Course: EGDF20 Diploma in Electronic and Computer Engineering

Module: EGE356 IoT System Architecture & Technology

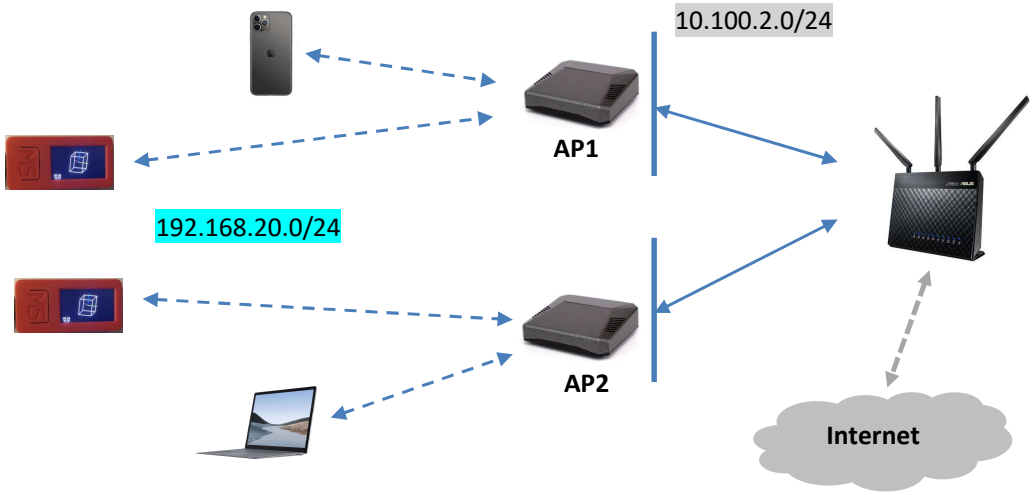
Lab 4: Device to Gateway Connectivity

Objectives:

1. Configure Hostapd and Setup AP using Raspberry Pi
2. Connect Edge device to Gateway
3. Deploy and Access Gateway WebServices

Part 1: Build an on-premise Access Point

1. In this part of the lab, the following network will be setup (see network diagram below):

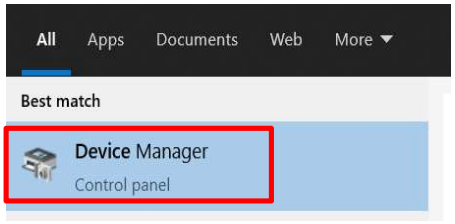


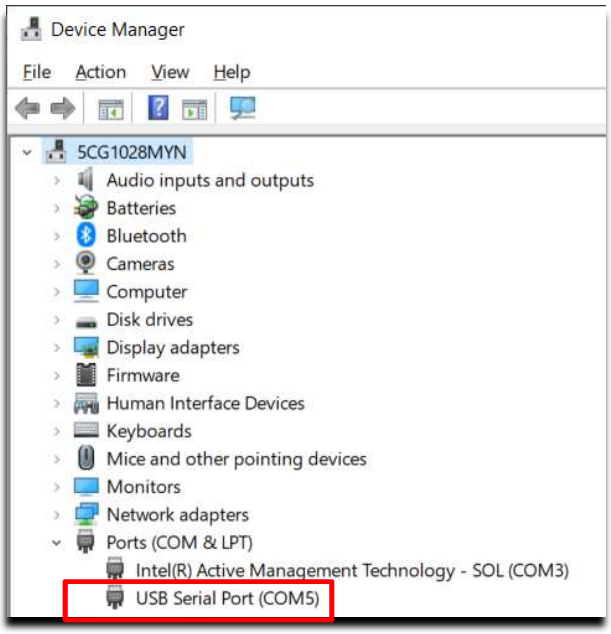
2. Note that ip addresses assigned for each device will be given by the lab instructor. Do not use the above set of address. The Raspberry PI contains the necessary software and pre-configurations setup. **Hostapd** has also been installed.

3. Follow the next steps below to setup the Raspberry PI as an AP.

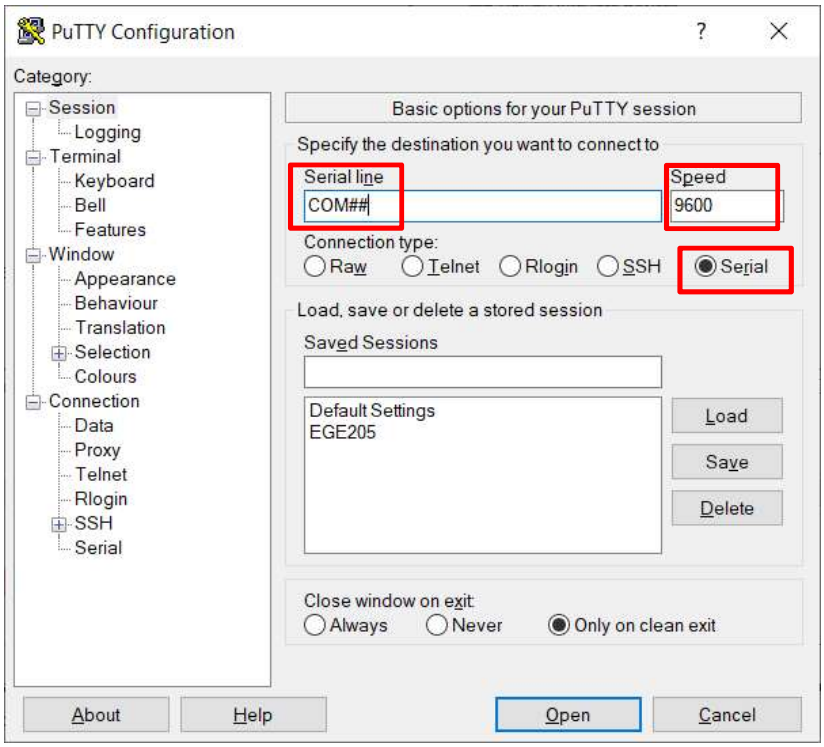
    a. Access the Raspberry Pi using putty.exe and the USBSerial connection.

    b. Once the USB cable is connected, launch Device Manager to identify the COM port number assigned to the connection. Note that the assigned port number may be different from that shown below. Use the COM port number assigned by your PC.

