HTTP Forms Lab

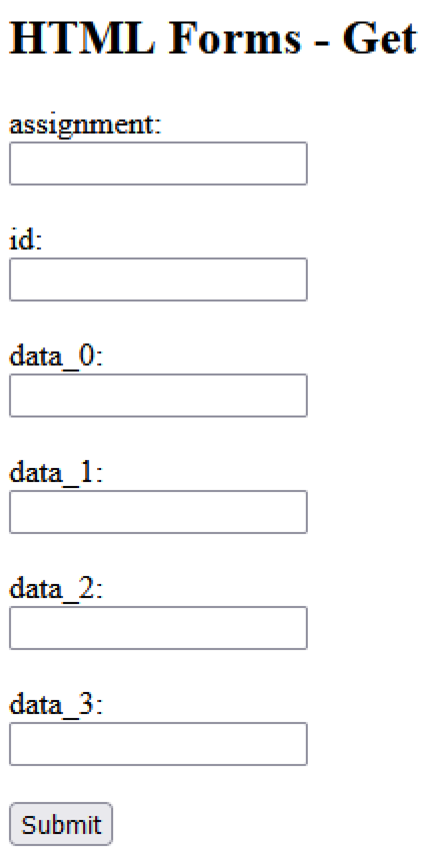
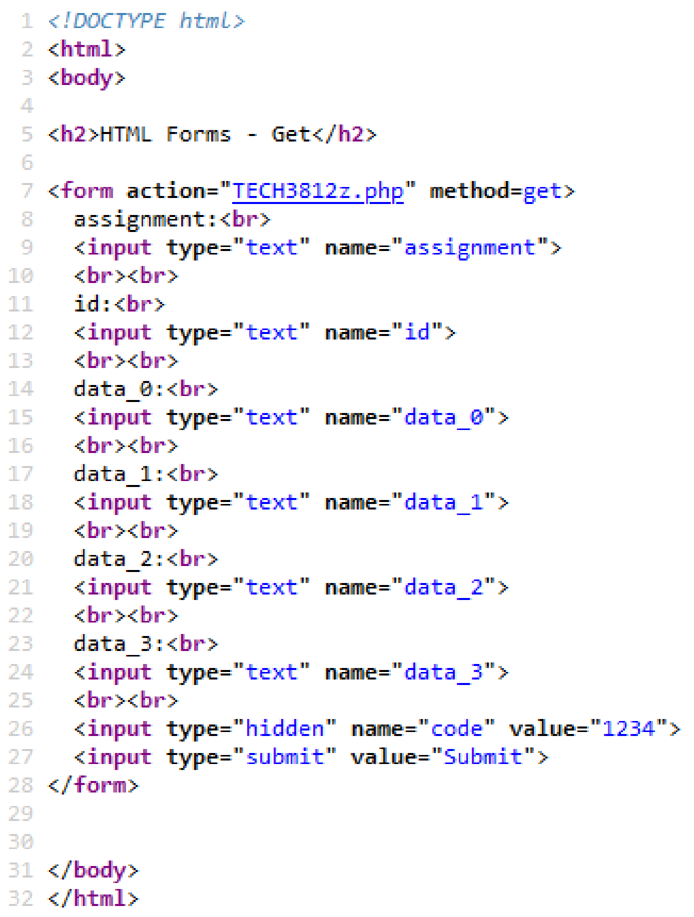
NOTE: TAKE NOTES DURING THIS LAB…this information is CRITICAL for TECH 4234 in the fall and comes up multiple times. Much of the IoT is based on the concepts introduced in this lab.

Web servers have the ability to accept user input via FORMS. Forms are webpages with special html tags that allow the user the pass information to the server. This form data is usually sent to a script that processes the information. The script can use the information (eg calculations), store the information to a file, enter the information into a database, or use the information in a multitude of other ways.

In IoT devices, many times we wish to send information from remote sensors to a server for computation and storage.

HTML forms can pass this information in two ways: GET and POST.

Lets take a look at GET webpage.

Website (as seen in broswer)	**HTML Code that generates the page shown**
	

HTML Forms - Get

assignment:

id:

data_0:

data_1:

data_2:

data_3:

```html
<!DOCTYPE html>
<html>
<body>

<h2>HTML Forms - Get</h2>

<form action="TECH3812z.php" method=get>
  assignment:<br>
  <input type="text" name="assignment">
  <br><br>
  id:<br>
  <input type="text" name="id">
  <br><br>
  data_0:<br>
  <input type="text" name="data_0">
  <br><br>
  data_1:<br>
  <input type="text" name="data_1">
  <br><br>
  data_2:<br>
  <input type="text" name="data_2">
  <br><br>
  data_3:<br>
  <input type="text" name="data_3">
  <br><br>
  <input type="hidden" name="code" value="1234">
  <input type="submit" value="Submit">
</form>


</body>
</html>
```

<h2></h2> creating a heading

<form></form> surround the actual form. The "action=" statement tells the form where to send the form information to for processing, in this case "TECH3812z.php" which is the script that processes the information and displays the information (although, as stated previously, we can do anything with the values sent to the server). The "method=" tells the browser to use the "GET Method" to send the data to the server.

