**Final Report**

**A)**

**Student Information:**

* **(1A)** Name: Paul Leung
  + ID: 14141026
  + Address: 2069 West 13th Street
  + Cell: 347-417-0926
* **(1B)** CISC 4900: FALL 2019

**Supervisor Contact Info:**

* **(1C)** Name: Gabriel Yarmish
  + Email: [yarmish@sci.brooklyn.cuny.edu](mailto:yarmish@sci.brooklyn.cuny.edu)
  + Phone: 718.951.5000 x2071
  + Office:CUNY Brooklyn College, 1214 Ingersoll Hall

**B) Individual Project**

**C) Supervisor Relationship:** CISC 3171 Professor (Fall 2019)

**D)**

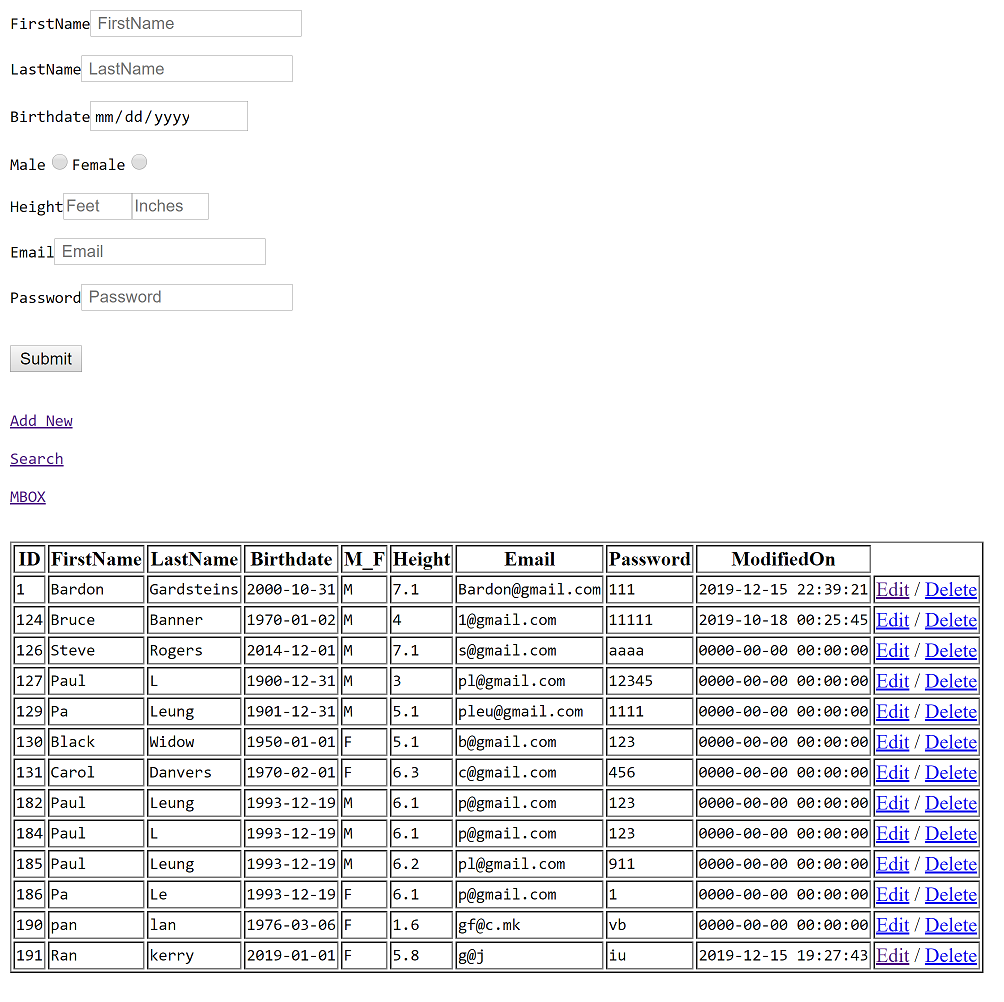
* **Introduction:** Front-end and Backend development of a Resume Website. The website creates a portfolio of an individual based on the input of user information (such as First/Last name, etc.). The information is then stored into a MySQL table called “RBasicInformation”, which can be queried to do a variety of Insertion, Deletion, Editing, and Selection commands. This website also can parse MBOX files, where each field of an email (Headers/Body Information and Attachments), is stored in another MySQL table named “messages”.
* **General Problem:** How to create and read an efficient database of resumes.
* **Solution:** The use of Objected-Oriented PHP, in addition to using PHP Data Objects (PDO) extension, providing an interface for accessing databases in PHP. Also, in contrast to hardcoding HTML forms, the creation of HTML form data is done by traversing through our MySQL table and generating respective inputs for each field.
* **Scope of Work Performed:** 
  + Creation of website domain and MySQL database.
  + Configuring SFTP connection on Visual Studio Code to our website.
  + Creation of HTML input forms.
  + INSERTION query to update database information with user input.
  + Retrieval of database Column Names and Data Types from information Schema.
  + Printing a table of our database on website using HTML tables and SQL Select Query.
  + Alternative implementation of printing HTML input forms automatically generated based on current database instead of hard-coding.
  + Implementation of Edit and Delete buttons for user information through UPDATE and DELETE query.
  + Implementation of Insertion query on separate page (Add.php)
  + Implementation of PHP Session-Based Flash messages through global Session key to indicate successful/unsuccessful queries of Edit and Delete.
  + Implementation of Search to filter through profiles based on specified inputs. To be displayed on a generated table.
  + Implementation of PDO prepared statements to prevent SQL injection for Search.
  + Created a separate Search on the website that allows SQL injection for testing purposes.
  + Implementation of a MBOX parser, that stores each parsed field of an email in a separate MySQL table named “messages”.
  + Storing MBOX attachments into database.
* **Summary:** A simple website that creates a portfolio of a user based on inputted HTML form data. Can also parse and store fields of an email and respective attachments into a MySQL database.