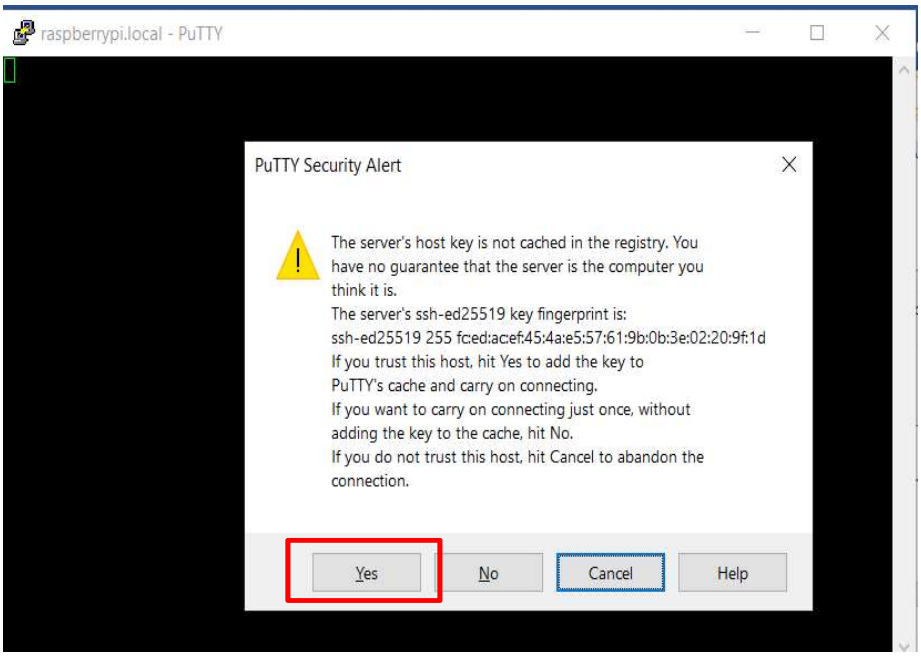




c. When connecting via usbserial, configure the following settings in putty as shown. Replace the ## with the assigned com port number (e.g. COM5, COM21, etc)



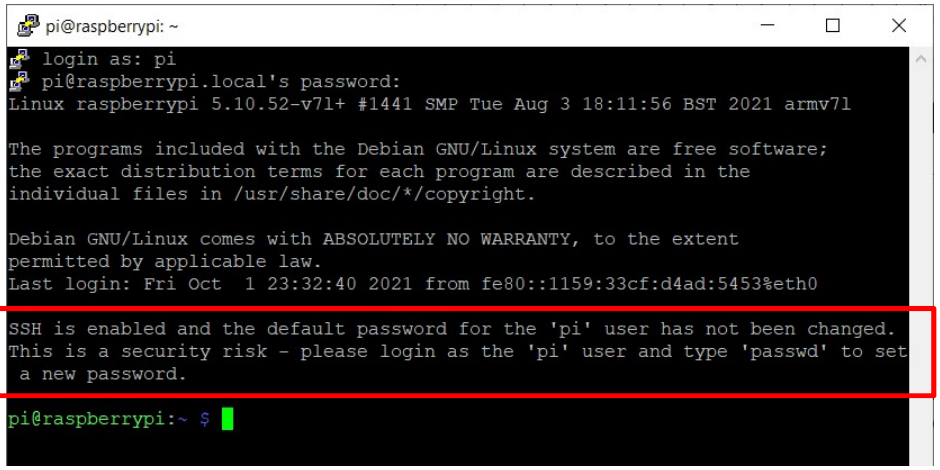
d. Once connection is successful, Putty Security Alert will prompt for you to trust the host and accept the certificate. Click on yes as highlighted in red to proceed.



e. The login prompt will appear as shown below after accepting the server's host key.

f. The default username is ***pi*** and the default password for Raspberry Pi is ***raspberry***. However, in this lab, the password has been set to ***ege356labs***. If the password does not work, check with the instructor on the password to be used.

g. On successful login, the console display is as shown below:



h. Use the highlighted commands to check for hostapd.conf file.

```
pi@iotgw-S1:/$ cd /etc/hostapd
pi@iotgw-S1:/etc/hostapd$ ls -l
total 8

-rwxr-xr-x 1 root root 3129 Apr 16 21:07 ifupdown.sh
pi@iotgw-S1:/etc/hostapd$
```

4. Use the command below to access the hostapd.conf file. It will create the hostapd.conf if it is not present, and if present, it will open the file.

```
sudo nano hostapd.conf
```

5. Take note that in modifying the hostapd.conf file, each learner will be allocated a unique SSID and wpa_passphrase to be configured for each Raspberry Pi Access Point. ***The values highlighted below are to be configured according to allocation by the lab instructor.***

6. Modify the highlighted areas to configure the hostapd.conf file as an Access Point:

```
country_code=SG
ieee80211d=1
ieee80211h=1
interface=wlan1
driver=nl80211
ssid=SCSC_IOTA
hw_mode=g
channel=6
wmm_enabled=0
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=SCSC_HOME_IOTA
wpa_key_mgmt=WPA-PSK
#wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

```
# the interface used by the AP
interface=wlan0
# "g" simply means 2.4GHz band
hw_mode=g
# the channel to use
channel=10
# limit the frequencies used to those
allowed in the country
ieee80211d=1
# the country code
country_code=FR
# 802.11n support
ieee80211n=1
# QoS support, also required for full
speed on 802.11n/ac/ax
wmm_enabled=1

# the name of the AP
ssid=somename
# 1=wpa, 2=wep, 3=both
auth_algs=1
# WPA2 only
wpa=2
wpa_key_mgmt=WPA-PSK
rsn_pairwise=CCMP
wpa_passphrase=somepassword
```

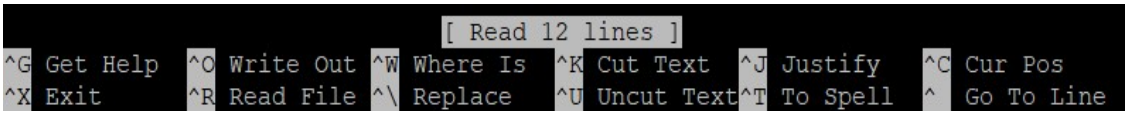
7. Check that the shell script, iotgwnet.sh, contains the commands to start the hostapd daemon.