<input> lets you define the form, here we use 'type="text"' for most of the form. The "name=" gives a name (sort of like a variable name) to the information. So he were have 6 pieces of information.

'type="hidden" is a way to pass information not entered by the user…the 'value="1234"' passes that value to the server with the name "code"

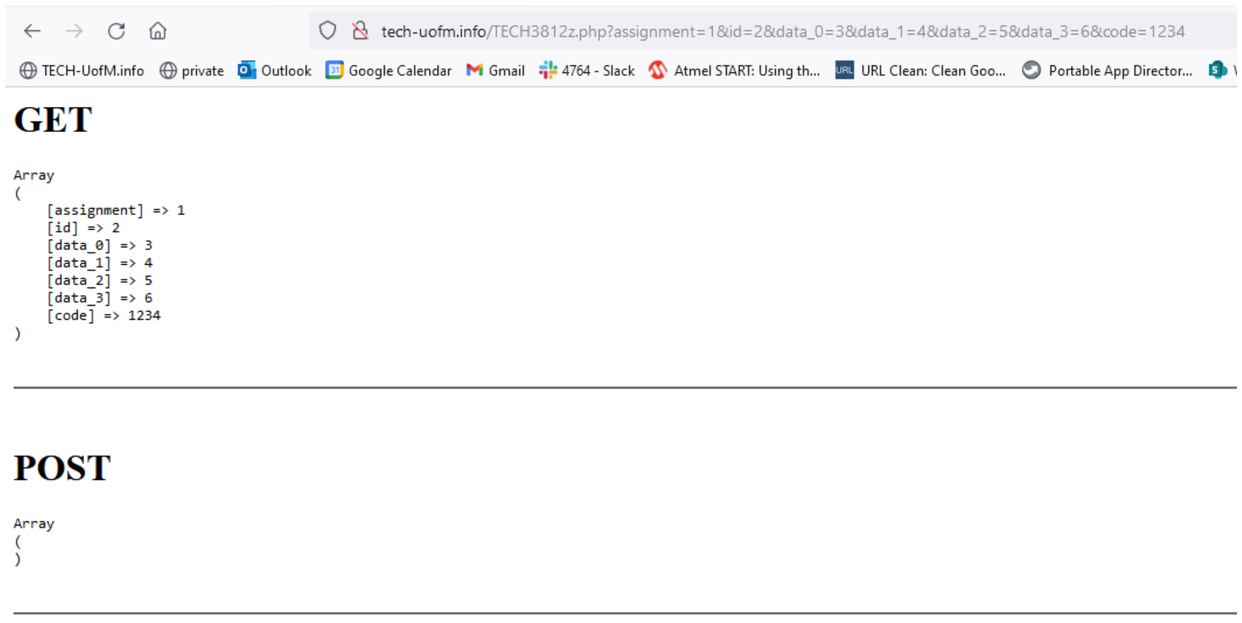
'type="submit"' is used for the submit button

(note there are many types of form inputs besides these…see https://www.w3schools.com/tags/tag_input.asp for a full list)

Open wireshark and start a capture.

Open the page http://tech-uofm.info/TECH3812f.html and fill out distinct values into each box and hit submit. Stop the capture.

On the browser you should see something like this (but with your entries after the => in the GET section of the page:



GET

```
Array
(
    [assignment] => 1
    [id] => 2
    [data_0] => 3
    [data_1] => 4
    [data_2] => 5
    [data_3] => 6
    [code] => 1234
)
```

POST

```
Array
(
)
```

More importantly you should note the URL (in the example above it reads:

http://tech-uofm.info/TECH3812z.php?assignment=1&id=2&data_0=3&data_1=4&data_2=5&data_3=6&code=1234

A few things to note:

- The "TECH3812z.php" matches the action of the form
- ? starts the data put into the form
- then the name of each input box, followed by =, followed by the user input
- & separates data from multiple boxes

Now take a look at the wireshark packets captured during this exchange (use http and possibly the ip.addr of your pc and the server to help filter out all but the packets in this exchange). You should have the form being requested, the form being retrieved, the data being sent to the server (request for TECH3812z.php) and then an OK displaying the values in the GET Section of the page.

Using what you learned from the above and from the HTML Minilab yesterday, use putty to try to send new data to the TECH3812z.php form processor using GET. Once you have successfully completed this, demonstrate to the instructor.

