

SCVTHS FBLA: CODING and Programming

By Shanmuka Sadhu

Introduction: My task

This year for Coding & Programming FBLA competition, the task was to create a computer program that tracked Community Service Awards program in your local chapter. My program checked off all of the requirements provided on the FBLA website.

Different Aspects of my program:

1. GUI Application(Frontend)
2. Database(Backend)
3. Interaction between Frontend and Backend

SDLC: Software Development Life Cycle

SDLC(Software Development Life Cycle)- a process used by the software industry to design, develop and test different types of software.

Steps in SDLC:

1. Planning and Requirement Analysis
2. Design
3. Implementation

Planning and Requirement Analysis

Access our different options and identify what options will match the goal of the project. We look at the different types of languages and different types of databases.

Programming Languages we can possibly use:

1. Python & Tkinter GUI platform
2. C/C++ & GTK GUI environment
3. PHP & Bootstrap environment

Databases that we can possibly use:

1. MySQL
2. MongoDB
3. Django

Design

We have decided to use Bootstrap for our frontend because it allows us to create a easy to interface GUI.

For our database, we chose to use MySQL because it was open source and very easy to use. We can connect our frontend to PHPmyadmin.

For the interaction between MySQL and Bootstrap, we are using PHP. PHP is easy to connect with the MySQL database.

Bootstrap 4

Bootstrap is a free and open-source tool for creating responsive websites and web applications.

MySQL

MySQL is an open-source relational database system. It's easy to interact with and is very popular.

PHP

PHP is a open source general-purpose scripting language that is used for web development

Implementation

Steps during implementation stage:

1. First was creating the frontend of the project.
 - a. Template
 - b. Buttons
 - c. Forms
 - d. Tables
 - e. Font Awesome
2. Creation of the Database.
 - a. Creating the databases
 - b. Creations of the tables in the databases
3. Getting data from forms and connecting them with the DB.
4. Creating the reports.
 - a. Creating the graphs
 - b. Identifying Gold, Silver, and Bronze awards

Frontend

We used a template for the base of the frontend. This template was provided by Bootstrap.

Link:

<https://getbootstrap.com/docs/4.0/examples/>

We also had to create multiple forms. We had to create forms for:

1. Sign In
2. Sign Up
3. Add Student
4. Add Award
5. Add Service hours
6. Update Student
7. Update Award
8. Update Service hours

Font Awesome: Font Awesome has a huge collection of icons was used for all of the icons.

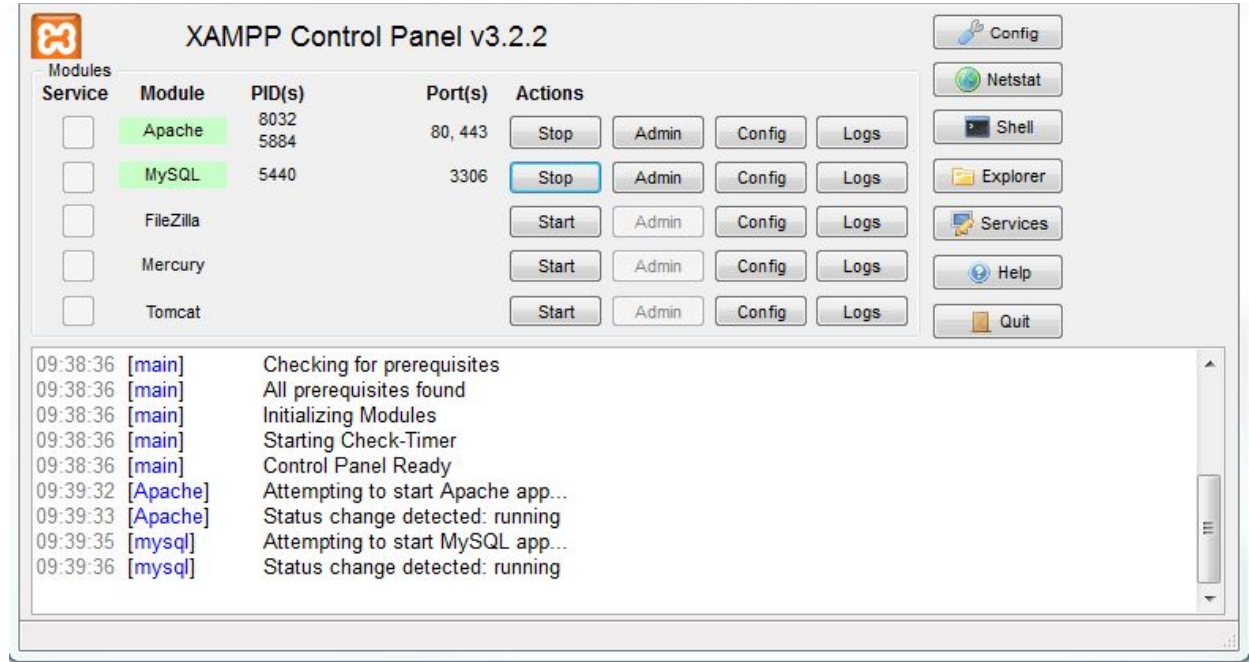
The forms contained:

