

INFO2180 - Lab 4 (20 marks)

Due Date: Sunday, November 17, 2024

For this lab you will be required to work through an AJAX example. You will need to ensure that you have PHP installed. You can install XAMPP or MAMP which will come with Apache, MySQL and PHP installed together. Links to download either can be found below:

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

MAMP: <https://www.mamp.info/>

Here is a short video on how to install XAMPP on Windows: <https://youtu.be/FiFJ1jTfITc>

You will be required to submit the URL of your Github repository by the deadline specified above.

Create your Repository

You are also required to create a Git repository on Github called **info2180-lab4** and after each exercise commit your code and push to Github. So let us start by creating that repository.

1. You can create a new Github repository by going to <https://github.com/new> or by clicking the "+" icon in the top right once you are logged into the website and select "New Repository".
2. For this lab, use the repository name as **info2180-lab4**.
3. Ensure that your repository is *Public* and that you select the option to initialize the project with a README file.

4. The remaining options can be left at their defaults. Then click on **Create Repository**.

The screenshot shows the 'Create a new repository' page on GitHub. At the top, there's a navigation bar with links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below this, the page title is 'Create a new repository'. A sub-header explains that a repository contains all project files, including revision history, and offers a link to 'Import a repository' if one already exists elsewhere. The form includes fields for 'Owner' (set to 'uwi-info2180') and 'Repository name' (set to 'info2180-lab1'). A note suggests great repository names are short and memorable, with an example 'fuzzy-octo-palm-tree?'. There's an optional 'Description' field. Under the 'Visibility' section, 'Public' is selected, with a note that anyone on the internet can see the repository. 'Private' is also an option. The 'Initialize this repository with:' section has three checkboxes: 'Add a README file' (selected), 'Add .gitignore', and 'Choose a license'. Each has a brief description and a 'Learn more' link. A note states that this will set 'master' as the default branch. At the bottom, there is a green 'Create repository' button.

5. On your newly created repository, click on the "Code" button and copy the URL in the box that appears.

The screenshot shows the GitHub repository page for 'info2180-lab1'. The top navigation bar includes links for '<> Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. Below this, the repository name 'info2180-lab1' is displayed, along with 'ylynfatt Initial commit', '1 branch', and '0 tags'. A 'Code' button is highlighted, and a dropdown menu is open showing options: 'Clone' (with sub-options for HTTPS, SSH, and GitHub CLI), 'Open with GitHub Desktop', and 'Download ZIP'. The 'Clone' option is selected, and the URL 'https://github.com/uwi-info2180/info2180-lab1' is shown. The main content area shows the 'README.md' file with the text 'info2180-lab1'. On the right side, there are sections for 'About', 'Releases', and 'Packages', each with a brief description and a link to create a new release or package.

6. Open your command prompt, navigate to the **htdocs** or **www** folder for your installation of XAMPP, MAMP or WAMP (e.g. using the cd command) where you would like to clone your repository and using the URL you copied in step 5, type the following:

```
$ git clone https://github.com/<your-username>/info2180-lab4.git
```

NOTE: Ensure you change the URL and use the one that you copied in Step 5 with your username.

7. Then navigate to your newly cloned repository by typing:

```
$ cd info2180-lab4
```

8. You can then type the command ls and you should see a README.md file.
9. In the file add:

```
# INFO2180 Lab 4
```

```
This is Lab 4 for <Your Name>
```

Of course, change <Your Name> to your actual name.

10. Save the file and then at the command line (or Windows Powershell), type:

```
$ git add README.md  
$ git status
```

You should then see that the file has been staged.

11. Commit these changes to your local Git repository:

```
$ git commit -m "Edited README.md"
```

12. Check the status of your Git repository:

```
$ git status
```

It should now mention that there is nothing new to commit and that the working tree is clean.

