VANIER COLLEGE - Computer Engineering Technology
247-615 Network Troubleshooting

Lab 3 : Introduction to HTML and PHP

Leonardo Fusser (1946995)

Purpose:

a) Use HTML POST methods.
b) Add inputs to HTML page.
c) Learn basic PHP server scripting language.

To be submitted :

1. Deadline : as stated in MS Teams Assignment.
2. **No formal report is required.** The followings are required as submission.
   a. Properly indented and commented final code, screen shots (clearly labelled).
   b. Include a final discussion and conclusion session.

Lab Work:

Part A : PHP installation

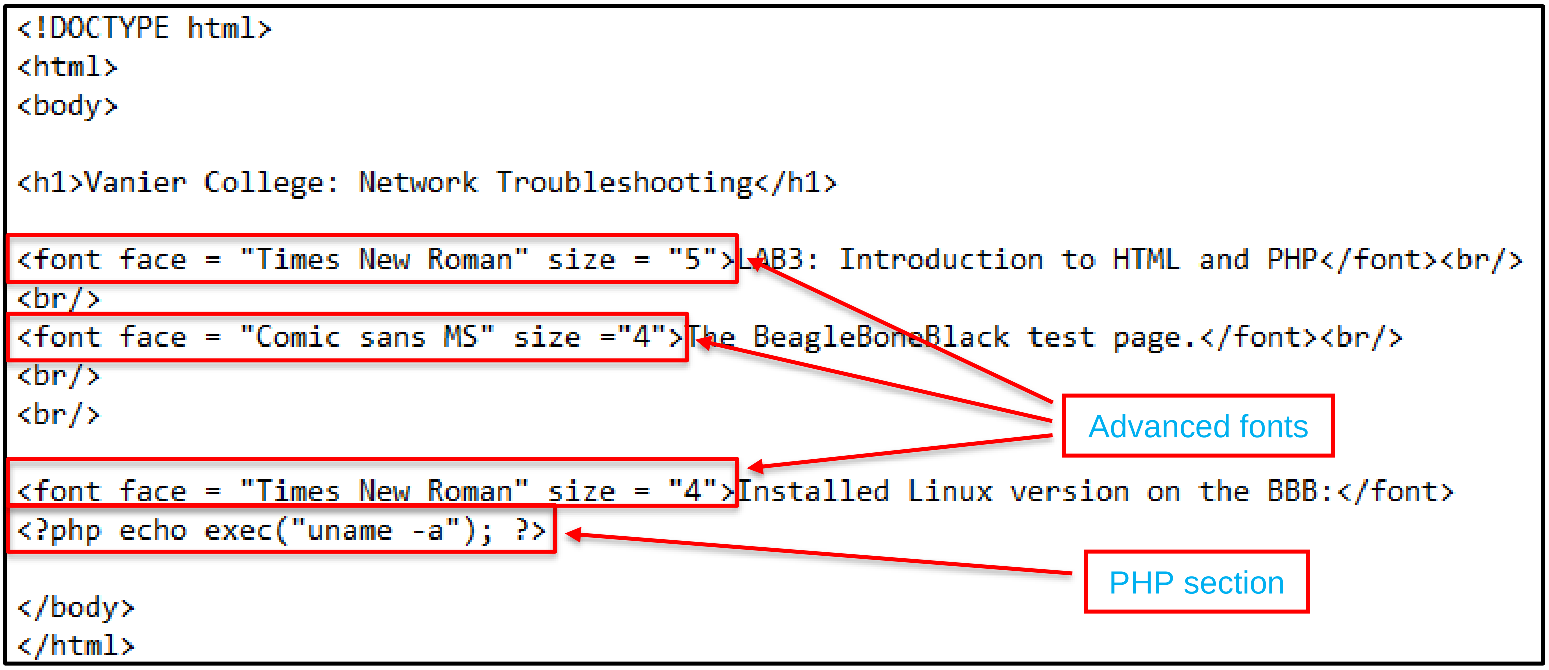
1. Connect BBB to internet and install PHP5 using the following commands. It might take a few minutes to install.

   ```
   #apt-get update
   #apt-get install php5-common libapache2-mod-php5 php5-cli
   ```

2. Make a copy of your html file from previous lab, and name it `lab3a.php`. Update the title of the lab in the webpage accordingly.

