ECM1417: Web Development

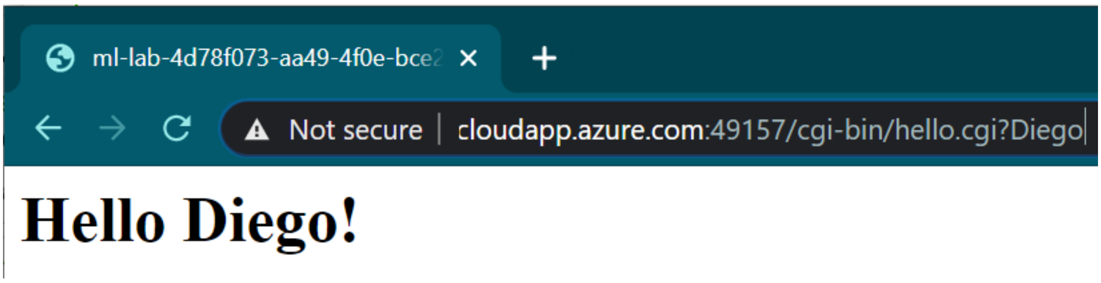
Workshop 04

Matt Collison

February, 2022

In the following workshop you will create dynamic content using CGI and PHP. To this end you will:

- Create a simple CGI script using Python.
- Create a simple PHP script.
- Process form elements using PHP.
- Validate input using PHP.



cloudapp.azure.com:49157/cgi-bin/hello.cgi?Diego

Hello Diego!

Figure 1: Customized greetings using CGI.

1 Common Gateway Interface

Write a CGI script `hello.cgi` in Python which prints a customized greeting in HTML as show in Fig. 1. The script expects a name to be passed via the query string part of the URL and returns a level 1 heading containing "Hello" followed by the name. To do so, execute the following steps:

1. Connect to your azure VM and check if Python is installed. Don't worry if it's an old version we will only be using this for a demo.

   ```
   python --version
   ```

2. Enable the CGI module for your Apache web-server by activating the corresponding module using the line below.

   ```
   sudo a2enmod cgi
   ```

3. Create the script below in the CGI directory, /usr/lib/cgi-bin/, using the commandline editor vi and save as hello.cgi. Note: the environment line is on the very first line of the file and QUERY_STRING can be accessed as an environment variable as discussed in the lecture.

   ```
    #!/usr/bin/env python
   import os
   import sys
   print("Content-type:  text/html")
   print("")
   print("<h1>Hello " + os.environ['QUERY_STRING'] + "</h1>")
   ```

4. Make your hello.cgi script executable by changing the file permissions.

   ```
   sudo chmod 705 /usr/lib/cgi-bin/hello.cgi
   ```

5. Access the script using your browser. Note, the query is any text after the question mark. I've included the link to my CGI script below, obviously you

will need to substitute your own domain, port, path and query to make your own URL.

http://ml-lab-4d78f073-aa49-4f0e-bce2-31e5254052c7.ukwest.cloudapp.azure.com:65495/cgi-bin/hello.cgi?Matt

2 PHP Basics

In the following you should create a simple PHP script to process some numbers:

1. First, activate PHP on your web server. You will need to install PHP and create the Apache modules so it can be executed within the web server environment. Note, you will probably need to update the linux distribution installer with the first command before installing php7.3. As this is installing you should see it creates a series of php modules.

   ```
    apt-get --allow-releaseinfo-change update
   sudo apt-get install php7.3
   ```

2. Next check that the php modules are installed and enabled. List the third party modules enabled by entering the command below.

   ```
   ls /etc/apache2/mods-enabled/
   ```

   You should see the config file for the php module. Before moving on make sure apache is restarted so the enabled modules are loaded. All being well your web server is now setup to run php files anywhere, not just within the CGI directory.

   ```
   sudo systemctl restart apache2
   ```

3. Then, create a PHP script which reads a variable ”number” sent via the GET method and stores it into a local variable. The script should analyse the value of the variable and output the following:

   - If the variable is not set: ”not set”
   - If it is set but not a number: ”not a number”
   - If it is set and an even number: ”even”
   - If it is set and an odd number: ”odd”