**E) Due to time constraints, the following was not completed:**

* As we primarily focused on functionality, the Web Design of the website through CSS has been minimal.
* Creation of SQL Legend in search.php, that displays the Table/Column names of each database for reference (Caused interference with the Search function).
* Generic SQL Search only accepts inputs for SQL tests(UPDATE,INSERTION,DELETE); doesn’t output table data when using Select query.
* Uploading MBOX files through user input. MBOX files are currently uploaded and parsed through source code path.
* Debugging Edit’s “Height” field. Difficult since we store Height as one field but generate two forms through Feet and Inches.
* Debugging Add Users. We separated the insertion of form data to another page; however, errors occur in the add.php page when we remove the form from our home page(index.php).

**F) Program Documentation:**

* **Index.php**
  + Homepage for the website. Displays links to Add User information, Search for a user, and a link to parse MBOX files. Also displays the entire database of user portfolios, along with options to Edit and Delete next to each row.

****

* **add.php**
  + Linked through “Add New” on the homepage, this page generates a HTML form of user information that, when submitted, stores a new user profile in the database.
  + Forms are automatically generated, in contrast to hard coding, by traversing through the database and matching the data-type of each field in the MySQL table to create their respective column forms.

switch ($row[DATA\_TYPE]) {

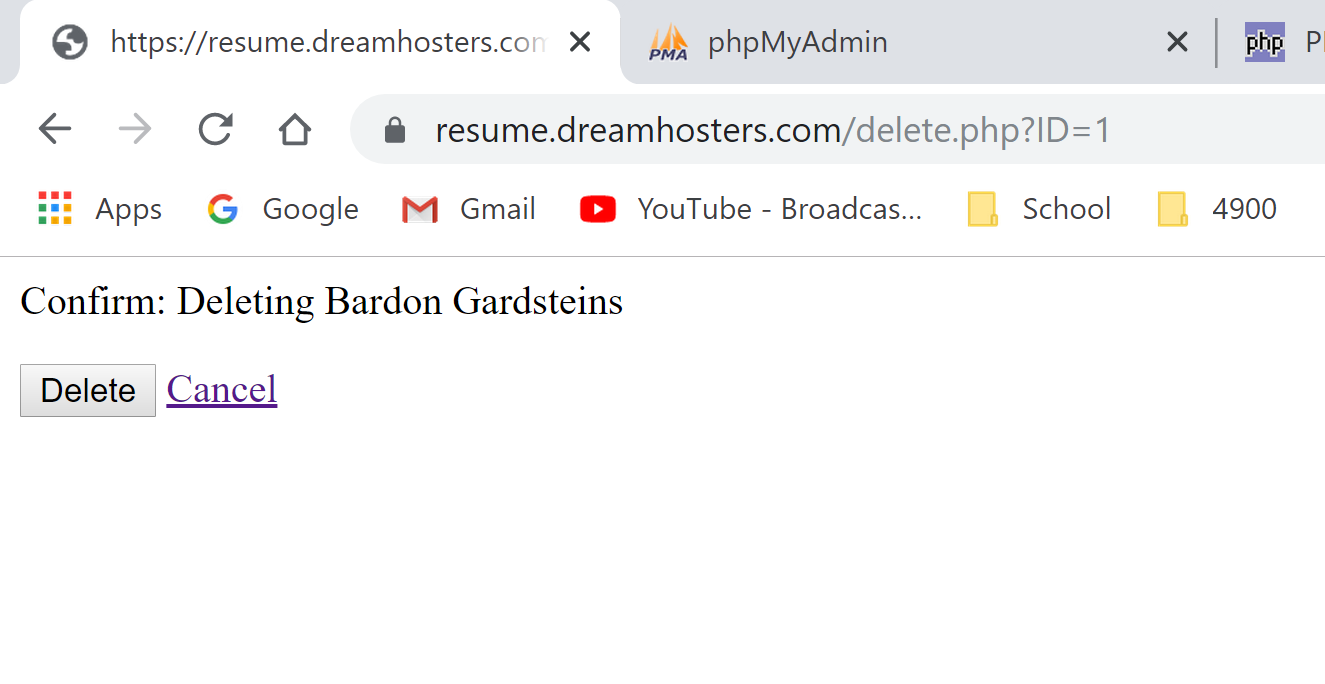
            //First and Last name

            case "text":