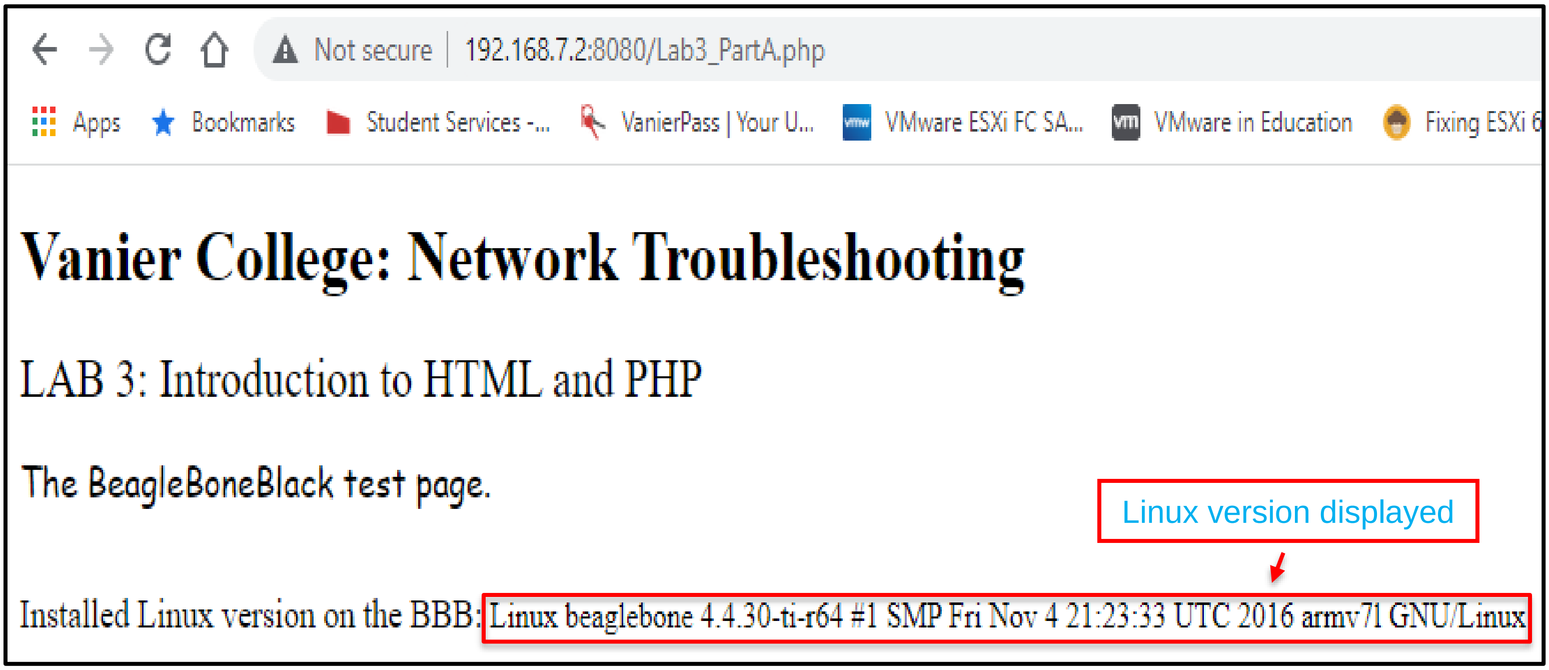
3. Add a PHP section to print the Linux version on the web page (hint: use `uname` command). Perform a screen shot. Attach screen shot of code changes in your report.

   - Refer to the next page for screenshots.



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

<h1>Vanier College: Network Troubleshooting</h1>

<font face = "Times New Roman" size = "5">LAB3: Introduction to HTML and PHP</font><br/>
<br/>
<font face = "Comic sans MS" size ="4">The BeagleBoneBlack test page.</font><br/>
<br/>
<br/>

<font face = "Times New Roman" size = "4">Installed Linux version on the BBB:</font>
<?php echo exec("uname -a"); ?>