```
sudo nano  /etc/iotgwnet.sh
```

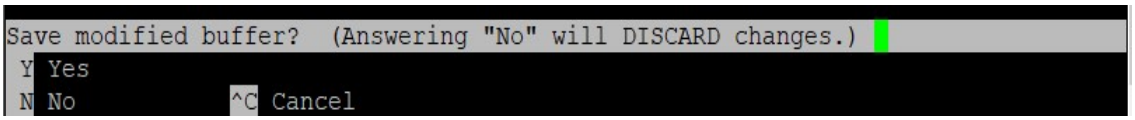
Check that the highlighted code is present and not commented out in the iotgwnet.sh file and save the file.

```
hostapd -B /etc/hostapd/hostapd.conf >> /var/log/hostapdup.log 2>&1
```

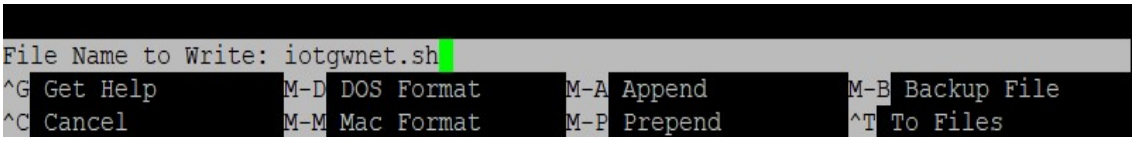
Type Ctrl X to exit if no changes are required. If changes were made, type Ctrl X, type Y as shown below to save the file.



Type Ctrl-X



Type Y



Press the Enter key to save the changes to the file.

8. Edit the rc.local file as shown below.

```
sudo nano /etc/rc.local
```

Use the command to open the rc.local file.

```
#!/bin/sh -e
#
# rc.local
#
. . .
. . .

if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi

sh /etc/iotgwnet.sh

exit 0
```

The current image used already has rc-local.service enabled, as such rc.local will be executed at bootup and the
'sh /etc/iotgwnet.sh'
will run hostapd to initialize and start AP mode

Check that the code highlighted is not commented out. If commented out, uncomment by remove the #. Do not make any other changes.

Type Ctrl X to exit if no changes are required. If changes were made, type Ctrl X, type Y as shown below to save the file.

9. Use the same command as in step 7 to check that the highlighted code in iotgwnet.sh is uncommented. If it is commented, uncomment it and save the file.

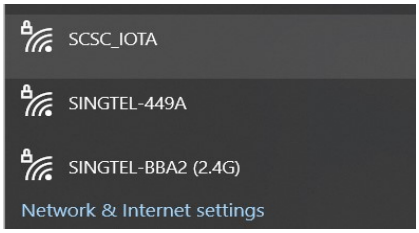
```
sudo nano  /etc/iotgwnet.sh
```

```
#!/bin/sh
gpio mode 26 out
gpio write 26 1
hostapd -B /etc/hostapd/hostapd.conf >> /var/log/hostapdup.log 2>&1
. . . .

sudo iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE
```

10. Reboot the Raspberry Pi using the command below:

```
sudo reboot
```

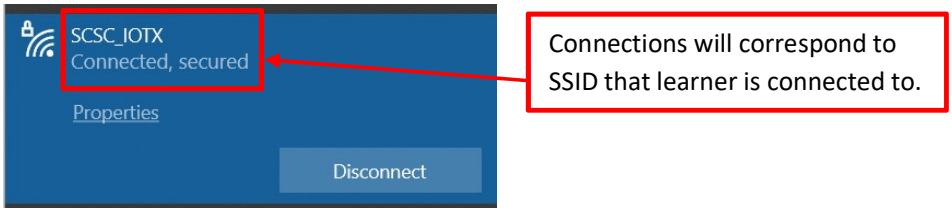
11. To check that the AP is now in operation, use either the PC, laptop or smart phone to check that the "SCSC_IOTA" SSID is listed after a wifi scan.




```
pi@iotgw-S1:~ $ ping google.com.sg
PING google.com.sg (142.251.10.94) 56(84) bytes of data.
64 bytes from sd-in-f94.1e100.net (142.251.10.94): icmp_seq=1 ttl=104 time=9.29
ms
64 bytes from sd-in-f94.1e100.net (142.251.10.94): icmp_seq=2 ttl=104 time=6.02
ms
64 bytes from sd-in-f94.1e100.net (142.251.10.94): icmp_seq=3 ttl=104 time=4.68
```

12. You may now connect the PC or laptop to the AP.

13. Note that when connected to the AP, the PC or laptop will automatically receive DHCP IP address. This has been pre-configured and setup using dnsmasq.conf file. No changes should be made to this file.



```
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : byteacs.com
   Link-local IPv6 Address . . . . . : fe80::1159:33cf:d4ad:5453%7
   IPv4 Address. . . . . . . . . . . : 192.168.23.193
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.23.1
```

IP address will correspond to RPi configured and allocated to learner's PC/laptop. Will likely be different for learners

```
C:\Users\SCSC-          >ping google.com.sg

Pinging google.com.sg [142.251.12.94] with 32 bytes of data:
Reply from 142.251.12.94: bytes=32 time=6ms TTL=103
Reply from 142.251.12.94: bytes=32 time=8ms TTL=103
Reply from 142.251.12.94: bytes=32 time=11ms TTL=103
Reply from 142.251.12.94: bytes=32 time=12ms TTL=103

Ping statistics for 142.251.12.94:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 12ms, Average = 9ms
```

14. Although no configuration is required, some parts of the file is shown here as it is important to understand the configurations that provide the ip addresses, and name resolution.

```
bogus-priv
no-resolv

server=1.1.1.1
server=10.100.2.1

listen-address=127.0.0.1,192.168.20.1
expand-hosts
local=/scsc.net/
domain=scsc.net