1. Checkboxes
2. Labels
3. Submit Button
4. Password
5. Email
6. Dropdown lists
7. Date

Turning on MySQL & Apache Web Server

The program is being run on the localhost on port 8080. To access our localhost we use Xampp.

Xampp Control center allows for us to turn on our Apache Web Server and MySQL.



Creation of the Database

We used MySQL as our database.

Code used to connect to MySQL db in PHP.

```
k?php
/* Database credentials. Assuming you are running MySQL
server with default settings */
define('DB_SERVER', 'localhost');
define('DB_USERNAME', 'root');
define('DB_PASSWORD', '');
define('DB_NAME', 'mydb');

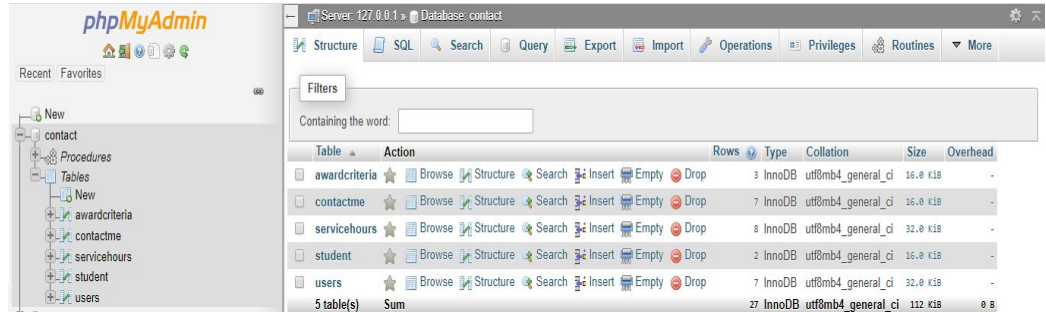
/* Attempt to connect to MySQL database */
$link = mysqli_connect(DB_SERVER, DB_USERNAME, DB_PASSWORD, DB_NAME);

// Check connection
if($link === false){
    die("ERROR: Could not connect. " . mysqli_connect_error());
}
?>
```

The tables we created:

1. awardcriteria
2. contactme
3. servicehours
4. student
5. users

PHPmyAdmin helped accessing our databases and creating tables.



The screenshot shows the PHPmyAdmin web interface. On the left, a tree view shows the database structure with a 'New' button and a list of tables: contact, awardcriteria, contactme, servicehours, student, and users. The main panel displays the 'Structure' tab for the 'contact' database. It shows a table with 5 columns: Table, Action, Rows, Type, Collation, Size, and Overhead. The table contains 5 rows of data for the tables: awardcriteria, contactme, servicehours, student, and users. The table is created with InnoDB engine and utf8mb4_general_ci collation.