</body>
</html>
```

Figure 1. Updated HTML code with more advanced fonts and that has a PHP section to display the installed Linux version running on the BBB.



Vanier College: Network Troubleshooting

LAB 3: Introduction to HTML and PHP

The BeagleBoneBlack test page.

Installed Linux version on the BBB: Linux beaglebone 4.4.30-ti-r64 #1 SMP Fri Nov 4 21:23:33 UTC 2016 armv7l GNU/Linux

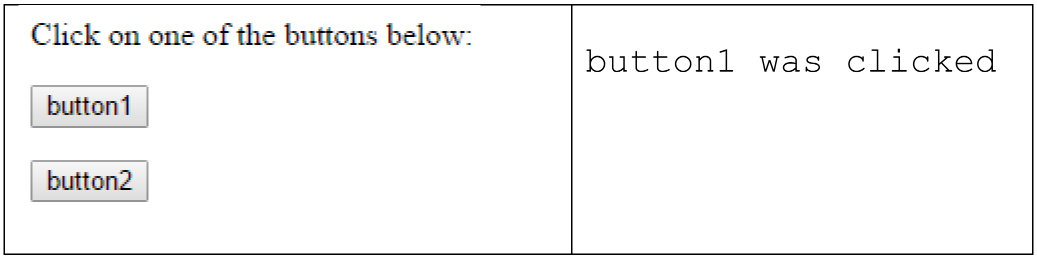
Figure 2. Result of running updated HTML code that includes more advanced fonts and displays the currently installed version of Linux on the BBB.

Part B : Basic PHP scripting

4. Create an html file to submit a form (`lab3b.html`) using GETs method. Also create a PHP file to process the data and print the results (`lab3b.php`) in new webpage, as shown below. Perform necessary screen shot to illustrate your results, including the change in your URL that display your selection.

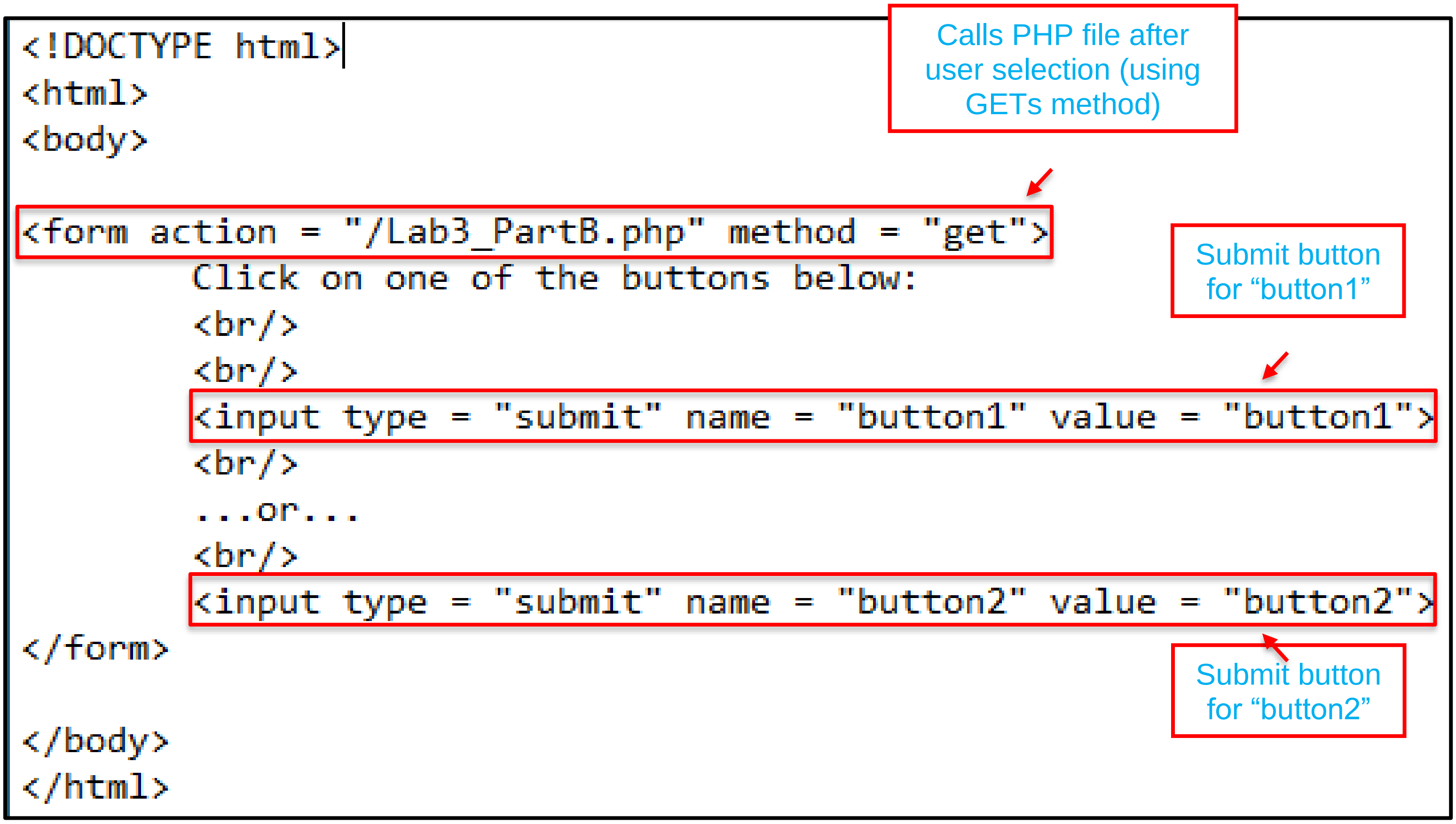
   Attach screen shot of code changes in your report.

Webpage: Response:



Click on one of the buttons below:

```
button1 was clicked
```



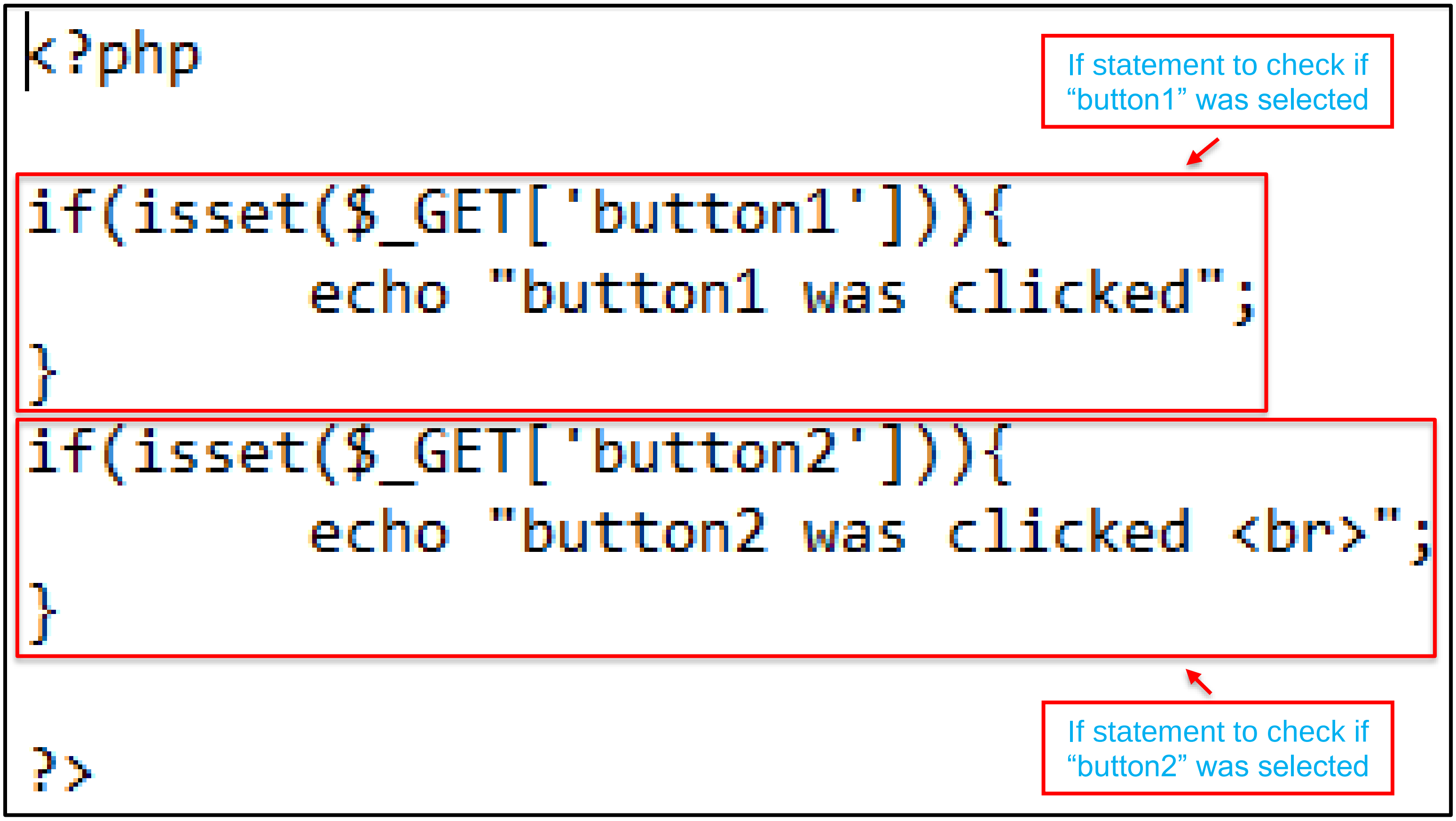
```
<!DOCTYPE html>
<html>
<body>

<form action = "/Lab3_PartB.php" method = "get">
        Click on one of the buttons below:
        <br/>
        <br/>
        <input type = "submit" name = "button1" value = "button1">
        <br/>
        ...or...
        <br/>
        <input type = "submit" name = "button2" value = "button2">
</form>

