

**Course** : Diploma in Electronics & Computer Engineering (EGDF20)

**Module**  : Connected System Design Project (EGE205)

**Laboratory No**. : Lab 5a

**Laboratory Title** : Networking: Setting up a Web Server using BeagleBone Black Wireless (BBBW)

Board

**Objective** : To use a BeagleBone Black Wireless (BBBW) Board to set up a web server.

**Hardware Boards** : BBBW Board x1

**Contents**

1. Setting Up a Web Server using BeagleBone Black Wireless (BBBW) Board
   1. Understanding of Web Server and Hypertext Transfer Protocol (HTTP)
   2. Installing the Flask-Socketio Python Library and Retrieving the IP Address from BBBW Board
   3. Establishing and Accessing the Web Server in BBBW Board
   4. Controlling and Monitoring the BBBW Board via Web Server
   5. Tinkering Time

# **Setting Up a Web Server using BeagleBone Black Wireless (BBBW) Board**

## Understanding of Web Server and Hypertext Transfer Protocol (HTTP)

**Web Server** can refer to hardware or software, or both working together.

On the hardware side, a web server is a computer that stores web server software and a website's component files. On the software side, a web server includes several parts that control how web users access hosted files.

The main job of a web server is to display website content through storing, processing, and delivering webpages to users. At a minimum, there is an HTTP server. An HTTP server is a software that understands URLs (web addresses) and HTTP (the protocol browser uses to view webpages). An HTTP server can be accessed through a domain name or IP address of the websites it stores, and it delivers the content of these hosted websites to the end user's device. Besides HTTP, web servers also support SMTP (Simple Mail Transfer Protocol) and FTP (File Transfer Protocol), used for email, file transfer and storage.

**How does web server work?**

When a browser requests a page from a web server, the process will follow a series of steps. First, a person will specify a URL in a web browser's address bar. The web browser will then obtain the IP address of the URL by translating the URL through DNS (Domain Name System). This will bring the browser to a web server. The browser will then request the specific file from the web server by an HTTP request. The web server will respond, sending the browser the requested page, again, through HTTP response. The browser will then be able to display the webpage. If the requested page does not exist or if something goes wrong, the web server will respond with an error message.

The HTTP request and response processes is shown in the Figure below.

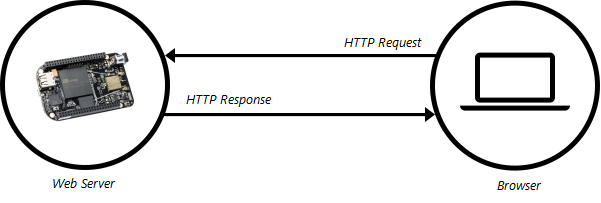


Figure 1.1a: Web Server & HTTP

## Installing the Flask-Socketio Python Library and Retrieving the IP Address from BBBW Board

**Installing the Flask-Socketio Python Library**

1. **Log in** to the BBBW Board through SSH using the default username “**debian**” and password “**temppwd**”.
2. **Type** in the command “**sudo connmanctl**” and **hit** the “Enter” key. **Type** in the password “**temppwd**” and **hit** the “Enter” key again as shown in the Figure below.

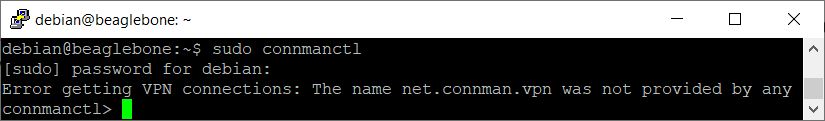


Figure 1.2a: Accessing Connmanctl CLI

1. **Type** in the command “**scan wifi**” and **hit** the “Enter” key. It is observed that the message “**Scan completed for wifi**” is returned as shown in the Figure below.



Figure 1.2b: The “scan wifi” Command

1. **Type** in the command “**services**” and **hit** the “Enter” key. It is observed that the surrounding detected Wi-Fis are listed as shown in the Figure below.