except-interface=lo
bind-dynamic

interface=wlan1
dhcp-range=192.168.20.100,192.168.20.200,24h
dhcp-option=option:router,192.168.20.1
dhcp-option=option:dns-server,192.168.20.1,10.100.2.1 ,1.1.1.1
dhcp-authoritative
dhcp-leasefile=/var/lib/dnsmasq/dnsmasq.leases
```

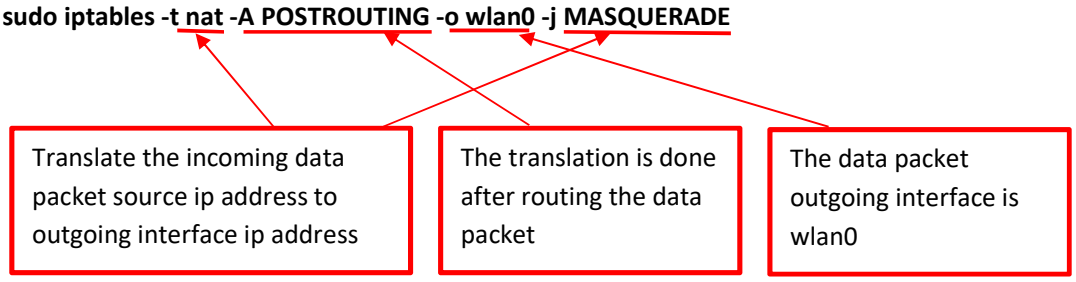
DNS server ip addresses for name resolution

Interface IP address to listen on and respond to DHCP queries, and provide DHCP IP range to DHCP clients

Domain name appended to hostname, e.g. iotgw1.scsc.net

DNS server info provided to DHCP clients

15. The network setup is a stub network. Note also that because the IP addresses used are non-routable, for the routers to connect to the internet, Network Address Translation (NAT) is used. In Raspbian OS, like-wise in other linux based OS, iptables can be used to enable NAT. The figure below provides a guide to understanding the iptables command used in the script.

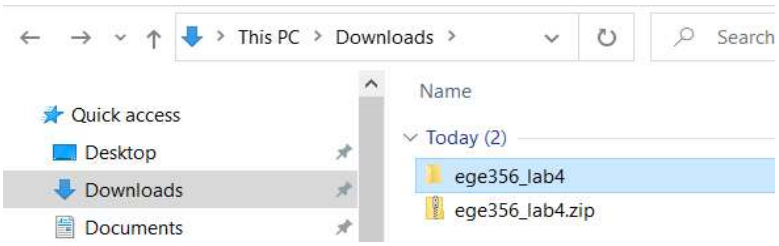


16. Summary: IoT Gateway provides services required for edge devices to connect to the network and the internet. The similarities between IoT gateway and wifi-routers are that they provide DHCP, NAT, Routing and DNS services.

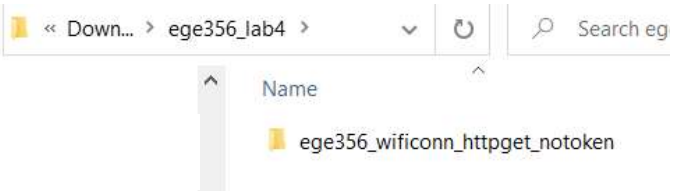
Part 2: Connect Edge Device to Gateway

1. Download the ege356_lab4.zip file.

2. Go to the downloads folder and extract the file as shown.

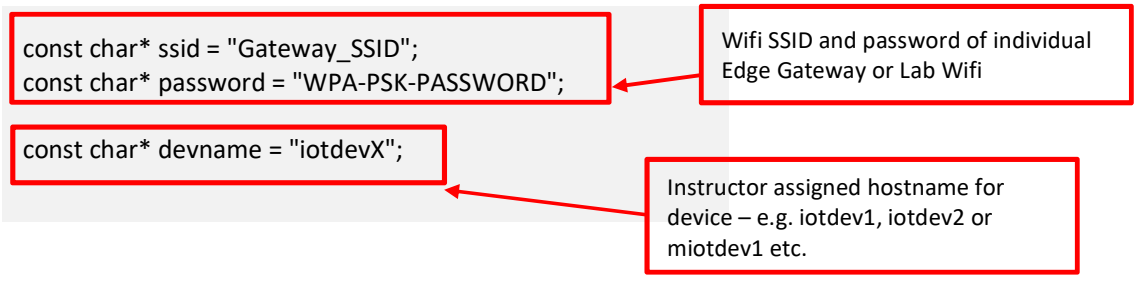


3. Double click on the highlighted folder in ege356_lab4



4. Double click on the highlighted folder and launch the .ino file as shown below.

5. Make the required modifications to the code so that the device will be able to connect to the IoT edge gateway. Modify the SSID and Password based on Lab 4 configuration – according to learner's Wireless A's SSID and SSID password. If the setup in Lab 4 was not completed, you may use the wifi network provided in the lab – obtain the SSID and Password from the instructor.



```
const int buttonA = 37;  //Button-B =39, Button-A = 37
const int buttonB = 39;
int last_valueA = 0;
int last_valueB = 0;
int cur_valueA = 0;
int cur_valueB = 0;