</body>
</html>
```

Calls PHP file after user selection (using GETs method)

Submit button for “button1”

Submit button for “button2”

Figure 3. HTML file code to handle submission form/getting user data using GETs method.

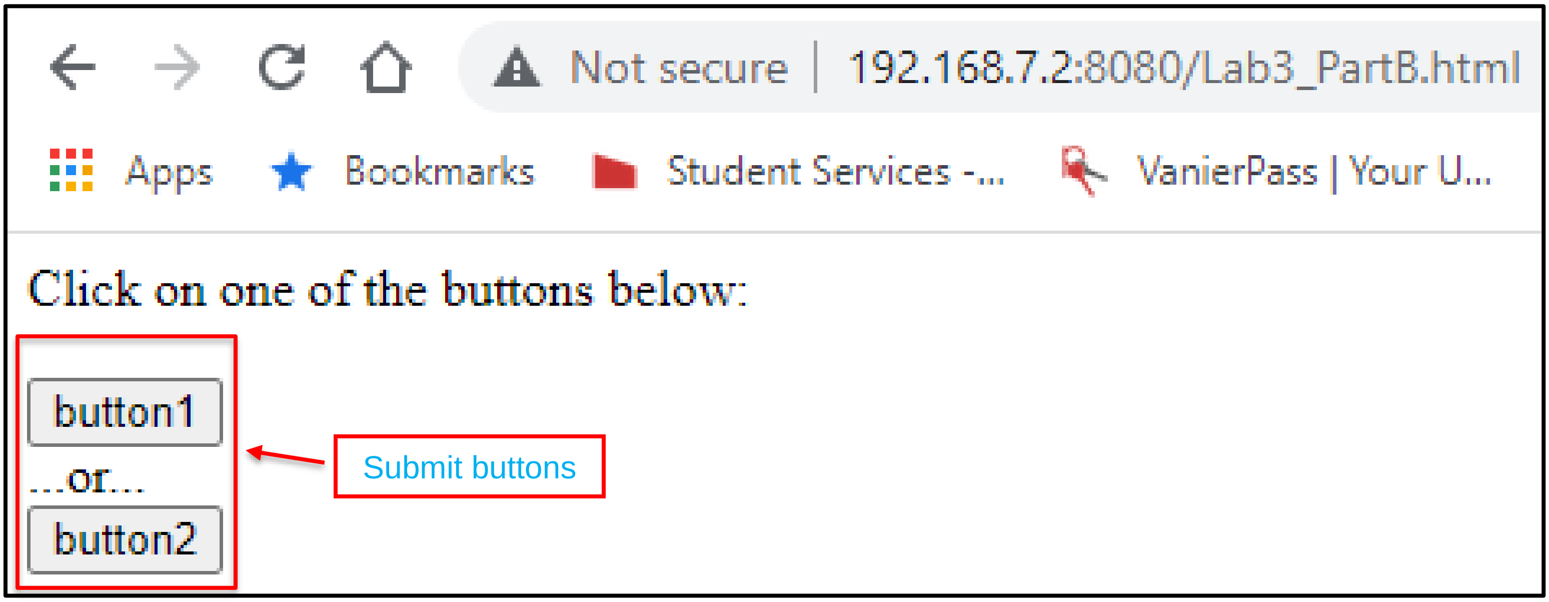


```
<?php

if(isset($_GET['button1'])){
        echo "button1 was clicked";
}
if(isset($_GET['button2'])){
        echo "button2 was clicked <br>";
}

?>
```

If statement to check if “button1” was selected

If statement to check if “button2” was selected

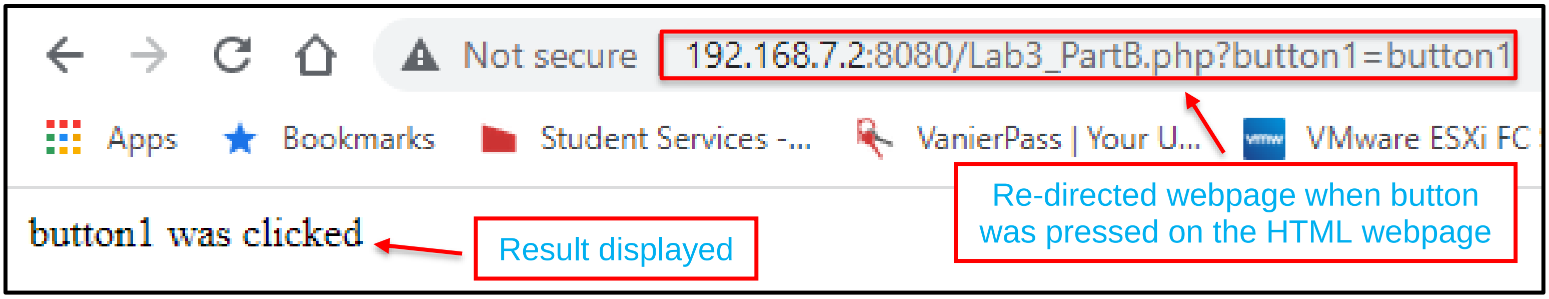
Figure 4. PHP file code to handle and print received data from HTML file.



Click on one of the buttons below:

Submit buttons

Figure 5. Result of running HTML code that handles user submission/getting user data before it is sent to the PHP code for processing and printing.



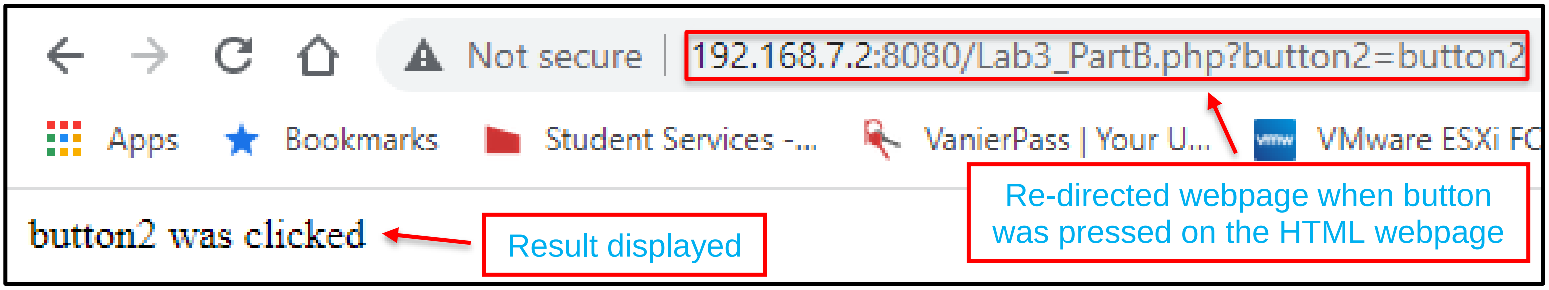
192.168.7.2:8080/Lab3_PartB.php?button1=button1

button1 was clicked

Result displayed

Re-directed webpage when button was pressed on the HTML webpage

Figure 6. Result of running PHP code that handles data processing and printing. Screenshot above shows what happens when the button "button1" was clicked on the HTML webpage.



192.168.7.2:8080/Lab3_PartB.php?button2=button2

button2 was clicked

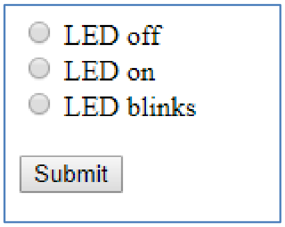
Result displayed

Re-directed webpage when button was pressed on the HTML webpage

Figure 7. Result of running PHP code that handles data processing and printing. Screenshot above shows what happens when the button "button2" was clicked on the HTML webpage.

Part C : PHP with radio buttons

5. Create a new php file `lab3c.php` (with PHP code embedded in HTML) to provide control on USR3 LED using **POST** method.
6. Example of the basic interface is shown below. You are required to display a text to show the real time status of the USR3 LED too.



Notes:
To control I/Os from PHP, full permission must be granted :