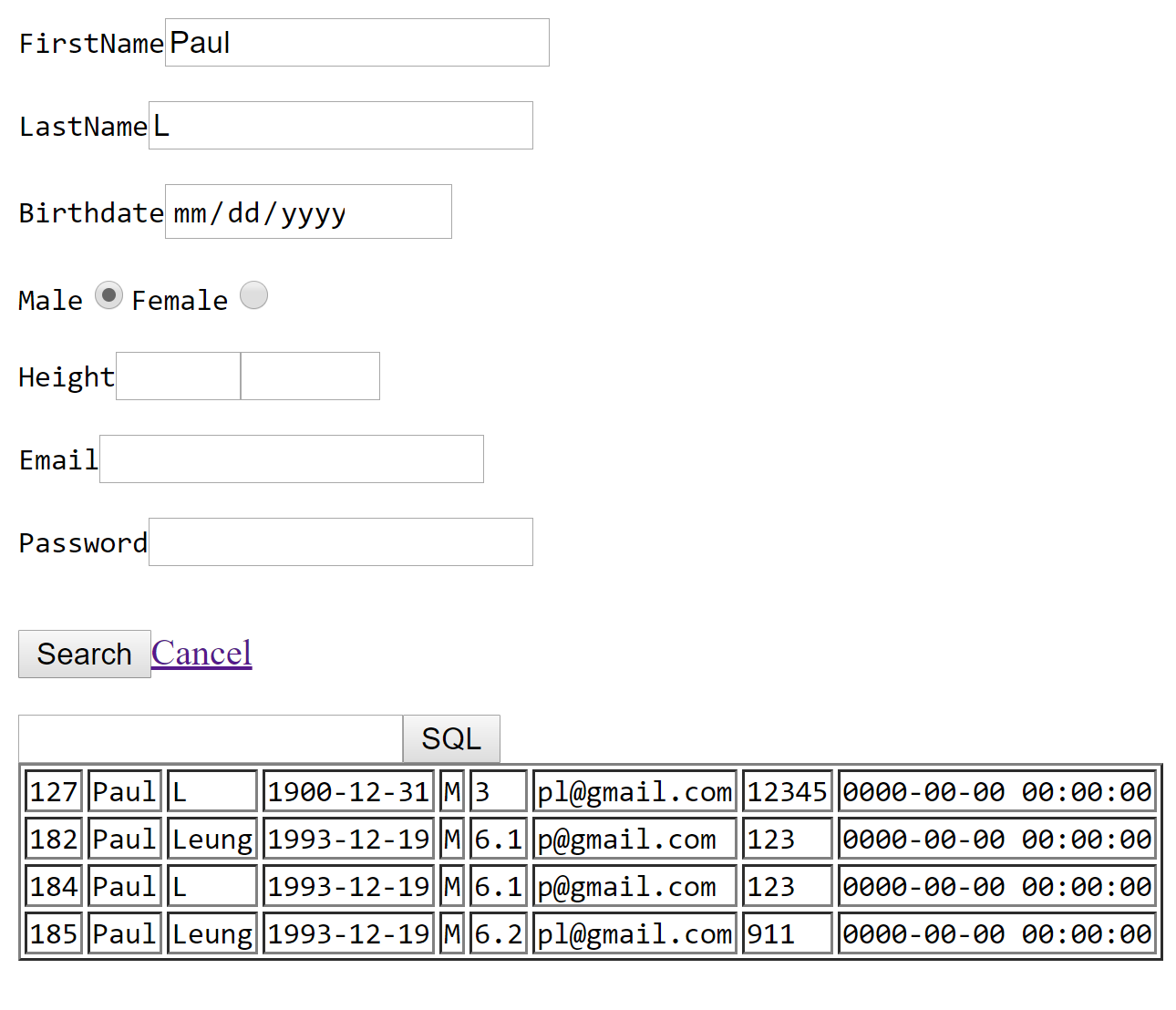
                echo $row[COLUMN\_NAME];

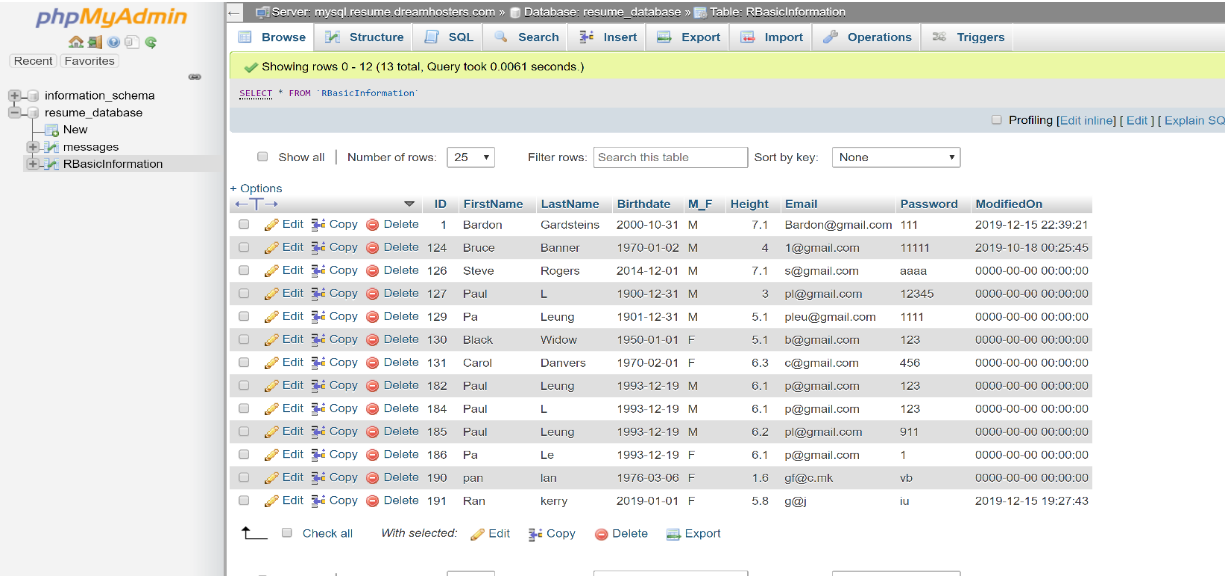
                echo '<input type="text" name="'.$row[COLUMN\_NAME].'" required placeholder=" '. $row[COLUMN\_NAME]. ' "><br><br>';