13. Great! Now we should push this information to GitHub.

```
$ git push
```

It may ask you to enter your Github username and password. Enter it and then press enter. If all goes well you can then view your Github repository on the Github website and you should see that the file exists on Github with the changes you made to the file.

14. To see the history of the commits you can run the following command:

```
$ git log
```

15. Next, Create a new branch called **ajax-implementation**.

```
$ git checkout -b ajax-implementation
```

16. Now download the starter files (if you have not already done so) for the lab at the following URL:

<https://github.com/uwi-info2180/info2180-ajax-superheroes/archive/master.zip>

17. Unzip and copy the starter files (superheroes.php) from the link above into your **info2180-lab4** folder that you cloned in Step 6 and do an initial commit.

```
$ git add .
```

```
$ git commit -m "Added initial starter files"
```

18. Now begin the next set of exercises.

Exercise 1 - Creating a PHP based AJAX Endpoint that prints a list of Superheroes

If you haven't already done so, ensure that you get the **superheroes.php** file from the following repository <https://github.com/uwi-info2180/info2180-ajax-superheroes/archive/master.zip>.

Once you have the file, save it within your **info2180-lab4** folder from the repository that you cloned in Step 1. Remember this needs to be placed in the **htdocs** or **www** directory for your installation of XAMPP, MAMP or WAMP.

For XAMPP or WAMP, you will open your web browser and visit:

<http://localhost/info2180-lab4/superheroes.php>

For MAMP, you might open your web browser and visit:

<http://localhost:8888/info2180-lab4/superheroes.php>

For those of you who have PHP installed separately and have it available on the command line, you can start the built-in PHP development server by running the following command from within your info2180-lab4 folder:

```
$ php -S localhost:8080
```

And in that case open your web browser and visit <http://localhost:8080/superheroes.php>.

If everything was setup correctly you should see a page that looks like Figure 1 below:

- Captain America
- Ironman
- Spiderman
- Captain Marvel
- Black Widow
- Hulk
- Hawkeye
- Black Panther
- Thor
- Scarlett Witch

FIGURE 1: LIST OF SUPERHEROES

Exercise 2 - Make an AJAX call that returns the list of Superheroes

Create a HTML page called `index.html`, a CSS file called `styles.css` and a JavaScript file called **app.js**. Now write some JavaScript code so that when a user clicks a “**Search**” button, the list of Superheroes appears as a JavaScript alert.

Hint: Listen for a “**click**” event on the button. Fetch the data by opening an AJAX request which returns the result of `superheroes.php`.

Hint 2: For your AJAX request you may use the XMLHttpRequest object, jQuery `ajax()` method or the `fetch()` API.

You should see something that looks like Figure 2 below:

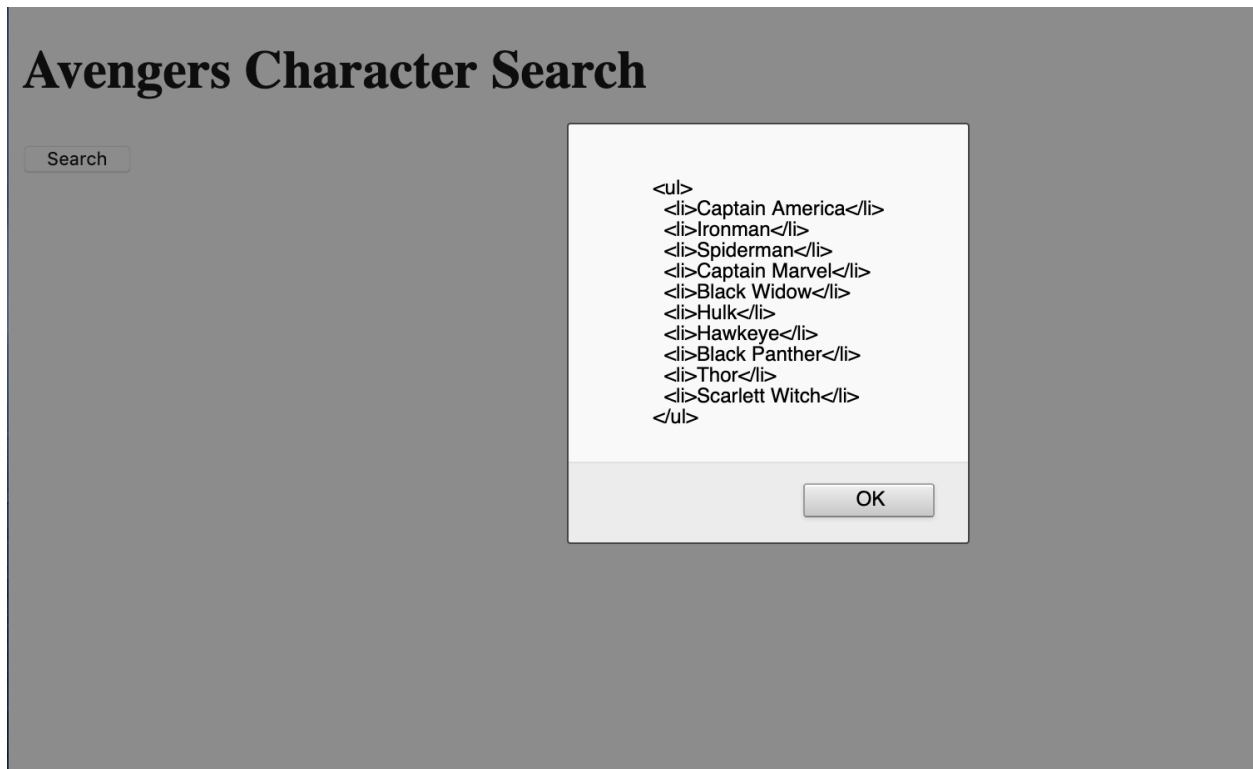


FIGURE 2: LIST OF SUPERHEROES IN A JAVASCRIPT ALERT WHEN SEARCH BUTTON CLICKED

NOTE: You **MUST** make a commit after doing this exercise and push to Github.

Exercise 3 - Look up any Superhero by typing in the name or alias in a Textfield

Modify your code so that a user can either search for a specific superhero by typing the **name or alias** into a text field and retrieve only that superhero (See Figure 4) or leave it blank in which case it will return the original list of superheroes (See Figure 3).

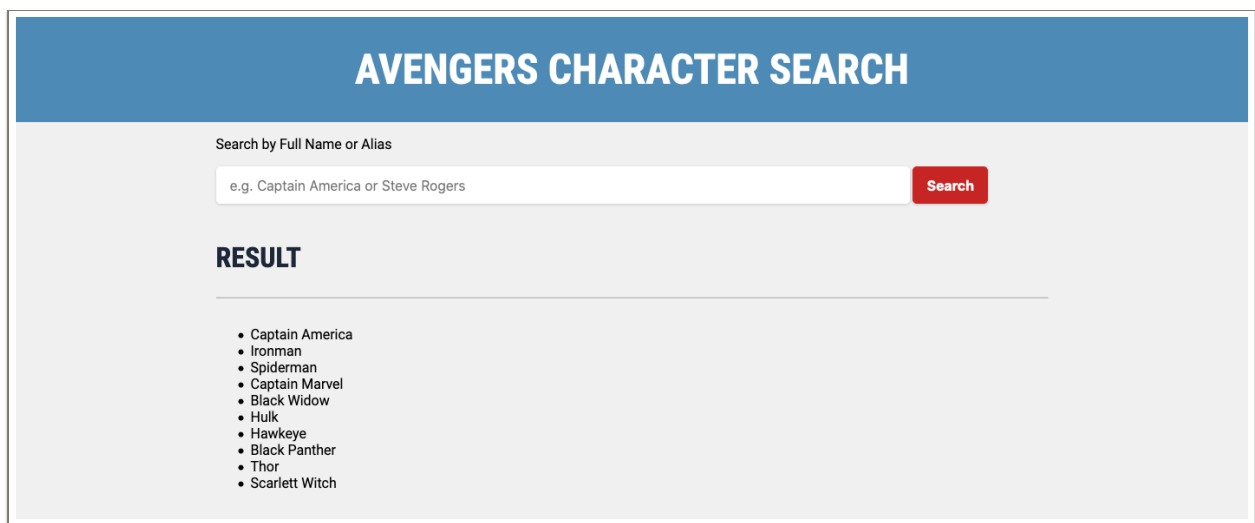
- Show the result inside a div on your page with id of “**result**”, rather than an alert.
- Ensure you are still using the **superheroes.php** that was created earlier. **NOTE:** *You are not allowed to change the \$superheroes array.*
- Ensure you sanitize the user input from the search text field so that someone can't inject any malicious code into your web page.
- The search query should be passed as a query param in the query string for your AJAX request to superheroes.php. For example, **superheroes.php?query=Ironman**