const char* ssid = " ";
const char* password = " ";
const char* devname = " ";
String locIPaddr;
```

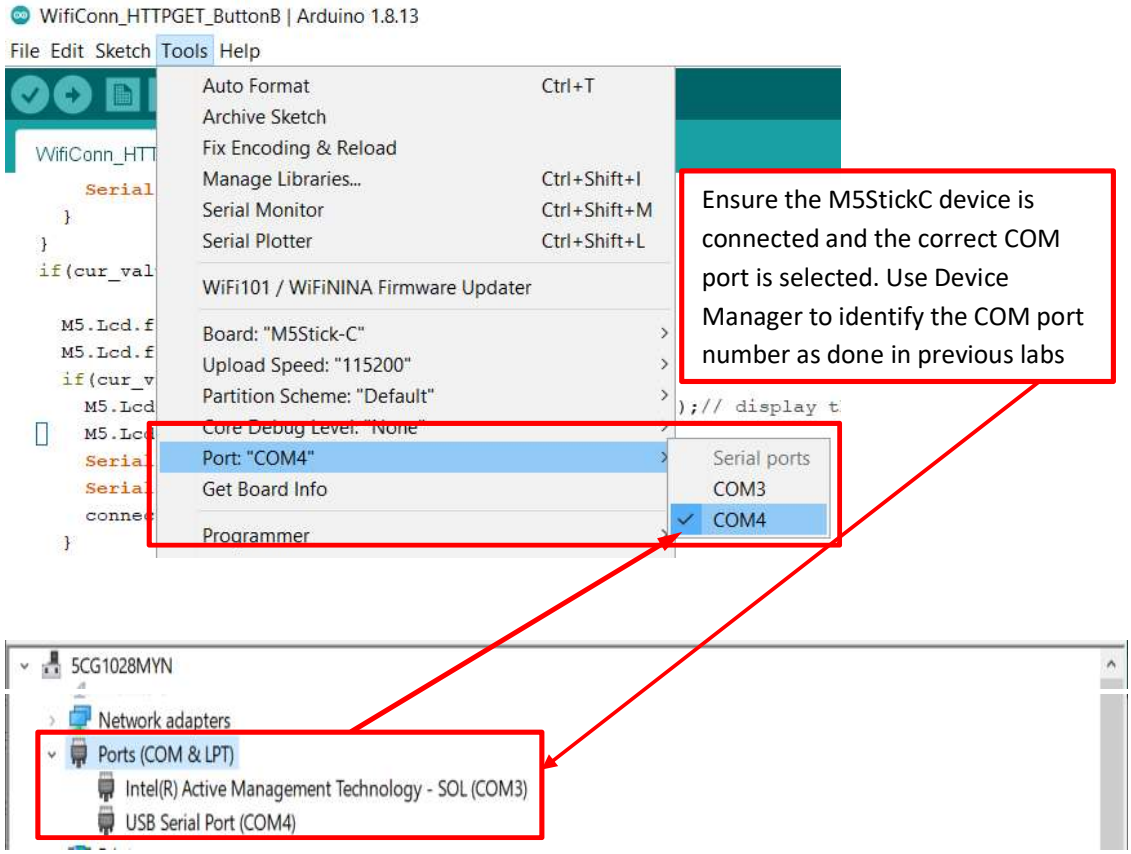
location of the code

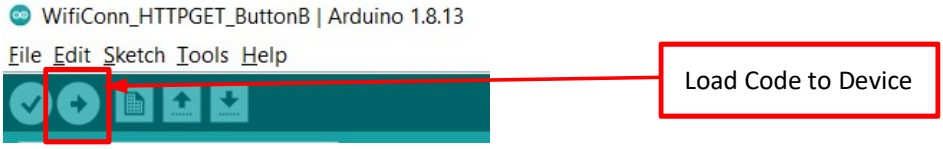
Scroll down to modify the web server URL connection for part 3.

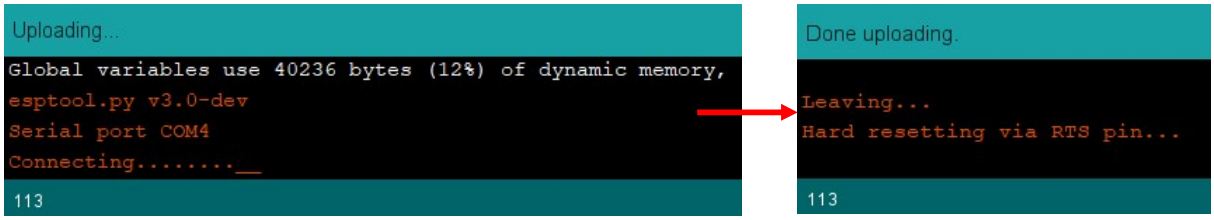
```
void connectToWeb(){

  String serverAPI = "http://192.168.X.X:8000/devices";

  String serverPath = String(serverAPI + "/" + devname); //"?temperature=24.37";

  http.begin(serverPath.c_str());

  // Send HTTP GET request
  int httpResponseCode = http.GET();
```

Modify the code to use the IP Address of the Gateway

6. After modifying the Arduino C code, compile and load the Arduino code to the M5StickC device.





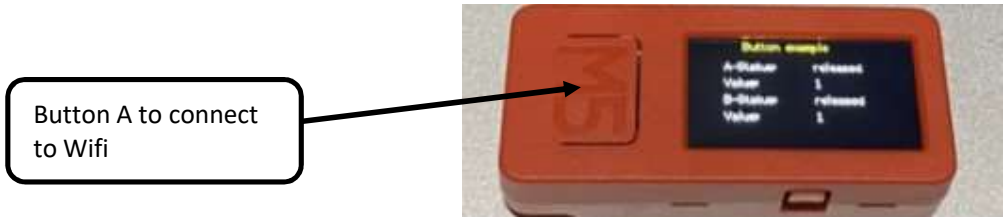


Once the code has been loaded to device, click on Serial Monitor at top right-hand corner.

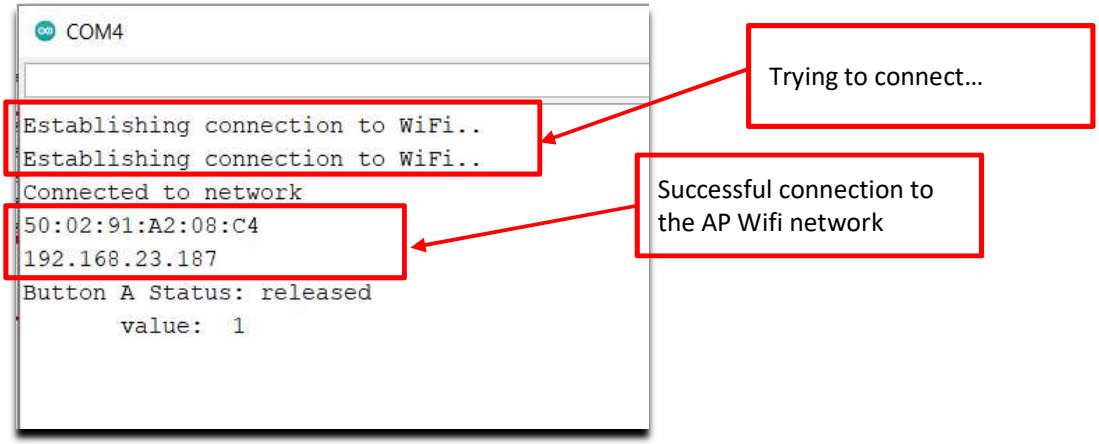
Click on Serial Monitor to connect and view Serial Output from device. Take note of item 7 before continuing.

7. The following codes are required to enable the IoT device to connect to the Wifi network. Note that the function "***connectToNetwork()***" contains the code to connect to the Wireless AP. To activate the function, press Button A on the device as shown below.