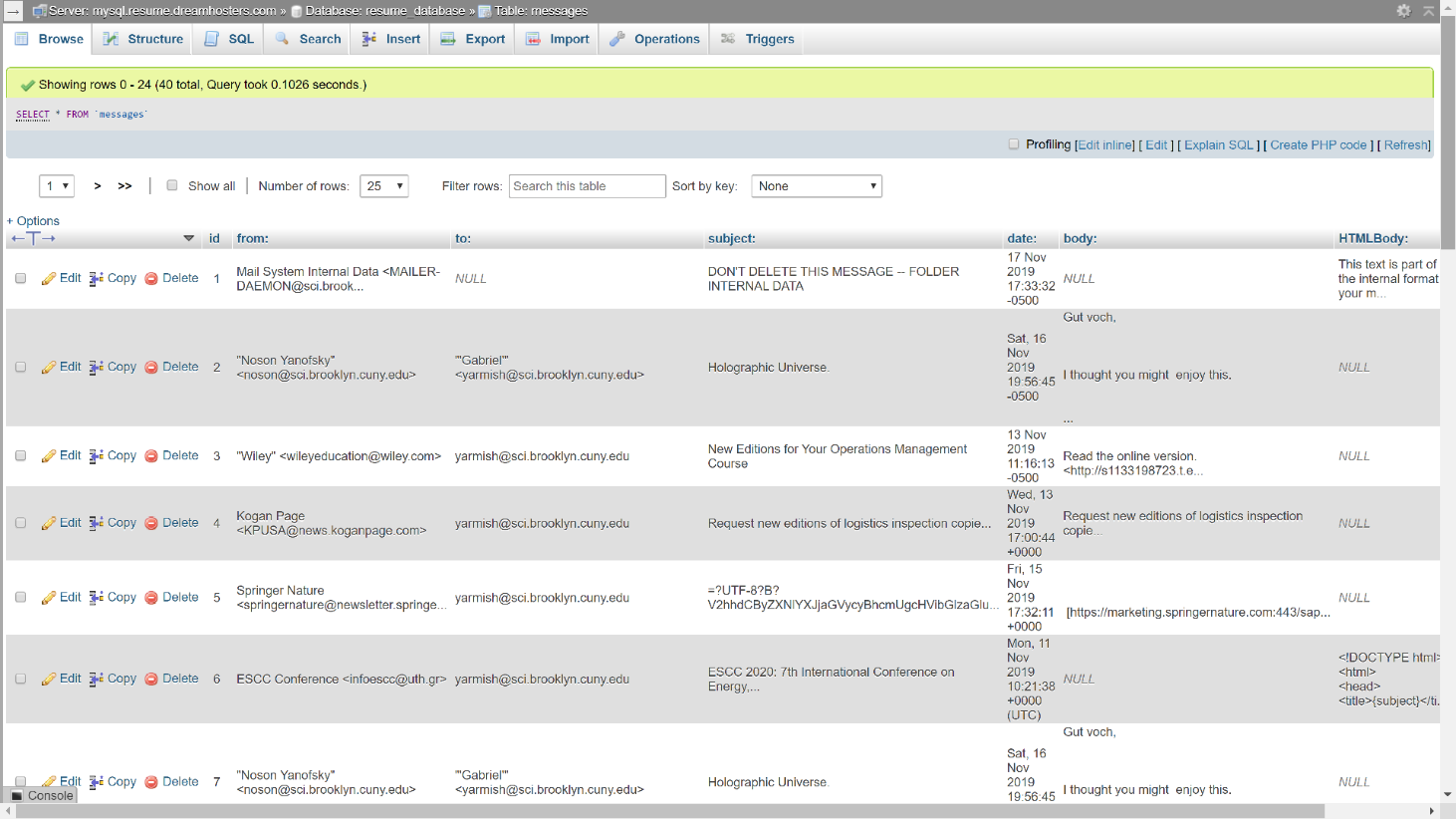
* + Switch($row[Data\_Type]) contains the data type of the current column in the table. We match this data-type in a switch statement, and each case condition prints the respective HTML form.
  + Inputs:
    - FirstName: First name of the user.
    - LastName: Last name of the user.
    - Birthdate: Birthday of the user. Can only range from birthdays of 1900-01-01 to 2019-01-01.
    - M\_F: Gender of the user.
    - Height: Height of the user. Appended through two separate forms (Feet and inches). Range from ‘11 Inches to 7 Feet ‘11 inches.
    - Email: Email of the user. Must contain “@” and “.”
    - Password: Password of the user. Currently no restrictions apply.
  + Outputs:
    - User portfolio entry in database table “RBasicInformation”.
    - Jumps to index.php on “Submit”
    - **$\_POST[‘FIELD’]:** returns POST data of each respective input field.
    - **$\_SESSION[‘success’]:** Returns “Record Added” upon successful query.
    - **$\_SESSION[‘error’]:** Returns “bad data” for invalid form inputs.
* **edit.php**
  + Linked through “Edit” on the homepage next to each row of the table. Clicking this button allows access to change a user’s information based on the previous HTML form fields.
  + Input: Same HTML forms as add.php. However, each field is already filled based on existing data on the user.
  + Output:
    - Updated user portfolio in database table “RBasicInformation”
    - Jumps to index.php on “Submit”
    - **$\_SESSION[‘success’]:** Returns “Record Updated” upon successful query.
    - **$\_SESSION[‘error’]:** Returns “Bad value for ID” if no $id is set.
* **delete.php**
  + Linked through “delete” on the homepage next to each row of the table. Clicking this button allows the user to delete their portfolio.
  + Input: $id of the user (Primary key), $FirstName, $LastName
  + Output:
    - ‘Confirm deleting $FirstName $LastName
    - **$\_SESSION[‘success’]:** Returns “Record deleted” upon successful query.
    - **$\_SESSION[‘error’]:** Returns “Missing ID” if no $id is set.