- When returning a single superhero the Superhero's Alias should be in an **<h3>** tag, the Superhero's Name in an **<h4>** tag and the Superhero's biography in a **<p>** tag.
- If a specific superhero cannot be found then you should display a message stating **"Superhero not found"** in the result div. See Figure 5.

Hint: You can use the JavaScript innerHTML property (plain JavaScript) or the jQuery html() method if you are using jQuery.

Hint 2: You may find some of PHP's handy builtin array methods helpful in searching through the array of superheroes for the one matching what the user typed into the search field.

The interface will look something like this (obviously you can call it whatever you would like and style it however you would like), but it **MUST** have the form field, button and display the output.



The screenshot shows a web interface titled "AVENGERS CHARACTER SEARCH" in a blue header. Below the header is a search bar with the placeholder text "Search by Full Name or Alias" and "e.g. Captain America or Steve Rogers". A red "Search" button is to the right of the search bar. Below the search bar is a section titled "RESULT" with a horizontal line underneath. Under the line is a bulleted list of superhero names: Captain America, Ironman, Spiderman, Captain Marvel, Black Widow, Hulk, Hawkeye, Black Panther, Thor, and Scarlett Witch.

FIGURE 3: RESULT WHEN NOTHING IS ENTERED IN THE SEARCH FIELD AND SEARCH IS CLICKED

The screenshot shows a web interface titled "AVENGERS CHARACTER SEARCH". Below the title is a search bar with the placeholder text "Search by Full Name or Alias". The search bar contains the text "Captain America". To the right of the search bar is a red button labeled "Search". Below the search bar, the word "RESULT" is displayed in bold. Underneath, the character name "CAPTAIN AMERICA" is shown in bold, followed by "A.K.A STEVE ROGERS". A short description follows: "Recipient of the Super-Soldier serum, World War II hero Steve Rogers fights for American ideals as one of the world's mightiest heroes and the leader of the Avengers."

FIGURE 4: RESULT WHEN YOU SEARCH FOR A SPECIFIC SUPERHERO.

The screenshot shows the same "AVENGERS CHARACTER SEARCH" interface. The search bar now contains the text "Batman". The "Search" button is still present. Below the search bar, the word "RESULT" is displayed in bold. Underneath, the text "SUPERHERO NOT FOUND" is shown in red, indicating that no character matches the search criteria.

FIGURE 5: ERROR MESSAGE WHEN NO SUPERHERO MATCHES THE SEARCH

NOTE: You **MUST** make a commit after doing this exercise and push to Github.

When you are done and are sure everything is working, switch back to your master (or main) branch and then merge the changes you made in your "ajax-implementation" branch back into your master (or main) branch. Then ensure you push those changes back to your Github repository.

Submission

Submit your information via the “**Lab 4 Submission**” link on OurVLE, with your Github repository URL (e.g. <https://github.com/yourusername/info2180-lab4>)