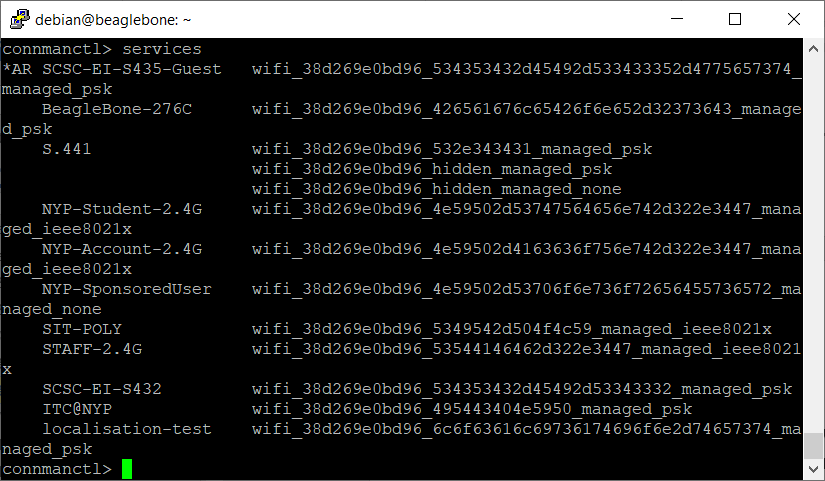


Figure 1.2c: The “services” Command

1. **Type** in the command “**agent on**” and **hit** the “Enter” key. It is observed that the message “**Agent registered**” is returned as shown in the Figure below.



Figure 1.2d: The “agent on” Command

1. **Type** in the command

“**connect wifi\_38d269e0bd96\_534353432d45492d533433352d4775657374\_managed\_psk**” and **hit** the “Enter” key. **Type** in the Wi-Fi passphrase “**helloworld**” and **hit** the “Enter” key again. This is to connect to the Wi-Fi with the SSID of S435-CSDL\_dev. It is observed that the message “**Connected wifi\_38d269e0bd96\_534353432d45492d533433352d4775657374\_managed\_psk**” is returned as shown in the Figure below.

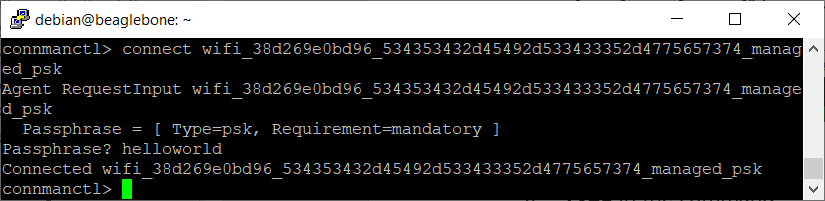


Figure 1.2e: The “connect” Command

1. **Type** in the command “**quit**” and **hit** the “Enter” key as shown in the Figure below.



Figure 1.2f: The “quit” Command

1. **Type** in the command “**pwd**” and **hit** the “Enter” key. **Ensure** that the current working directory is “**/home/debian**” as shown in the Figure below.



Figure 1.2g: Current Working Directory

1. **Type** in the command “**ls**” and **hit** the “Enter” key. **Confirm** that the folder named PythonLibrary has been created as shown in the Figure below. If it is not created, **type** in the command “**mkdir PythonLibrary**” to create the folder.



Figure 1.2h: Ensuring PythonLibrary Folder Is Created

1. **Type** in the command “**cd PythonLibrary**” and **hit** the “Enter” key as shown in the Figure below.

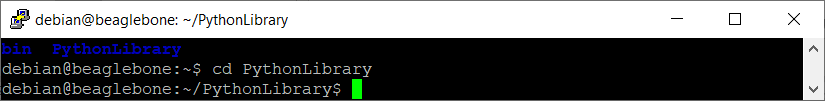


Figure 1.2i: Accessing the PythonLibrary Folder

1. **Type** in the command “**git clone https://github.com/nypege205/Flask-SocketIO.git**” and **hit** the “Enter” key. **Type** in the github username “**nypege205**” and password “**ghp\_Dr3jDaeKJ8fgDH06ZrtG1qUKgsmKux3XffG5**” and **hit** the “Enter” key again. A copy of the Flask-SocketIO Python Library is now cloned into the PythonLibrary folder as shown in the Figure below.