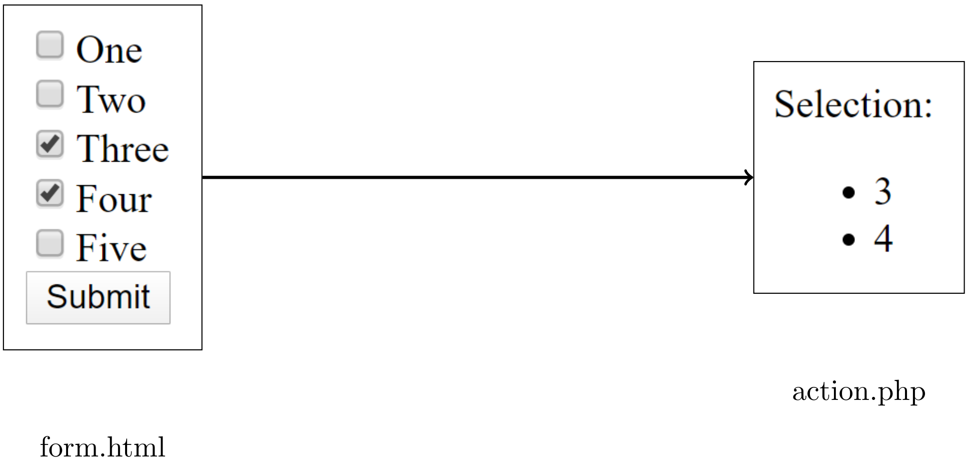
   Note: the solution for this php file is on ELE.

4. Deploy the script to your web server by simply transferring it using FileZilla (sftp protocol through the ssh port) or scp from the commandline.

5. Use your browser to send some GET requests changing the GET variables from the URL.

6. Change the script to read POST variables instead of GET.

7. Use your browser to send some POST requests - wait you can’t because the browser requires a html form action to generate a POST request - continue to the next step or try sending the POST request from telnet.



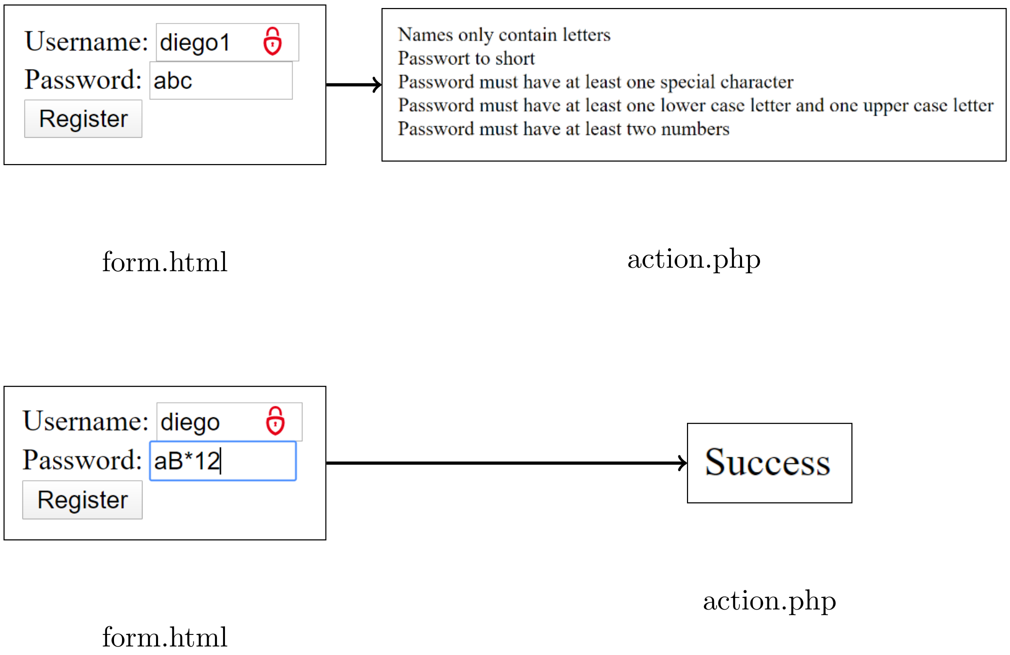
```
Selection:

  • 3
  • 4
```

Figure 2: A simple form with a corresponding action.

3 PHP Forms

Create a simple HTML form and a corresponding action script as shown in Fig. 2 or download the form and action02 php file from ELE. The form contains five different checkboxes and the script uses a ”loop” condition to output the value of all of the checked boxes using an unordered list. If no checkbox is selected the output should say: ”Nothing selected”.

Try extending the form and practice extracting and processing inputs from the POST request with php.



Username: diego1
Password: abc

```
Names only contain letters
Passwort to short
Password must have at least one special character
Password must have at least one lower case letter and one upper case letter
Password must have at least two numbers
```

form.html

action.php

Username: diego
Password: aB*12

```
Success
```

action.php

form.html

Figure 3: A simple form to validate passwords.

4 Form Validation

In this exercise you should create another form which allows the user to input a name and a password as shown in Fig. 3. The form should validate whether the name only contains upper or lowercase letters from A to Z.

The password should be validated according to the following criteria:

- It is at least 5 characters long,
- It has at least one special character (*,#,+,-),
- It has at least one lower case letter and one upper case letter,
- It has at least 2 numbers, and
- It does not contain a white-space.

If successful, the script should redirect the user to a new page which displays ”success”. If one or more conditions are violated, the script should output all the violated conditions and not redirect the user to the success page. Again, a basic solution has been provided on ELE and you should experiment with extending the code.