POST, is similar to GET, except the information being sent to the server is NOT included in the URL, but is embedded within the exchange between the client (web browser) and the server.

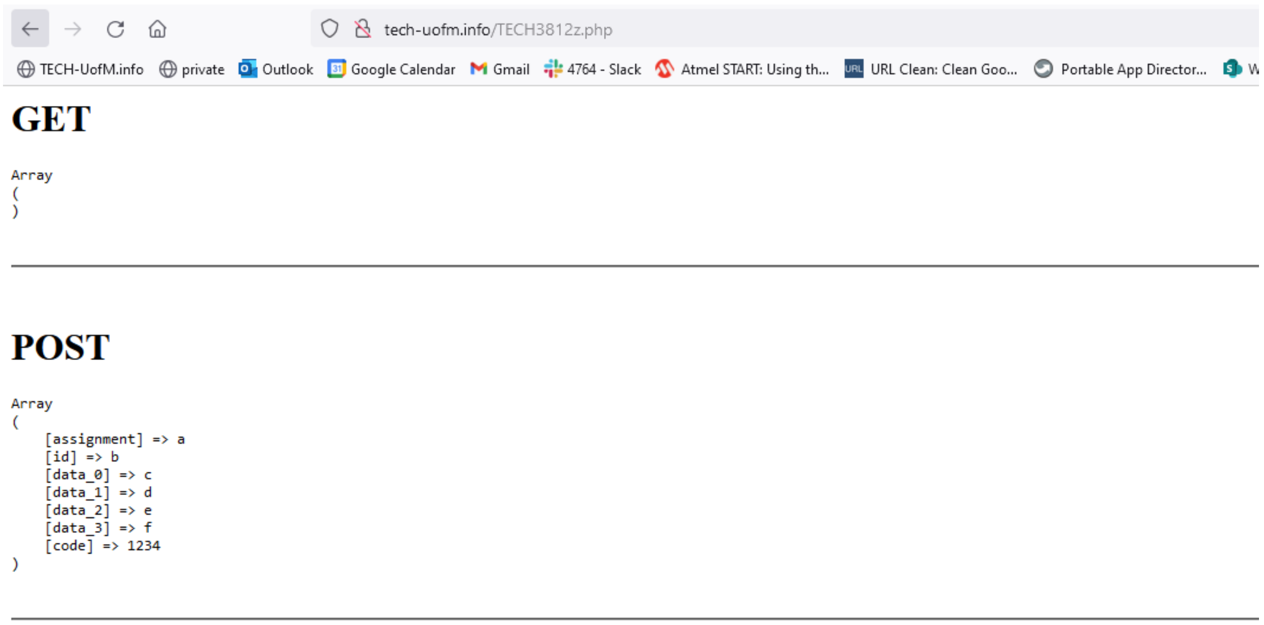
The only thing changed in the form is using "method=POST" instead of "GET".

Let's see how this works via wireshark.

Open wireshark and start a new capture.

Open the page http://tech-uofm.info/TECH3812g.html and fill out distinct values into each box and hit submit. Stop the capture.

On the browser you should see something like this (but with your entries after the => in the POST section of the page:



GET

```
Array
(
)
```

POST

```
Array
(
    [assignment] => a
    [id] => b
    [data_0] => c
    [data_1] => d
    [data_2] => e
    [data_3] => f
    [code] => 1234
)
```

A few things to note:

- We used the same "TECH3812z.php" matches the action of the form
- The URL does NOT contain any of the information (unlike get)

Look at the packets captured for this exchange. You should have a packet sending the request for the form TECH3812g.html, the server sending back the form, a packet sending the form information and then an OK displaying the values in the POST Section of the page.

Look carefully at the packet sending the form information. What is different about the info (in the top window)?

Dive down and look at the "Hypertext Transfer Protocol" section of the packet as we did in yesterday's mini lab.

Using what you learned from the above and from the HTML Minilab yesterday, use putty to try to send new data to the TECH3812z.php form processor using POST.

A few major hints:

- You will need the 2 similar lines to GET info you sent previously
- You will need to tell the server the content-type is a form (look in packet)
- You will need to say how many bytes are in the data being transmitted
- **There must be a blank line before the data!**
- You will need the data itself (HTML FORM Data)
    - Wireshark shows this last line broken up (making it hard to copy), although it is shown in the ascii dump in the bottom window, that is very hard to copy as well.
    - To get the data format, without a lot of editing….click on the "HTML FORM URL ENCODED" line in the center window. Then save the data via FILE | Export Packet Bytes | and save the file to a known location.
    - You can open and modify the data in notepad or notepad++ and then highlight the string to get the byte count which is displayed at the bottom of the window in both programs (although it seems like notepad is off by one, so subtract one from that number).

Once you have successfully completed this, demonstrate to the instructor.