```
#chmod 777 /sys/class/leds/beaglebone:green:usr3/brightness
```

To run an executable from PHP both the directory AND the file must be granted full permissions.

Example:
To run `/home/exec_file` from PHP, you must grant permission to both `/home` and `exec_file`

```
#chmod 777 /home
#chmode 777 /home exec_file
```

7. Perform screen shots to illustrate various results of your webpage. Attached screen shot of source code in your submission.

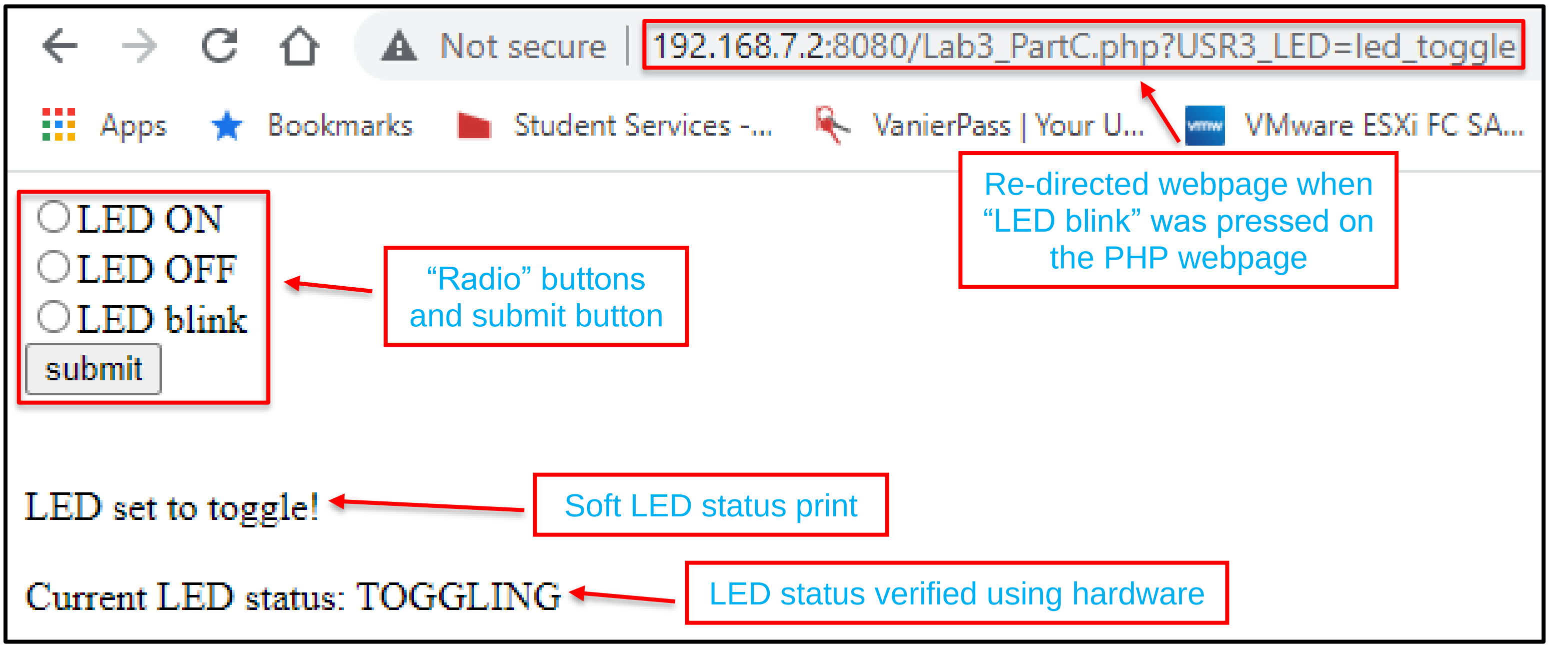
   - ➢ Refer to next few pages for screenshots.

8. Demo your final output from all the 2 parts to instructor for approval.

[Lab3_PartC.php code]