```
void connectToNetwork() {
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Establishing connection to WiFi..");
  }
  M5.Lcd.fillRect(40,105,100,25,BLACK);
  Serial.println("Connected to network");
  Serial.println(WiFi.macAddress());
  Serial.println(WiFi.localIP());

  M5.Lcd.setCursor(40,105, 2); M5.Lcd.print(WiFi.localIP());// display the status
}
```



Click ONCE on **Button A Only,** to connect to the Wifi network. DO NOT click or press any other button.



Once the device has connected to the AP network, ***check that the IP address of the M5StickC device and the IP address of the PC are both in the same subnet. In the Raspberry Pi, type ifconfig -a and note that the ip address for wlan1 should be the same subnet as the M5StickC edge device.***

```
wlan1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.23.1  netmask 255.255.255.0  broadcast 192.168.23.255
        inet6 fe80::d6fe:562e:932b:9c3f  prefixlen 64  scopeid 0x20<link>
        ether 34:c9:f0:90:42:e1  txqueuelen 1000  (Ethernet)
```

Ensure that the LabPC/Laptop is connected to the IOT edge gateway. Do a ping to the M5StickC edge device from both the PC/Laptop and the gateway (Rpi).

Ping from gateway

```
pi@iotgw-S1:~ $ ping 192.168.23.187
PING 192.168.23.187 (192.168.23.187) 56(84) bytes of data.
64 bytes from 192.168.23.187: icmp_seq=1 ttl=255 time=79.0 ms
64 bytes from 192.168.23.187: icmp_seq=2 ttl=255 time=99.2 ms
^C
--- 192.168.23.187 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 2ms
rtt min/avg/max/mdev = 78.979/89.065/99.151/10.086 ms
pi@iotgw-S1:~ $
```

Ping from LabPC/Laptop

```
Select Command Prompt
Microsoft Windows [Version 10.0.19044.1586]
(c) Microsoft Corporation. All rights reserved.

C:\Users\SCSC-EI>ping 192.168.23.187

Pinging 192.168.23.187 with 32 bytes of data:
Reply from 192.168.23.187: bytes=32 time=57ms TTL=255
Reply from 192.168.23.187: bytes=32 time=57ms TTL=255
Reply from 192.168.23.187: bytes=32 time=66ms TTL=255
```

Do a ping to the M5StickC edge device from both the gateway (Rpi) to PC/Laptop and the M5StickC edge device.

Ping from Gateway to LabPC/Laptop

```
pi@iotgw-S1:~ $ ping 192.168.23.193
PING 192.168.23.193 (192.168.23.193) 56(84) bytes of data.
64 bytes from 192.168.23.193: icmp_seq=1 ttl=128 time=1.79 ms
64 bytes from 192.168.23.193: icmp_seq=2 ttl=128 time=1.86 ms
64 bytes from 192.168.23.193: icmp_seq=3 ttl=128 time=1.08 ms
```

Ping from Gateway to M5StickC Edge device

```
pi@iotgw-S1:~ $ ping 192.168.23.187
PING 192.168.23.187 (192.168.23.187) 56(84) bytes of data.
64 bytes from 192.168.23.187: icmp_seq=1 ttl=255 time=40.7 ms
64 bytes from 192.168.23.187: icmp_seq=2 ttl=255 time=63.1 ms
64 bytes from 192.168.23.187: icmp_seq=3 ttl=255 time=80.0 ms
```

8. Summary: Edge devices and Gateway connected to the same network will be able to communicate with each other. However, as edge devices have more constrained resources, expect the performance to be slower, and the latency to be higher.

Part 3: Deploy and Access Gateway Services

1. The required files and software for the services have already been uploaded and/or installed in the gateway. To deploy and run the Django Web Server Application, execute the following steps below.

cd ege356

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Apr  9 05:30:18 2022 from 192.168.23.193
pi@iotgw-S1:~ $ cd ege356
pi@iotgw-S1:~/ege356 $
```

source ege356-labs/bin/activate

```
pi@iotgw-S1:~/ege356 $ source ege356-labs/bin/activate
(ege356-labs) pi@iotgw-S1:~/ege356 $
```

cd ege356-lab4_http-mqtt-notoken

```
(ege356-labs) pi@iotgw-S1:~/ege356 $ ls
ege356-lab4_http-mqtt-notoken  ege356-labs  ege356_labs.txt
(ege356-labs) pi@iotgw-S1:~/ege356 $ cd ege356-lab4_http-mqtt-notoken
(ege356-labs) pi@iotgw-S1:~/ege356/ege356-lab4_http-mqtt-notoken $
```

