Project Phase03

Objectives

- Create a database from a SQL script
- Create a database and user on your Webhost
- Separate db_credentials.php file from your code
- Upload your local repo to GitHub
- Load your code on your Webhost

The tutorials

Watch chapters 5 and 6 from the InLearning tutorial PHP and MySQL Basics: Part 1.

Use the MySQL CLI (Command Line Interface) or PHPMyAdmin while using MySQL. If you run into a socket error when launching MySQL from the command line, copy and paste the error into Google. If you are still stuck, then contact your instructor.

Once you have finished the tutorials, we will continue building the Salamander site.

Setup

Code

- Continue working in your **sas** folder. Do not create separate folders for each phase. I have done this only for educational purposes.
- Download the Starter files or continue building from your completed code to start this phase.

- Download the **salamander.sql** file from Moodle and paste it into your **sas** folder. This makes access to the file accessible.

Git revisited

These instructions are for anyone who needs to start fresh with Git.

Mac

```
cd /Applications/MAMP/htdocs/web182/sas
```

Windows

```
cd c:/xampp/htdocs/web182/sas
```

Now your are in the correct file path. Start using git.

In Git, files are in three stages.

1. Untracked
2. Staged (use add . to stage)
3. Committed

Initialize Git

```
git init
```

Check your status

```
git status
```

Stage the files

```
git add .
```

Commit

```
git commit -m"Initial commit."
```

GitHub Repo

Continue to use the repo you started in the previous phase. These instructions are for anyone

who needs to create a new repo.

Time to create a repo on GitHub's site.

Delete your existing repo if necessary

- Open GitHub
- Select your **sas** repo.
- Click on the settings icon on the right-hand side of the toolbar.
- Go all the way to the bottom to the **Danger Zone**
- Delete your repo.

Creating a new repo

- Open GitHub
- Click on the **New** button
- Name the repository **sas**
- Put the following in the description

`A small database project for school and practice using version control.`
* Click on **Create Repository** * Scroll to the section labeled **...or push an existing repository from the command line**. * Copy the first line (technically, you can do them all at once) and paste it in your terminal.

This is my information. Yours will be a little different.

```
git remote add origin https://github.com/charliekwallin/sas.git
```

Continue with the following lines

```
git branch -M main
git push -u origin main
```

Create the Database

The textbook introduced databases earlier in the term, but a little review is usually helpful. In addition to reviewing the chapter in your textbook on creating databases or watching the tutorials from LinkedIn/Learning, you should have enough information to make the database.

Here is a video on creating a database locally using XAMPP/MAMP. You have seen this before.

There is no need to watch it if you are comfortable with this process.

The salamander.sql code

- Read through the SQL code. It is not long, but it contains a lot of important information.
- Open PHPMyAdmin or the MySQL CLI, then copy, paste, and run the code.
- Check to make sure you created the database and the user.
- Use the username and password supplied in the *salamander.sql** file.

- Copy and paste the **db*connect*guide.php** file from chapter 06_02 and paste it in your **web182/sas** folder. The **db*connect*guide.php** file is useful and Kevin keeps coming back to it throughout the tutorials.

Fill the **db*connect*guide.php** file with the following information.

```php
<?php

// This guide demonstrates the five fundamental steps
// of database interaction using PHP.

// Credentials
$dbhost = 'localhost';
$dbuser = 'sally';
$dbpass = 'somePa55word';
$dbname = 'salamanders';

// 1. Create a database connection
$connection = mysqli_connect($dbhost, $dbuser, $dbpass, $dbname);

// 2. Perform database query

// 3. Use returned data (if any)

// 4. Release returned data

// 5. Close database connection
mysqli_close($connection);
```

Retrieve Data for Salamanders

You must modify the code so it works with the salamander database. Follow along with the video tutorial and start by changing **public/salamanders/index.php**.

Use this SQL statement that mimics the code the author has written. Even when this runs correctly, we will need to modify some other files soon.

```
$sql = "SELECT * FROM salamander ";
$sql .= "ORDER BY name ASC";
```

Right now, everything should work because we are still using the `salamander` array in the **salamanders/index.php** file.

I have added a couple of new fields for the salamander database. Here is the list of field names.

- id
- name
- habitat
- description

The important part to remember is that PHP receives data from MySQL in the form of an array.