```
<!-- Beginning of HTML section & Beginning of file -->
<!DOCTYPE html>
<html>
<body>

<!-- Sends user selection using GETs method: sends to PHP section -->
<form action = "/Lab3_PartC.php" method = "get">
<input type = "radio" name = "USR3_LED" value = "led_on">LED ON<br>
<input type = "radio" name = "USR3_LED" value = "led_off">LED OFF<br>
<input type = "radio" name = "USR3_LED" value = "led_toggle">LED blink<br>
<input type = "submit" value = "submit">
</form>

<!-- Beginning of PHP section -->
<?php
<!-- Retrieves user selection using GETs method: retrieves from HTML section -->
$rx = $_GET['USR3_LED'];

<!-- Turns LED ON -->
if ($rx == "led_on"){
       exec("echo none > /sys/class/leds/beaglebone:green:usr3/trigger");
       exec("echo 1 > /sys/class/leds/beaglebone:green:usr3/brightness");
       echo "<br>";
       echo "<br>";
       <!-- Soft print for LED status -->
       echo "LED set to turn on!";
}

<!-- Turns LED OFF -->
elseif ($rx == "led_off"){
       exec("echo none > /sys/class/leds/beaglebone:green:usr3/trigger");
       exec("echo 0 > /sys/class/leds/beaglebone:green:usr3/brightness");
       echo "<br>";
       echo "<br>";
       <!-- Soft print for LED status -->
       echo "LED set to turn off!";
}
```

```
<!-- Makes LED blink -->
elseif ($rx == "led_toggle"){
        exec("echo heartbeat > /sys/class/leds/beaglebone:green:usr3/trigger");
        echo "<br>";
        echo "<br>";
        <!-- Soft print for LED status -->
        echo "LED set to toggle!";
}

echo "<br>";
echo "<br>";
<!-- Makes webpage refresh every 2 seconds -->
header("refresh: 2");

<!-- Reads the status of the LED via hardware -->
$LED_PowerState = exec("cat /sys/class/leds/beaglebone:green:usr3/brightness");
$LED_ToggleState = exec("cat /sys/class/leds/beaglebone:green:usr3/trigger");

<!-- Prints message saying LED is blinking if LED is blinking -->
if ($LED_ToggleState == "none rc-feedback kbd-scrollock kbd-numlock kbd-capslock kbd-kanalock kbd-shiftlock kbd-altgrlock kbd-ctrllock kbd-altlock kbd-shiftllock kbd-shiftrlock kbd-ctrlllock kbd-ctrlrlock nand-disk usb-gadget usb-host mmc0 mmc1 timer oneshot [heartbeat] backlight gpio default-on"){
        echo "Current LED status: TOGGLING";
}

<!-- Prints message saying LED is ON if LED is ON -->
elseif ($LED_PowerState == "1"){
        echo "Current LED status: ON";
}

<!-- Prints message saying LED is OFF if LED is OFF -->
elseif ($LED_PowerState == "0"){
        echo "Current LED status: OFF";
}
?>
<!-- End of PHP section -->