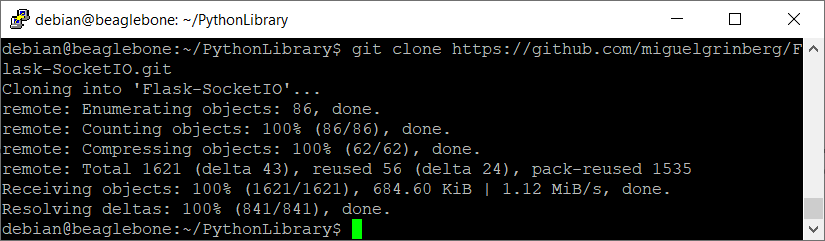


Figure 1.2j: Cloning the Flask-SocketIO Python Library

1. **Type** in the command “**ls**” and **hit** the “Enter” key. A folder named Flask-SocketIO is listed as shown in the Figure below.

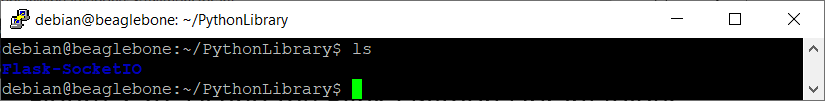


Figure 1.2k: Listing the Flask-SocketIO Python Library Folder

1. **Type** in the command “**cd Flask-SocketIO**” and **hit** the “Enter” key as shown in the Figure below.

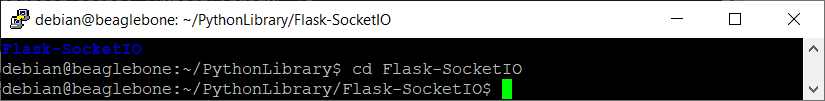


Figure 1.2l: Accessing the Flask-SocketIO Folder

1. **Type** in the command “**sudo python3 setup.py install**” and **hit** the “Enter” key. **Type** in the password “**temppwd**” and **hit** the “Enter” key again. The installation may take up to 5 minutes. The completion of the installation process is shown in the Figure below.

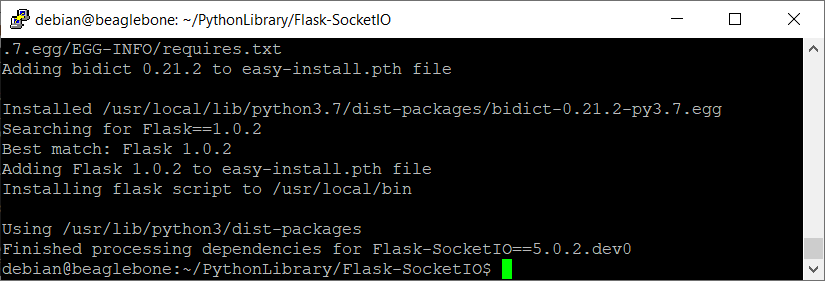


Figure 1.2m: Completion of Flask-SocketIO Python Library Installation

**Retrieving the IP Address from BBBW Board**

1. **Type** in the command “**ifconfig**” as and **hit** the “Enter” key. **Take note** of the IP address for wlan0 of the BBBW board. In this example, the IP address of the BBBW board is **“192.168.1.124”** as shown in the Figure below.

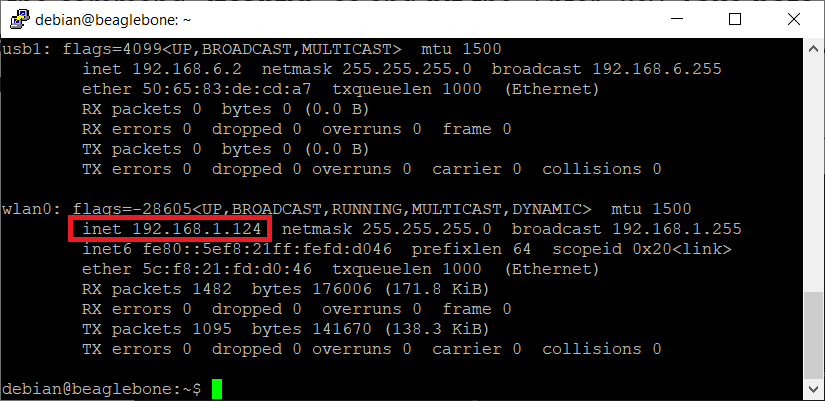


Figure 1.2n: BBBW Board’s IP Address

## Establishing and Accessing the Web Server in BBBW Board

1. **Ensure** that the BBBW board is powered up and connected to the computer through a USB cable. **Open** the web browser (preferably Chrome browser) and **type** “**http://192.168.7.2:3000**” in the address bar.
2. **Right click** on the folder “**MyFirstPythonProject”** and **select** the “**New File**” from the drop-down menu to create a new python file. **Name** the file as “**WebServer.py**”.
3. **Double click** on the newly created file “**WebServer.py**” and **enter** the following code into the file under the Editor section. **Modify** the IP address of “**192.168.X.X**” according to the IP address you have retrieved from the BBBW board earlier.