Table	Action	Rows	Type	Collation	Size	Overhead
awardcriteria	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 K18	-
contactme	Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_general_ci	16.0 K18	-
servicehours	Browse Structure Search Insert Empty Drop	8	InnoDB	utf8mb4_general_ci	32.0 K18	-
student	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 K18	-
users	Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_general_ci	32.0 K18	-
5 table(s)	Sum	27	InnoDB	utf8mb4_general_ci	112 K18	0 B

Getting Data from forms to DB

PHP allows to get the data from the components of the form using the name element. After the submit button is pressed, the data is immediately transferred to the PHP code. The PHP code will validate the user's response and decide to store it in the database or throw a response back to the user.

Getting Student name from form in PHP:

```
if($_SERVER["REQUEST_METHOD"] == "POST"){
    // Validate name
    $input_name = trim($_POST["name"]);
    if(empty($input_name)){
        $name_err = "Please enter a name.";
    } elseif(!filter_var($input_name, FILTER_VALIDATE_REGEXP, array("options">=>array("regexp">="/^[a-zA-Z\s]+$/"))){
        $name_err = "Please enter a valid name.";
    } else{
        $name = $input_name;
    }

    // Validate student number
    $input_number = trim($_POST["number"]);
    if(empty($input_number)){
        $number_err = "Please enter student number.";
    } else{
        $number = $input_number;
    }
}
```

Example of data validation:

```
// Check if username exists, if yes then verify password
if(mysqli_stmt_num_rows($stmt) == 1){
    // Bind result variables
    mysqli_stmt_bind_result($stmt, $id, $username, $hashed_password);
    if(mysqli_stmt_fetch($stmt)){
        if(password_verify($password, $hashed_password)){
            // Password is correct, so start a new session
            session_start();

            // Store data in session variables
            $_SESSION["loggedin"] = true;
            $_SESSION["id"] = $id;
            $_SESSION["username"] = $username;

            // Redirect user to welcome page
            header("location: ../dashboard/index.php");
        } else{
            // Display an error message if password is not valid
            $password_err = "The password you entered was not valid.";
        }
    }
} else{
    // Display an error message if username doesn't exist
    $username_err = "No account found with that username.";
}
```

Creating Reports

We read data from the database multiple times. We read from the database when signing in, when identifying who gets award, creating the model, and updating the submissions.

```
<!-- Graphs -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.7.1/Chart.min.js"></script>
<script>
var ctx = document.getElementById("myChart");
var myChart = new Chart(ctx, {
  type: 'line',
  data: {
    labels: <?php echo $labels; ?>,
    datasets: [{
      data: <?php echo $data; ?>,
      lineTension: 0,
      backgroundColor: 'transparent',
      borderColor: '#007bff',
      borderWidth: 4,
      pointBackgroundColor: '#007bff'
    }]
  },
  options: {
    scales: {
      yAxes: [{
        ticks: {
          beginAtZero: false
        }
      }]
    },
    legend: {
      display: false,
    }
  }
});
</script>
```

Reading data from Database

```
if(isset($_GET["student_id"]) && !empty(trim($_GET["student_id"]))) {
    // Get URL parameter
    $id = trim($_GET["student_id"]);

    // Prepare a select statement
    $sql = "SELECT * FROM student WHERE student_id = ?";
    if($stmt = mysqli_prepare($link, $sql)){
        // Bind variables to the prepared statement as parameters
        mysqli_stmt_bind_param($stmt, "i", $param_id);

        // Set parameters
        $param_id = $id;

        // Attempt to execute the prepared statement
        if(mysqli_stmt_execute($stmt)){
            $result = mysqli_stmt_get_result($stmt);

            if(mysqli_num_rows($result) == 1){
                /* Fetch result row as an associative array. Since the result set
                contains only one row, we don't need to use while loop */
                $row = mysqli_fetch_array($result, MYSQLI_ASSOC);

                // Retrieve individual field value
                $name = $row["student_name"];
                $number = $row["student_number"];
                $grade = $row["student_grade"];
            } else {
                // URL doesn't contain valid id. Redirect to error page
                header("location: error.php");
                exit();
            }
        } else {
            echo "Oops! Something went wrong. Please try again later.";
        }
    }
}
```

Demotime