****

* **search.php**
  + Linked through “Search” on the homepage. This page allows the user to search through the table of portfolios in “RBasicInformation”. HTML fields are similar to previous entries, and each field can be left incomplete or empty. The use of the “%” wildcard in our search allows incomplete fields to display user information containing up to those strings.
  + Input: HTML Fields as previously mentioned.
  + Output: A table of the user as a result of the search.



* **test\_message\_decoder.php**
  + Linked through the “mbox” button on the homepage. This page should accept a user’s mbox file for parsing, where each parsed field is stored in the database table “messages”.
  + Input: A file containing extension “.mbox”
  + Output: Returns an associative array of email headers and body information.
    - $decoded[$message]: The associative array of parsed data for this $message. Contains header information of the email.
    - $results[]: The associative array containing email data such as attachments and body data.
* **MySQL Tables:**
  + **RBasicInformation:** Containing user profiles.
  + **messages:** Containing email headers, body message, and attachments.

****

**G) Project Log:**

**September 5, 2019 (5:30 PM-6:00 PM)**

* General outline of project tasks for the semester. Discussion of the best possible tools to approach this project, ability to port the website to apps. Can any code be salvaged when porting to mobile in future? Scheduling of project meeting times.

**September 9, 2019(2:30 PM-3:30 PM)**

* Setting up access to the database and website domain.
* Configuring SFTP connection on Visual Basic Code.
* Discussion of HTML input forms
* Introduction to PHP code

**September 10, 2019 (2:00PM-6:00PM)**

* Review of HTML and CSS fundamentals
* Studying of various forms for user input

**September 13, 2019(10:00AM-11:00AM)**

* Troubleshooting SFTP connection
* Discussion of inserting user data into database
* Study of introductory PHP concepts(**4:00PM-7:00PM)**

**September 15, 2019(3:00PM-8:00PM)**

* Creating HTML input forms
* Attempt at inserting form data into database

**September 16, 2019(2:30PM-3:00PM)**

* Troubleshooting insertion query
* Review of PDO and queries in PHP
* Discussion of retrieving column names from database instead of hardcoding

**September 17, 2019(3:00PM-8:00PM)**

* Study of PDO functions and various PHP tutorials

**September 18, 2019(10:00AM-2:00PM)**

* Successful completion of inserting form data into database
* Brief discussion of empty form data inserted into database upon page refresh**(8:50PM-9:00PM)**

**September 20, 2019(10:00AM-11:30AM)**

* Discussion of separating PHP files (ie; login information in separate login PHP file)
* Tasked with creating a table of database information printed on webpage using a foreach loop.
* Discussion of “ISSET” function
* Discussion of SQL injection and input sanitization.

**September 24, 2019(2:00PM-6:00PM)**

* Successful completion of creating table with database information. Retrieval of information consisted of foreach loops using an array of column names.
* Implemented require PDO to fetch database information only once.
* Implementing ISSET function for each field of input prior to insertion.