|  |
| --- |
| from flask import Flask  app = Flask(\_\_name\_\_)  @app.route('/')  def hello\_world():  return 'My First BeagleBone Web Server is Running!'  if \_\_name\_\_ == '\_\_main\_\_':  **app.run(host='192.168.X.X')** |

1. **Click** on the “Run” button located beside the Menu Tab to execute the “**WebServer.py**” file.
2. **Launch** the internet browser of a PC (chrome) or mobile phone (safari) that are connected to the same Wi-Fi network in which the BBBW board is connected to.
3. **Type** the following IP address with port number “**192.168.X.X:5000**” on the URL address bar of the browser and **hit** the Enter key. **Replace** the **X** in the IP address according to the IP address you have retrieved from the BBBW board earlier. In this example, **“192.168.1.2:5000”** is used and it is observed that the text “**My First BeagleBone Web Server is Running**!” is printed on the browser as shown in the Figure below. You have successfully accessed your BBBW web server!

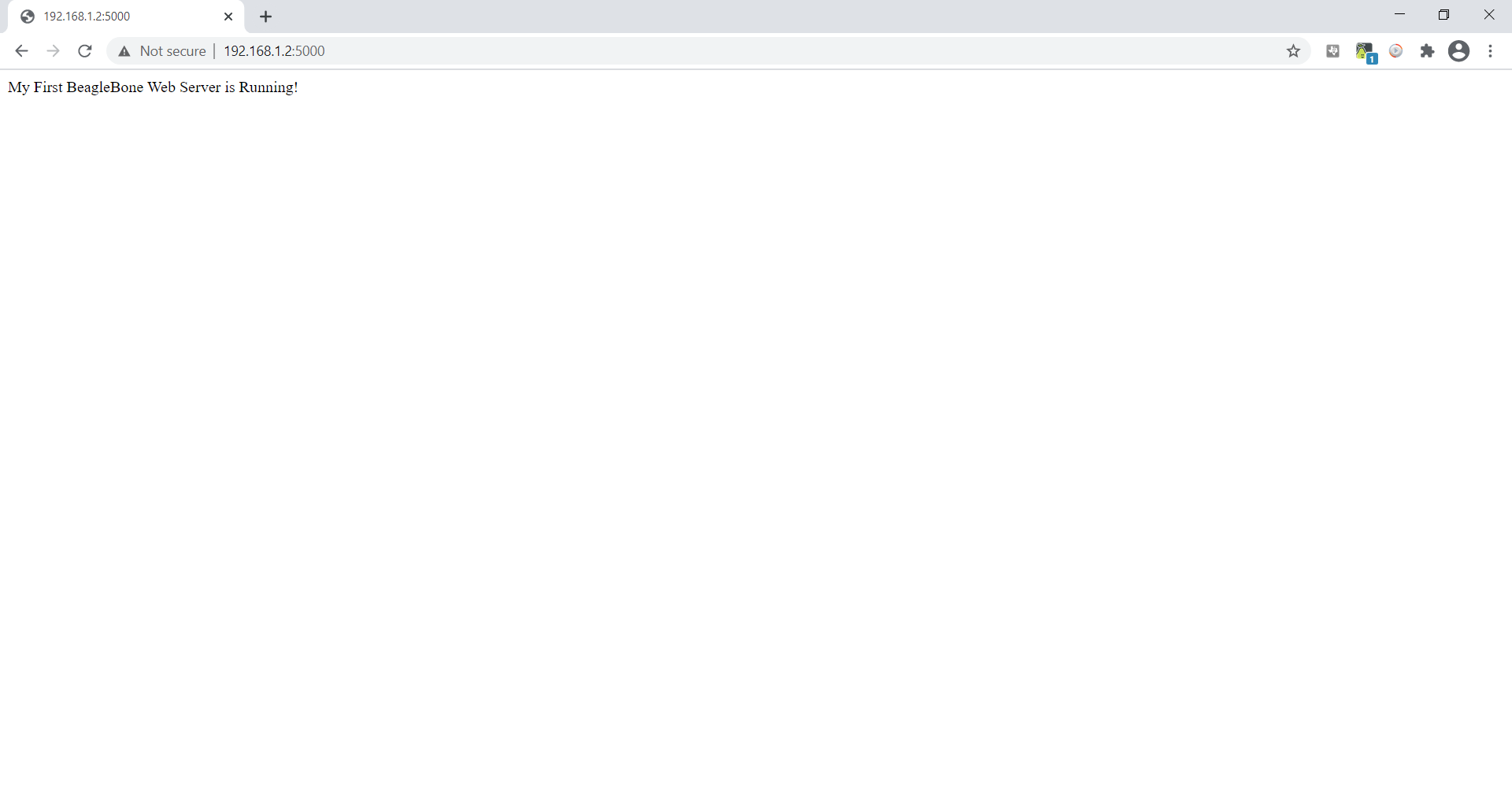


Figure 1.3a: Accessing the BBBW Web Server

## Controlling and Monitoring BBBW Board via Web Server

1. **Ensure** that the BBBW board is powered up and connected to the computer through a USB cable. **Open** the web browser (preferably Chrome browser) and **type** “**http://192.168.7.2:3000**” in the address bar.
2. **Right click** on the **MyFirstPythonProject** folder and **select** the “**New Folder**” from the drop-down menu to create a new folder. Name the folder as “**templates**” as shown in the Figure below.

|  |  |
| --- | --- |
|  |  |

Figure 1.4a: Creating a New Folder

1. **Right click** on the newly created folder “**templates”** and **select** the “**New File**” from the drop-down menu to create a new html file. **Name** the file as “**index.html**” as shown in the Figure below.