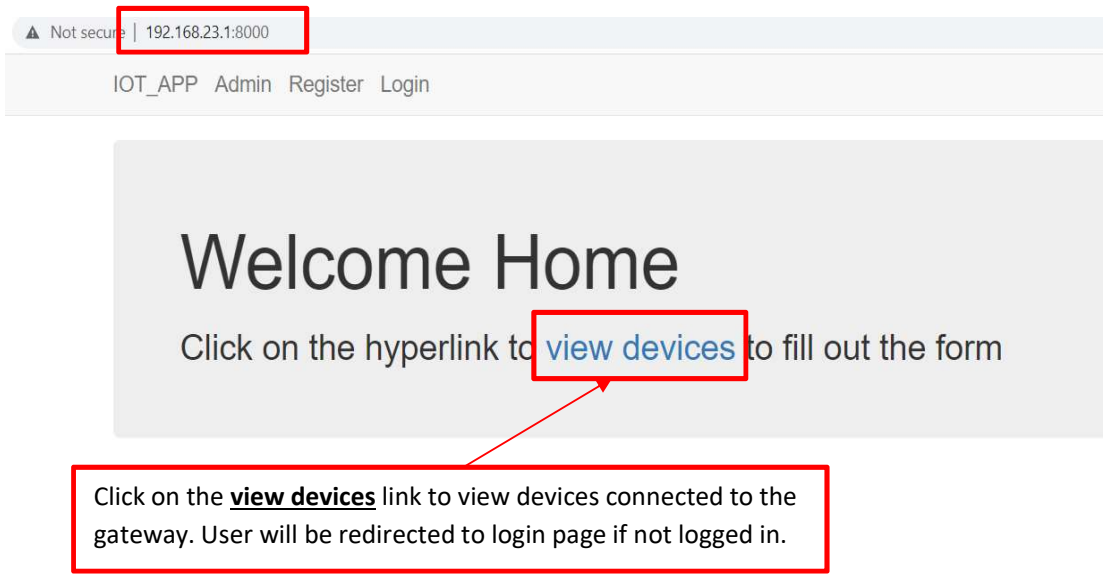
ls
cd devicewebservice

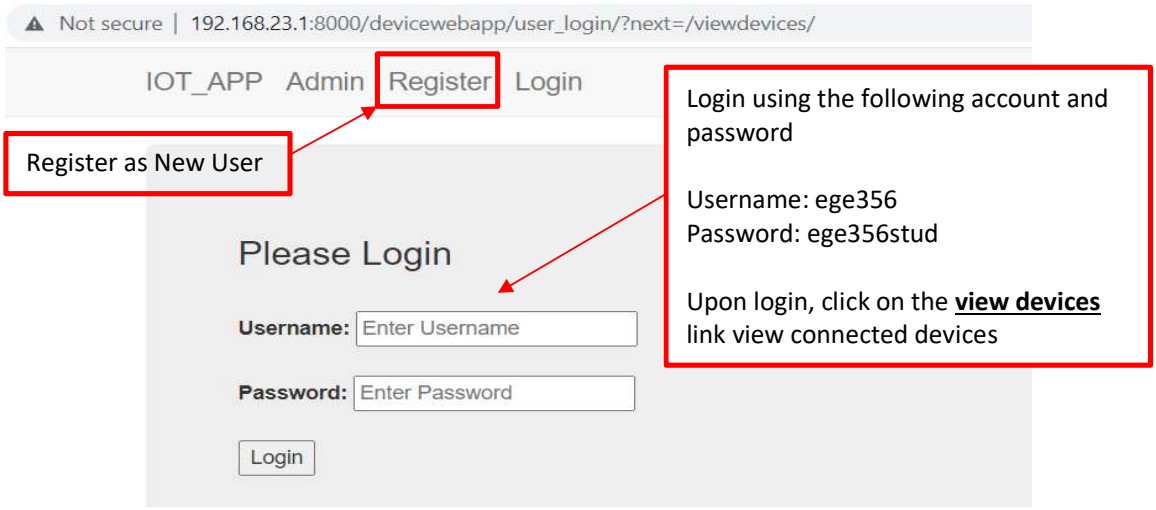
python manage.py runserver 192.168.X.X:8000

Use the gateway's IP address, e.g. 192.168.23.1

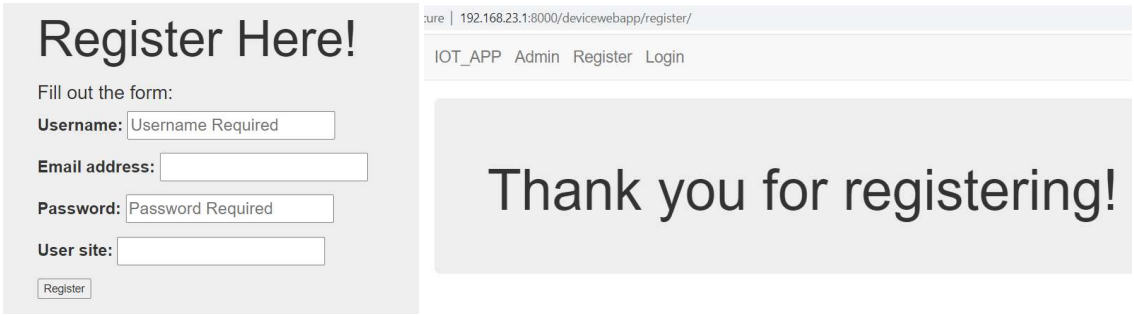
```
(ege356-labs) pi@iotgw-S1:~/ege356/ege356-lab4_http-mqtt-notoken $ ls
devicewebservice
(ege356-labs) pi@iotgw-S1:~/ege356/ege356-lab4_http-mqtt-notoken $ cd devicewebs
ervice
(ege356-labs) pi@iotgw-S1:~/ege356/ege356-lab4_http-mqtt-notoken/devicewebservice $
 python manage.py runserver 192.168.23.1:8000
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
April 09, 2022 - 01:08:06
Django version 3.2.10, using settings 'devicewebservice.settings'
Starting development server at http://192.168.23.1:8000/
Quit the server with CONTROL-C.
```

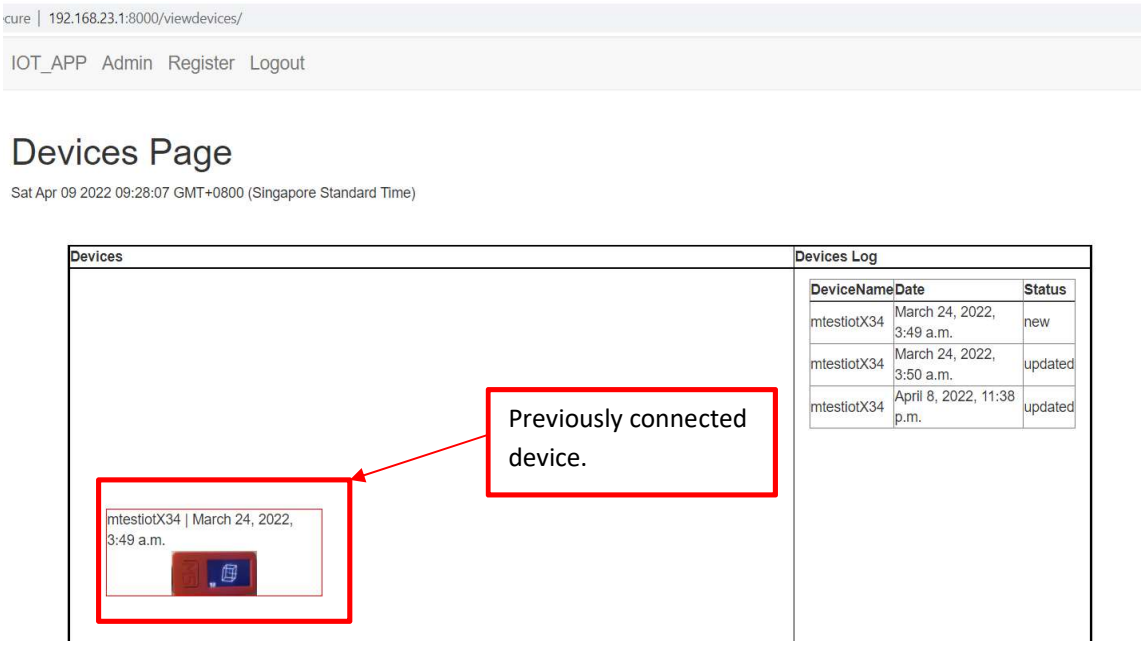
2. Launch google web browser to access the webpage. To access the web server application, type in http://192.168.X.X:8000. Note that the IP address should be the same as ip address used in step 1, the gateway IP address.



