input,form select{font-size:1rem;padding:5px;border:1px solid #bdbdbd;color:#212121}form input:active,form input:focus,form select:active,form select:focus{outline-color:#00c853}form input[type="submit"]{padding:10px;background-color:#00c853;color:#000;border-radius:5px;border:none;cursor:pointer}form input[type="submit"]:active,form input[type="submit"]:focus{outline:none}form input[type="submit"]:active{background-color:#212121;color:#00c853}form input[type="submit"]:disabled{background:#bdbdbd;cursor:not-allowed}form input[type="submit"][secondary]{background-color:#212121;color:#00c853}form input[type="submit"][secondary]:active{background-color:#00c853;color:initial}textarea{color:#212121;width:-webkit-fill-available;font-size:1rem;padding:5px}textarea:active,textarea:focus{outline-color:#00c853}button{padding:10px;background-color:#00c853;color:#000;border-radius:5px;border:none;cursor:pointer}button:active,button:focus{outline:none}button:active{background-color:#212121;color:#00c853}button:disabled{background:#bdbdbd;cursor:not-allowed}button[secondary]{background-color:#212121;color:#00c853}button[secondary]:active{background-color:#00c853;color:initial}body>footer{background-color:#212121;position:relative;bottom:0;width:100%;padding:5px;color:#fff}

/\* GitHub Link: https://github.com/raj457036/attriCSS \*/

/\*# sourceMappingURL=brightlight-green.css.map \*/

#login-box {

    border: 1px solid darkgray;

    text-align: center;

    display: inline-block;

    padding: 70px;

    padding-top: 35px;

    padding-bottom: 10px;

    position: fixed;

    top: 50%;

    left: 50%;

    -webkit-transform: translate(-50%, -50%);

    transform: translate(-50%, -50%);

}

#activity-box {

    text-align: center;

    display: inline-block;

    position: fixed;

    left:50%;

    -webkit-transform: translate(-50%, 0%);

    transform: translate(-50%, 0%);

}

#experiences {

    width:20%;

    float:left;

    margin:40px;

    padding:50px;

    padding-top:20px;

    border: 1px solid darkgray;

}

# **Appendix C**

**Ms. xx Evaluation Meeting Transcript:**

Me: Alright, so that’s the website for now. Uh, right now the extensions I have for myself are for it to work across multiple networks so someone can use it from home and a remote emailing service that notifies people of service opportunities. But overall what do you think of the product?

Ms. xx: I think it’s really good! The account system makes sense and I can use it really easily. I really like the colors and the overall look of it. I thought it would be a lot more bare bones than it looked, but it looks really good.

Me: Thank you! I found the template online but the organization was done myself. What did you think of the features themselves? Like the sign ups, activity form, all that?

Ms. xx: I mean I don’t know much about programming or computer science but it’s really easy to use for me and almost everything I want is there! I really like how the main page is laid out, its really easy to follow with the descriptions, and the other pages work really well with it. I also liked the service page and how its separated into two parts. It helped emphasize the information, but kept the sign-up form on the bottom as well. If you are looking for feedback, I guess the activity creation form could use more information because there are a lot of things that students would want to know, but teachers aren’t required to give.

Me: What things do you think I should add?

Ms. xx: I mean the CAS stuff would be nice like strands and learning outcomes.

Me: Right, I remember we discussed that last time too. I had trouble putting them in the form though because a lot of people don’t know what CAS is or why they need to fill it out. Do you think there is a way I could add an optional fill out?

Ms. xx: I mean, again I don’t know much about programming, but couldn’t you just put text recommending people put it in the description like you did?

Me: I suppose I could. Is there anything else you’d recommend adding?

Ms. xx: I mean, other than what you already mentioned, I think that’s really good! We should definitely try and integrate this with the current StuCo website so people can actually start using it.

Me: For sure, thanks for your time!