|  |  |
| --- | --- |
|  |  |

Figure 1.4b: Creating a New File

1. **Double click** on the newly created html file “**index.html**” and **enter** the following code into the file under the Editor section.

|  |
| --- |
| <html>  <body>  <h1>{{ MyText }}</h1>  </body>  </html> |

1. **Double click** on the python file “**WebServer.py**” and **make** the changes into the existing code under the Editor section with the following code in **BOLD and BLUE**. **Modify** the IP address of “**192.168.X.X**” according to the IP address you have retrieved from the BBBW board earlier.

|  |
| --- |
| from flask import Flask  **from flask import render\_template**  app = Flask(\_\_name\_\_)  @app.route('/')  def hello\_world():  **template\_data={**  **'MyText':'My First BeagleBone Web Server is Running!',**  **}**  **return render\_template('index.html', \*\*template\_data)**  if \_\_name\_\_ == '\_\_main\_\_':  **app.run(host='192.168.X.X')** |

1. **Click** on the “Run” button located beside the Menu Tab to execute the “**WebServer.py**” file.
2. **Launch** the internet browser of a PC (chrome) or mobile phone (safari) that are connected to the same Wi-Fi network in which the BBBW board is connected to.
3. **Type** the following IP address with the port number “**192.168.X.X:5000**” on the URL address bar of the browser and **hit** the Enter key. **Replace** the **X** in the IP address according to the IP address you have retrieved from the BBBW board earlier. In this example, **“192.168.1.2:5000”** is used and it is observed that the text “**My First BeagleBone Web Server is Running**!” is printed on the browser as shown in the Figure below.