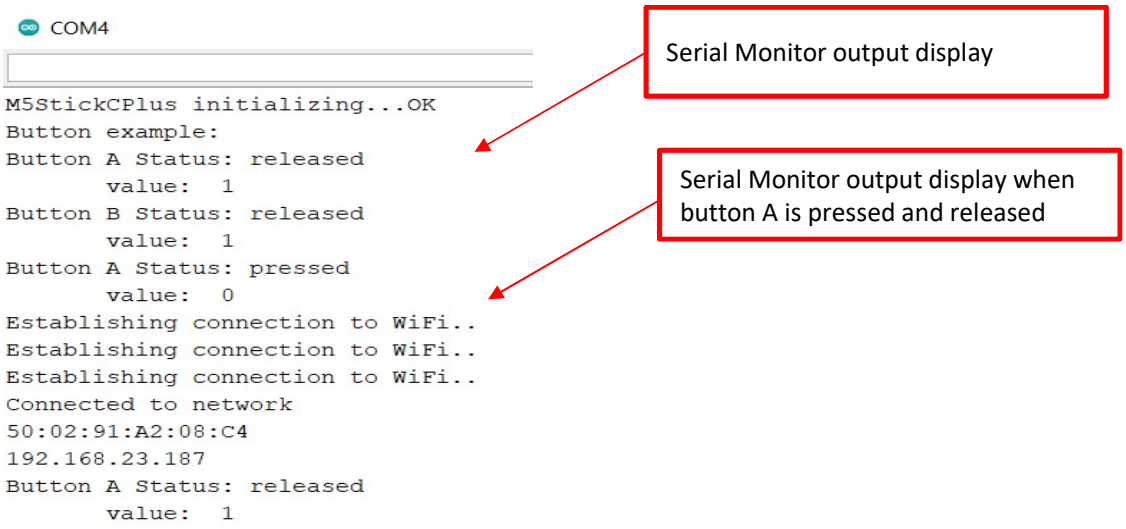


Alternatively, learners may also create a new account and login using the account to access the **view devices** link

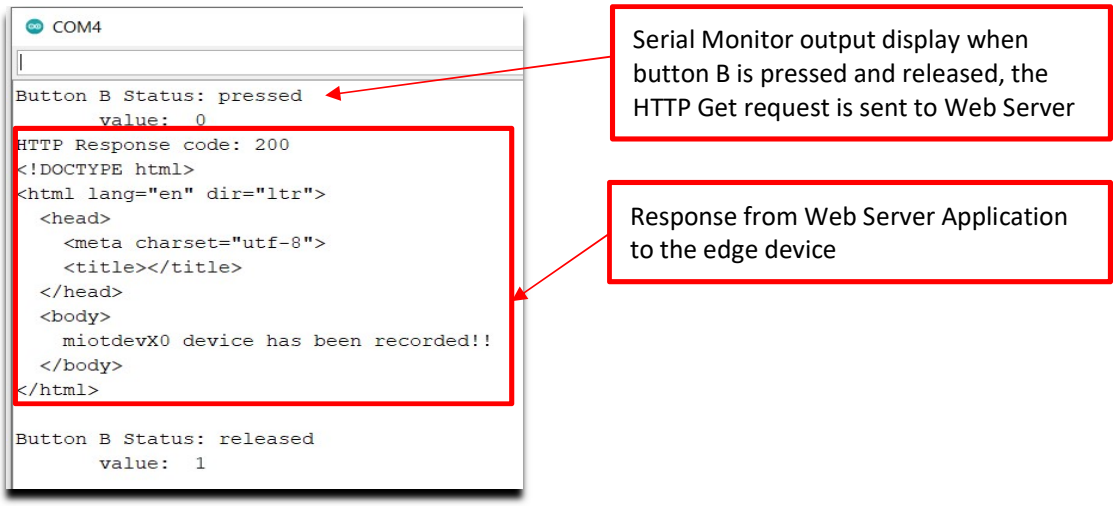


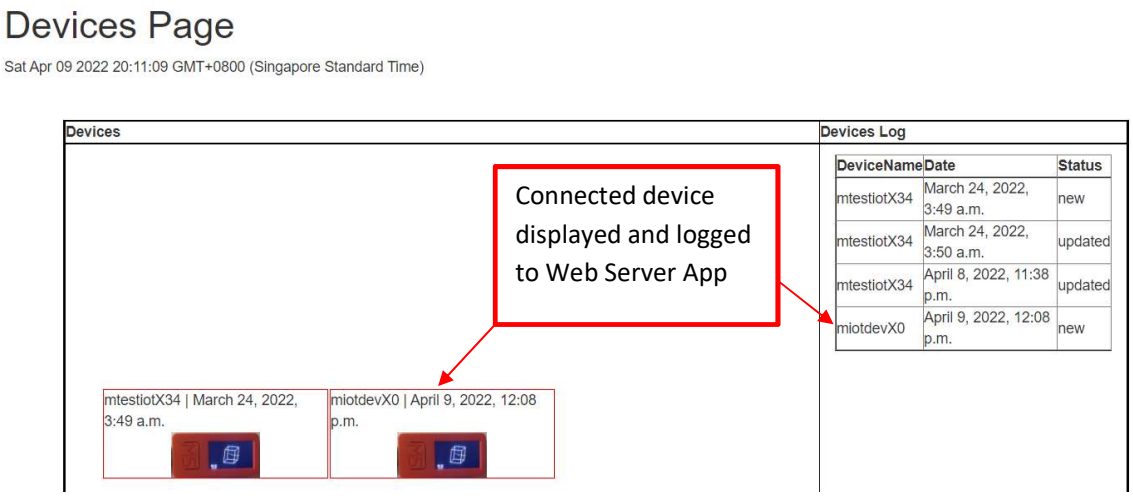


Check to ensure that M5stickC edge device is still connected to the network by pinging the device. If it is not connected, reload the code and press Button A to connect the device to the Gateway Wifi-network as done in part 2. Ensure that the Arduino serial monitor is running.



Once connected, press Button B.





To test that multiple edge devices can be connected to the gateway, learners may connect to instructor's gateway and observe that devices get added to the display in "real time" as they are connected.

Learners may also connect to each other's gateway to observe the effects. Note also that the Web Server Application displays a different device image when the name used starts with "iotdevXX" as shown below.

3. Mouse over the miotdevXX device and the iotdevXX. Observe that iotdevXX is not clickable while for miotdevXX, the user is redirected to an MQTT web page.

4. Summary: Web Server Applications provide the web services devices via HTTP URLs – HTTP REST APIs. When accessing the APIs, the Web Server and edge device communicate and exchange data via HTTP.