* Brief phone call discussing the addition and separation of PHP files for our project (~5:30PM)

**September 27, 2019(10:00AM-11:30AM)**

* Meeting: reviewing progress and debugging of ISSET functions using Try-catch blocks.
* Tasked with looping input form data without hardcoding each data field to be retrieved. Possibly by retrieving data type of each column field and using a switch case statement for appropriate input form.

**September 28, 2019(12:00PM-3:30PM)**

* Retrieved Data Types and Column Name of each column in RBasicInformation table through Information Schema.
* Implementation of generating HTML forms through switch statements, each case matching the data type of the form to be generated.

**September 30, 2019(2:00PM-3:30PM)**

* Meeting: Review of example Edit, Delete, and Add files on Professor’s online folder.
* Review of errors on HTML forms:
  + Birthdate and Gender fields can be left empty.
  + Email not validating @ on input

**October 3, 2019(2:00PM-7:00PM)**

* Implementation of Edit, Delete, and Add files. Repurposed example code provided by Professor for our auto-generated forms.
* Debugged Birthdate and Gender fields. Added required attributes for the data.
* Discovered input type for Email for HTML forms. Requires “@” and “.”

**October 4, 2019(10:30PM-12:00PM)**

* Meeting:
  + Error of Height HTML-Form in edit.php
  + Insertion query should be BEFORE HTML forms.
  + Discussion: How do people store gender and height fields in database.

**October 8, 2019(2:00PM-5:00PM)**

* Adding edit and delete links to each row of the table in index.php.
* Debugging PHP Flash-Session message for edit and delete.
* Researching $\_POST data. Understood $\_POST is always set, isset() doesn’t work.
  + $\_POST data is always type string
* Debugging Add New Button: remains on same page even when clicked!

**October 11, 2019(10:00AM-11:00PM)**

* Meeting: Discussion of implementing search function to filter and display user profiles.
* Review of Prepared Statements and try-catch in PHP.

**October 13,2019(1:00PM-5:00PM)**

* Attempt at creating search function through Select query.
  + Reviewing PDO and Select query execution.

**October 15, 2019(9:00PM-12:00AM)**

* Received email from Professor discussing basic site issues.
* Debugged insert query on homepage.
* Debugged Edit function. Only allowed 3 fields to be edited before.
* Debugged Delete: Not returning any result

**October 18, 2019(10:30AM-10:40AM)**

* Phone call instead of meeting. Discussed search functionality again.
  + A textarea tag that allows a full SQL search. No SQL protection.
  + Discussed allowing incomplete forms to search. Uses AND operator for all fields search.
  + Discussed creating a legend that displays table/column names for reference.

**October 22, 2019(1:0PM-5:00PM)**

* Implementation of search function. Review and use of SQL wildcards, specifically “%”, for incomplete form data.

**October 24, 2019(1:0PM-3:00PM)**

* Implementation of generic search with no SQL protection.
* Reviewing HTML input forms

**October 25, 2019(10:00AM-11:30AM)**

* Review of search function.
  + Need to implement prepared statements to allow for SQL protection for main search.