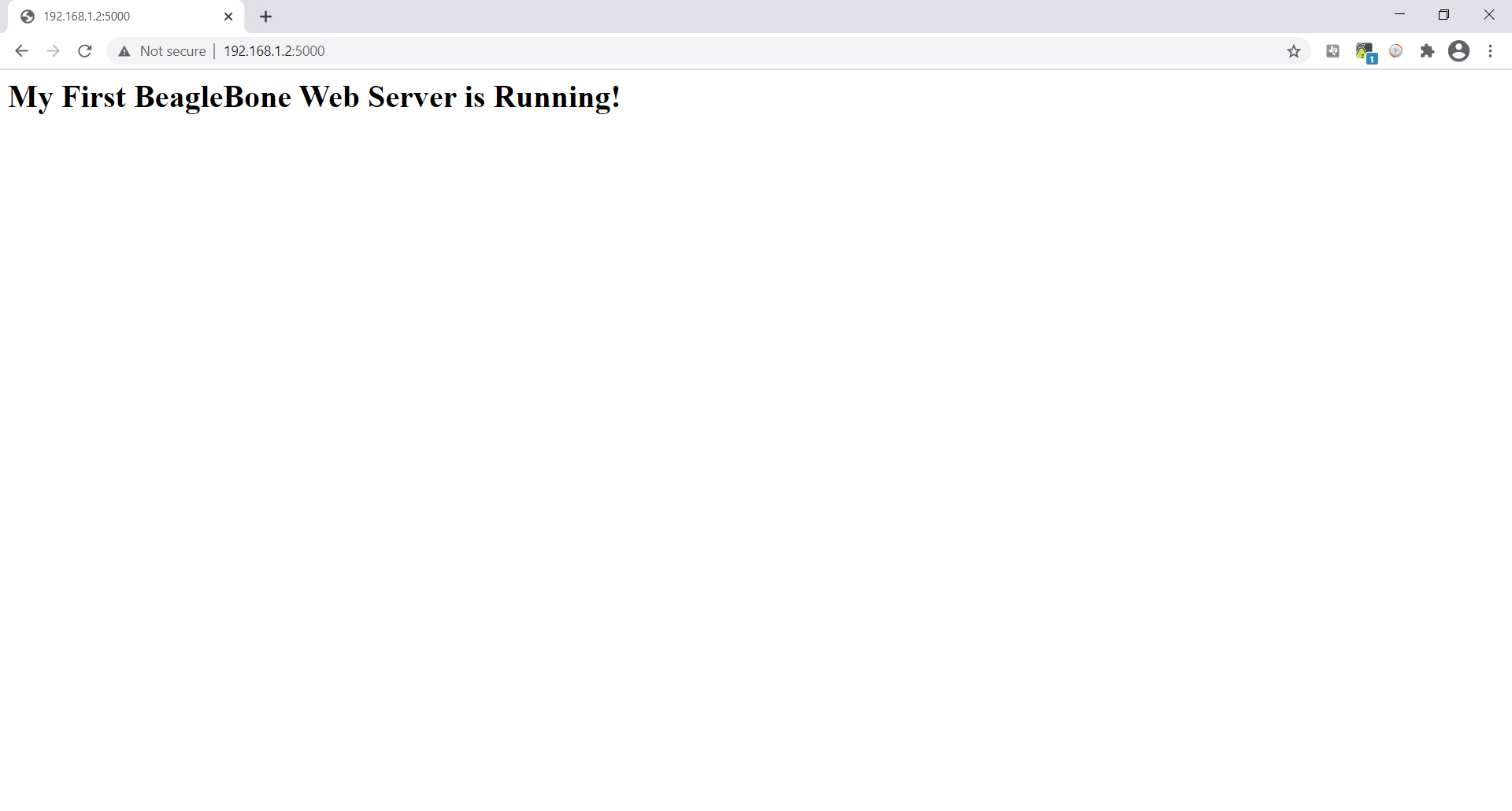


Figure 1.4c: Accessing the BBBW Web Server

1. **Create** a folder under the **MyFirstPythonProject** folder and **name** the folder as “**static**”. Under the static folder, **create** another folder and name it as “**css**”. Under the css folder, **create** a new css file and name it as “**style.css**” as shown in the Figure below.