</body>
</html>
<!-- End of HTML section & End of file -->
```



192.168.7.2:8080/Lab3_PartC.php?USR3_LED=led_toggle

○ LED ON
○ LED OFF
○ LED blink
submit

LED set to toggle!

Current LED status: TOGGLING

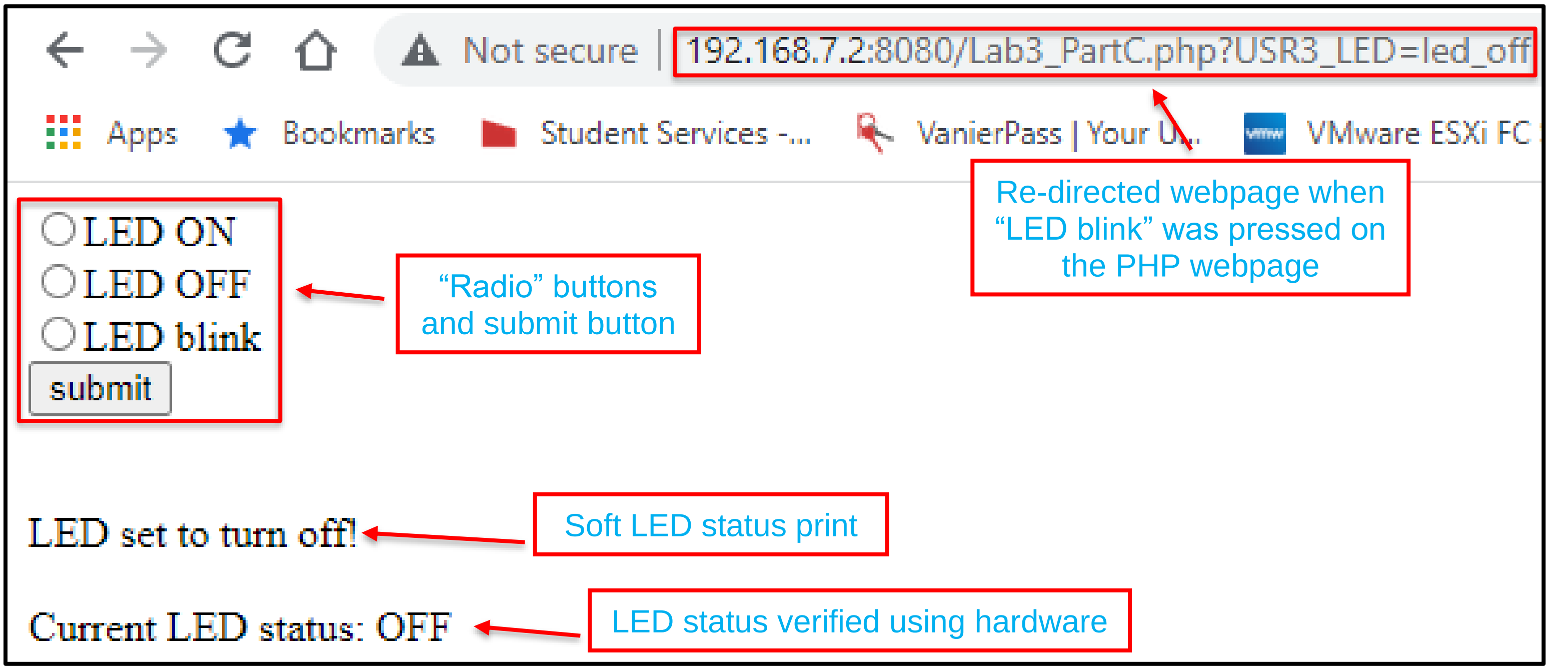
Re-directed webpage when "LED blink" was pressed on the PHP webpage

"Radio" buttons and submit button

Soft LED status print

LED status verified using hardware

Figure 8. Result of running PHP code from BBB when the user asks to toggle the LED.



192.168.7.2:8080/Lab3_PartC.php?USR3_LED=led_off

○ LED ON
○ LED OFF
○ LED blink
submit

LED set to turn off!

Current LED status: OFF

Re-directed webpage when "LED blink" was pressed on the PHP webpage

"Radio" buttons and submit button

Soft LED status print

LED status verified using hardware

Figure 9. Result of running PHP code from BBB when the user asks to turn the LED off.

Discussion:

- For the first part of the lab, PHP server was installed on the BBB, since it did not come with it, unlike from the previous lab with the Apache web server. Once PHP server was installed, a new HTML file was created to utilize the new feature of PHP. This new HTML file used more advanced fonts for appearance and used PHP to retrieve the installed version of Linux on the BBB (displayed to the user). This was tested via the BBB's Apache web server by entering the IP address of the BBB in a web browser and selecting the corresponding HTML file to open and test.

  For the second part of the lab, PHP was utilized even further. This time, there were two new files created: a HTML file and a PHP file. The goal was to display from the HTML file two options for the user to choose from: to click either a fictitious "button1" or "button2" option. The HTML file would then re-direct the user input to the PHP file so that it can process the data and print the "button" that was selected. When redirected to the PHP file, because GETs method was used, the URL would display the "button" that the user had selected. The HTML file has to be one opened and tested, since the PHP cannot do this initially without the start of the HTML file. This was tested in the same manner as mentioned above.

  For the last part of the lab, PHP was utilized with the BBB hardware. A new PHP file was created with the goal of having the ability to change the state of an LED on the BBB based on user selection. There were three options: to turn on an LED, to turn off and LED and to blink and LED. The file compromised of HTML and PHP code. The HTML section had the "radio" buttons for the user form selection. The PHP section had the control functionality to change the state of the LED based on the user's input (done using GETs method). Additionally, a message is printed to the user showing the current state of the LED (done using a soft message and via hardware using "cat") and the webpage is refreshed every two seconds. Note: the "brightness" and "trigger" settings for the LED on the BBB need to have granted full access beforehand in order for the PHP file to be able to control the LEDs directly on the BBB.

  The overall lab was a success.

Conclusion:

- Successfully implemented and tested HTML POST and GET methods.
- Successfully adopted HTML webpages for user input functionality.
- Successfully understood the basics of the PHP server scripting language.