**October 27, 2019(1:00PM-4:00PM)**

* Attempt to debug search through prepared statements. Errors for Gender and Height Fields.
* Studying alternative attempts to query select.

**November 1, 2019(10:00AM-12:00AM)**

* Debugging Search function with professor.

**November 4, 2019(2:30PM-3:30PM)**

* Meeting with Professor:
  + Discussed and debugged errors with Search.
  + Discussed errors with $\_Session key printing proper messages on successful/unsuccessful queries.

**November 8, 2019(10:00AM-12:00PM)**

* Meeting with Professor:
  + Discussed mbox parsing. Tasked to find a class online that does this successfully and do a black-box implementation
  + Tasked: Create an excel document that lists the mbox formats parsed.
  + Tasked: Create a new table in database for mbox fields.
  + Tasked: Write a class that will take the mbox file and insert into table.

**November 10, 2019(2:00PM-7PM)**

* Reading through numerous links Professor sent for mbox research.
* Found possible class to implement

**November 11, 2019(2:00PM-3:00PM)**

* Meeting with professor. Once again discussed mbox files and headers we want to parse.
* Review of the mbox class I choose for black-box implementation.

**November 14, 2019(2:00PM-4:00PM)**

* Studying mbox class we are using through the provided User Documentation.

**November 15, 2019(10:00AM-11:30AM)**

* Review of user documentation of mbox file with professor.
* Discussed proper PHP practices and refactoring of code.
* Asked: What can we do to make maintainability?

**November 18, 2019(2:00PM-3:00PM)**

* Meeting with Professor:
  + Discussed Professor’s sample mbox file.
  + Downloaded mbox viewer.exe to view messages.
  + Discussed how to view mbox files in HTML (view source)

**November 27, 2019(8:30PM-9:00PM)**

* Meeting with Professor after class.
  + Discussed rubric for grade

**November 30,2019(1:00PM-7:00PM)**

* Attempt to parse mbox
* Finding easier, more concise mbox alternative.
* Successful parse of new mbox file, however difficulty in Insertion to table.

**December 1,2019(2:00PM-5:00PM)**

* Another attempt to insert mbox to table.
* Attempt to debug search function.

**December 4,2019**(**2:15PM-3:30PM)**

* Meeting with professor:
  + Discussion to go back to original mbox parser.
  + Traversal of decoded message parts through associative arrays.

**December 7, 2019(12:00PM-5:00PM)**

* Traversing and sketching through multidimensional associative array outputted by mbox file.
* Creation of new database table to match these parsed outputs.
  + Testing access to specific mbox fields we want (to,from,date,etc)
* **(8:00PM-11:00PM)**
  + Successful insertion of mbox data into parsed fields

**December 8, 2019(11:00AM-5:00PM)**

* Successful debugging of search function
  + Used alternative method of appending wildcards to prepared statements.
  + Created excel sheet csv for mbox parsed data.
* Refactoring code for maintainability.
* Minimal CSS style sheet for table on homepage.

**December 13, 2019(10:00AM-12:00PM)**

* Meeting with Professor:
  + Discussed mbox attachments.
  + Discussed SQL generic search, no output available for Select queries.

**December 14, 2019(1:00PM-7:00PM)**

* Traversing through mbox associative arrays to discover Attachment data.
* Successfully accessing Attachment data and inserting into table.
* Creating separate table field for email messages with HTML body.
* Failed attempt at creating SQL Legend.
* Started on Final Report.

**December 15, 2019(1:00PM-5:00PM)**

* Completing majority of Final report.
  + Organizing project log.

**December 16, 2019(2:30PM-4:30PM)**

* Review of Final Report with professor.
  + Added adjustments based on Professor’s input.

**H) Evaluation:** Learning PHP from scratch. I started this course with no background in this language, and throughout this semester of independent study, I have become competent coding with it. Besides PHP, I had to refresh my knowledge of HTML/CSS and SQL. This was also my first time building a website from scratch, and it was satisfying learning new tools to accomplish it.