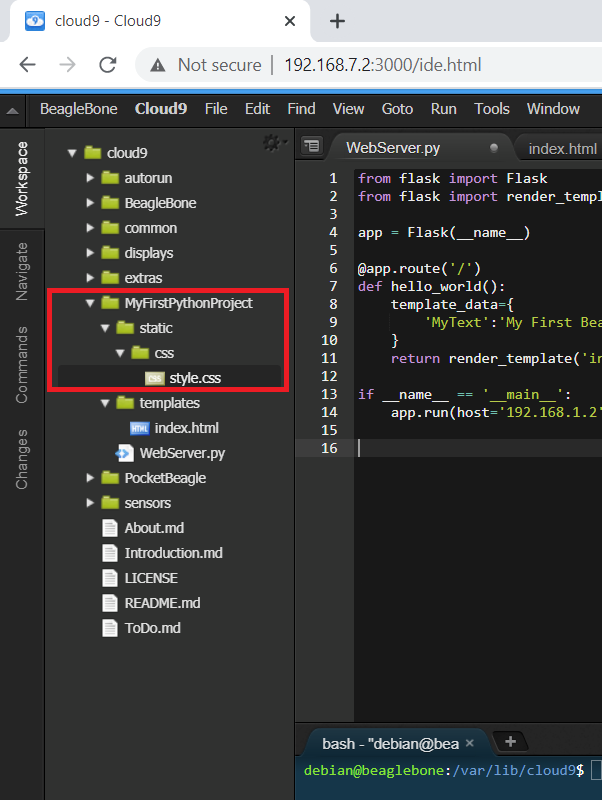


Figure 1.4d: Creating CSS Folder and File

1. **Double click** on the css file “**style.css**” and **enter** the following code into the file.

|  |
| --- |
| body {  font-family: Arial, Helvetica, sans-serif;  font-size: 20px;  color: gray;  }  .button1{  border-radius: 30px;  padding: 16px 32px;  font-size: 16px;  transition-duration: 0.4s;  cursor: pointer;  width: 100px;  background-color: white;  border: 2px solid #008CBA;  }  .button1:hover {  background-color: #008CBA;  color: white;  } |

1. **Double click** on the html file “**index.html**” and **make** the changes into the existing code under the Editor section with the following code in **BOLD and BLUE**.

|  |
| --- |
| <html>  **<head>**  **<link rel="stylesheet" href="{{ url\_for('static', filename='css/style.css') }}">**  **</head>**  **<body>**  **<center>**  **<p>Controlling the USR0 LED </p>**  **<a href="/on"><button class="button button1">ON</button></a>**  **<a href="/off"><button class="button button1">OFF</button></a>**  **<p>USR0 LED Status: {{ LedStatusText }}</p>**  **<center>**  **</body>**  </html> |

1. **Double click** on the python file “**WebServer.py**” and make the changes into the existing code under the Editor section with the following code in **BOLD and BLUE**. **Modify** the IP address of “**192.168.X.X**” according to the IP address you have retrieved from the BBBW board earlier.

|  |
| --- |
| from flask import Flask  from flask import render\_template  **import Adafruit\_BBIO.GPIO as GPIO**  app = Flask(\_\_name\_\_)  **GPIO.setup("USR0", GPIO.OUT)**  @app.route('/')  **@app.route('/<state>')**  **def UpdateLEDStatus(state=None, ledstatus='OFF'):**  **if state == 'on':**  **GPIO.output("USR0", GPIO.HIGH)**  **ledstatus = 'ON'**  **if state == 'off':**  **GPIO.output("USR0", GPIO.LOW)**  **ledstatus = 'OFF'**  **template\_data={**  **'LedStatusText':ledstatus,**  **}**  **return render\_template('index.html', \*\*template\_data)**  if \_\_name\_\_ == '\_\_main\_\_':  **app.run(host='192.168.X.X')** |

1. **Click** on the “Run” button located beside the Menu Tab to execute the “**WebServer.py**” file.
2. **Launch** the internet browser of a PC (chrome) or mobile phone (safari) that are connected to the same Wi-Fi network in which the BBBW board is connected to.
3. **Type** the following IP address with port number “**192.168.X.X:5000**” on the URL address bar of the browser and **hit** the Enter key. **Replace** the **X** in the IP address according to the IP address you have retrieved from the BBBW board earlier. In this example, **“192.168.1.2:5000”** is used it is observed that the following user interface is printed on the browser as shown in Figure below.