When he creates the function `find_all_subjects()`, you need to change it for salamanders. Here is what the function looks like

```
function find_all_salamanders() {
    global $db;
    $sql = "SELECT * FROM salamander ";
    $sql .= "ORDER BY name ASC";
    $result = mysqli_query($db, $sql);
    return $result;
}
```

Work with Retrieved Data

Again, you are still working through the InLearning videos, but you are using them as a guide to create a salamander site instead of globe bank.

Edit the **index.php** page, so it returns salamander data.

Here is some code to get you started

```
<?php while($salamander = mysqli_fetch_assoc($salamander_set)) { ?>
      <tr>
        <td><?= h($salamander['id']); ?></td>


        Your Code Here


     <td><a href="<?= url_for('salamanders/delete.php?id=' . h(u($salamar
        </tr>
    <?php } ?>
  </table>
```

Stubs

Programmers use **stubs** as a placeholder for pages or files. Click on any of the links such as **Edit** or **Create** to see an example. You will need to create **stubs** for the following pages.

- edit.php
- show.php
- create.php
- delete.php

Error Handling

Error handling is straightforward in the video and it is easy to apply the content to the salamander code.

Git Push

At this point, your site should be working on your localhost. It is time to stage, commit, and push your code to your GitHub account. It is basically the same process as before, but at the end you push your code. Here are the steps.

```
git status
git add .
git commit -m "Completed asgn09. Code displays salamander data"
git push
```

Check your work

Head out to GitHub and see how your files look.

Webhost

It is time to prepare it for your Webhost.

Create the Database and User First

Watch this video on creating a database and user on your webhost you've seen before on created a database and user on your Webhost. Although I am using A2Hosting, the process is the same no matter which webhost you use.

Here are the steps.

1. Create a database manually by clicking on the Database link. Do not use PHPMyAdmin for this process. *It will not work*.
2. Create a user
3. Pair the user with the database.

IMPORTANT: Your username and password are different on your Webhost compared to your localhost. Make sure you modify your **db_credentials.php** file before uploading it to your Webhost.

Add the table structure and data using PHPMyAdmin on your Webhost

This is the modified SQL code that you can use on your Webhost. Notice that it is missing the CREATE DATABASE and GRANT PRIVILEGES commands. Those commands are handled by your Webhost when you create the database and user.

```
CREATE TABLE `salamander` (
  `id` int(11) NOT NULL,
  `name` varchar(255) NOT NULL,
  `habitat` text,
  `description` text
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `salamander` (`id`, `name`, `habitat`, `description` ) VALUES
(1, 'Southern Red-Backed', 'The southern zigzag salamanders only occurs in
(2, 'Southern Zig Zag', 'The southern zigzag salamanders only occurs in a s
(3, 'Pigeon Mountain Salamander', 'These three species are very common in t
(4, 'Slimy Salamander', 'Slimy salamanders are entirely terrestrial.  In No
(5, 'Green Salamander', 'Green salamanders typically inhabit moist, shady c

--
-- Indexes for table `salamander`
--
ALTER TABLE `salamander`
  ADD PRIMARY KEY (`id`);
```

Back to Git and GitHub

Once your code is working locally, you will need to commit your changes. I always start with `git status` to see where things are at.

IMPORTANT - username and password

You now have your database login credentials in your `db_credentials.php` folder. You don't want the public to see these.

Before committing and pushing your files to GitHub, you should remove the **username** and **password**.

Store them temporarily and replace them once you have pushed your code to your GitHub repo.

Here are the commands.

```
git status
git add .
git commit -m"Phase03 is complete. The program displays all of the salamand
```

Upload your code to your Webhost

Use your Webhost's File Manager to upload files instead of SFPT if you had difficulty getting SSH to work. You can find the File Manager in your CPanel. Most Webhost file managers only allow you to upload a file. If you need to upload folders, you will need to

1. Zip the folder
2. Upload the zipped folder
3. Extract the zipped folder using the File Manager

Check your work.

Submit

- Put your GitHub address for this assignment in the Comments section of Moodle. You will **not** submit any code in Moodle.
- Your Website's address. It points directly to this assignment in the Comments section of Moodle. It will look something like. https://web182.charliewallin.com/sas/public/salamanders/