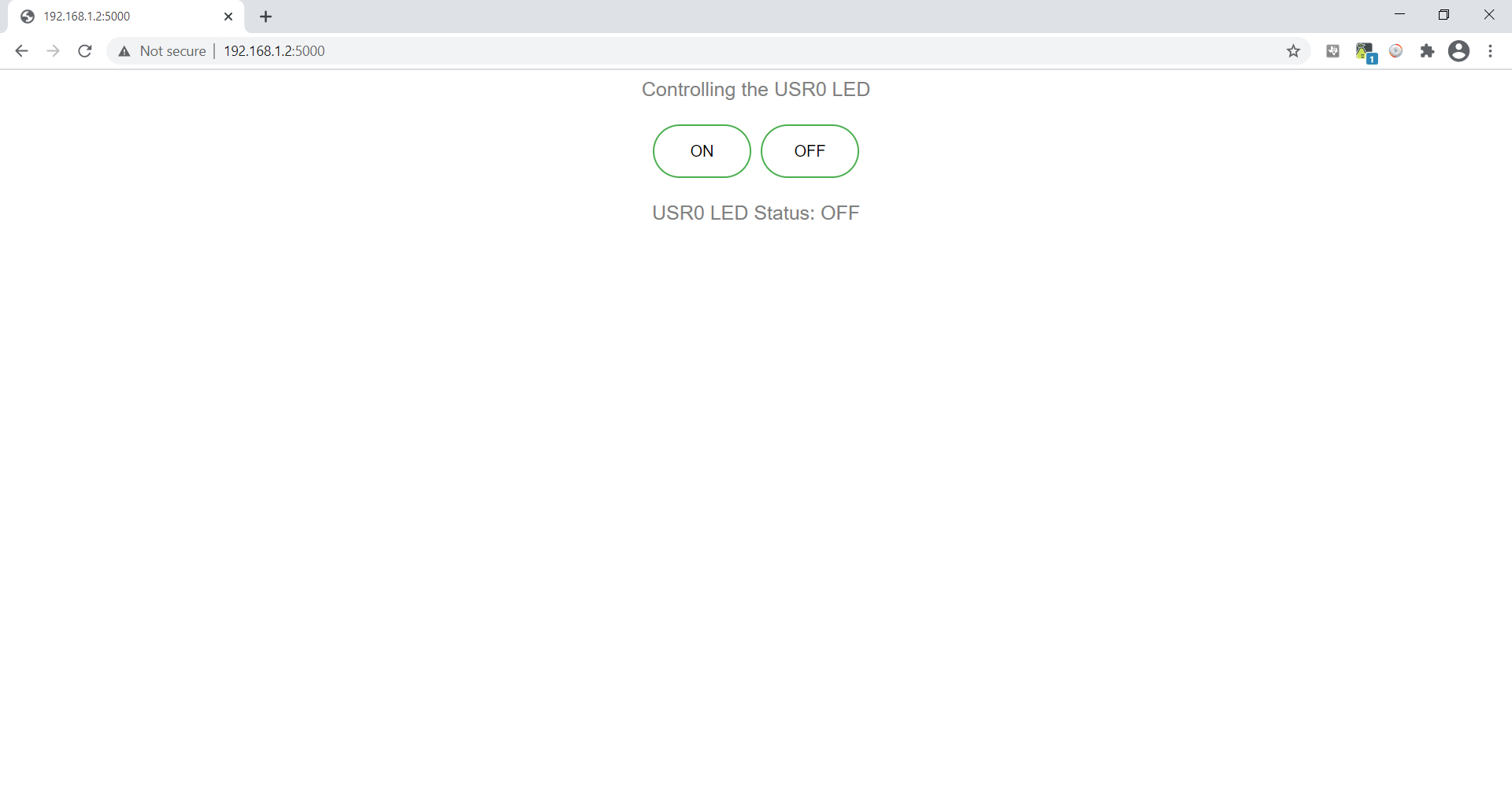


Figure 1.4e: Controlling the LED via the BBBW Web Server

1. **Click** on the ON and OFF button on the webpage. It is observed that the USR0 led located on the BBBW board responds according to the button pressed. The text indicating the status of the USR0 led on the webpage also change accordingly.

## Tinkering Time

1. Together with a classmate or two, **think** of a simple application that can control any of the 4 display clicks (BarGraph 2, 7Seg, 8x8R or OLEDB Click) and feedback to user on the display status through the webpage.
2. **Connect** the selected clicks to the mikroBUS cape and BBBW board.
3. **Create** a python file in Cloud9 IDE and start writing your code.
4. **Present** your complete work to your lecturer for advice.
5. **Share** your work with your other classmates and teach them how you do it if they are interested.

*Congratulations! You have successfully completed the Lab5a. Good job! Take a good break and stay tune for next lab. More exciting lab exercises coming to you!*