
Date 5/5/2023

CSM3313

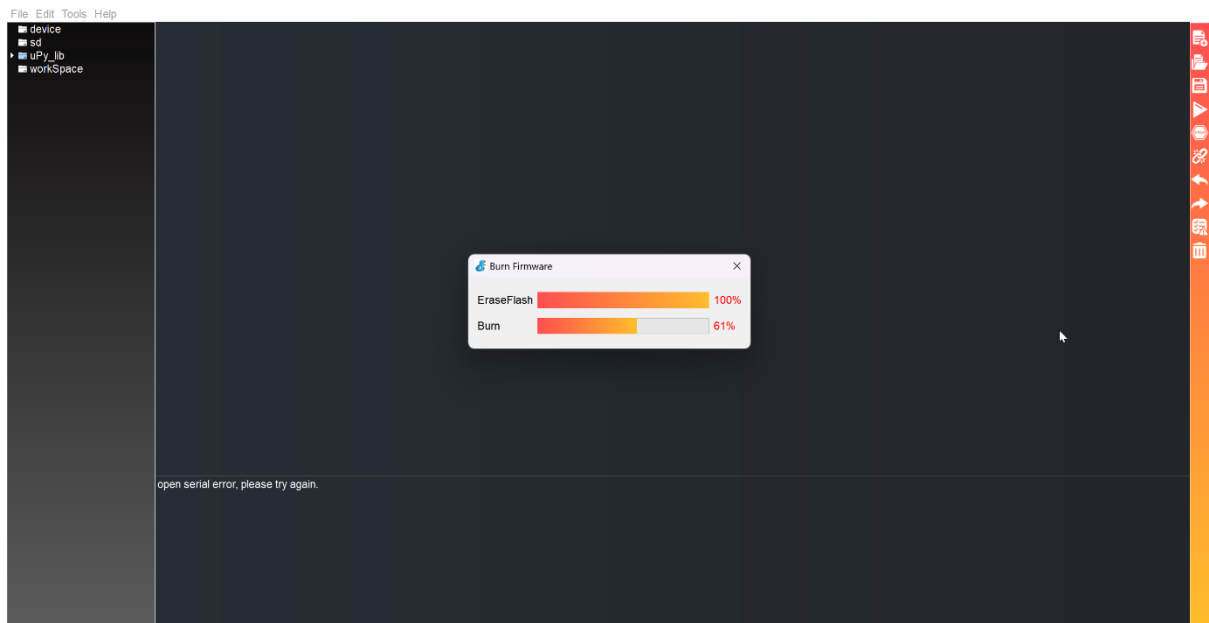
IOT COMPUTING

DR AHMAD SHUKRI BIN MOHD NOOR

OMAR ISMAIL ABDJALEEL ALOMORY

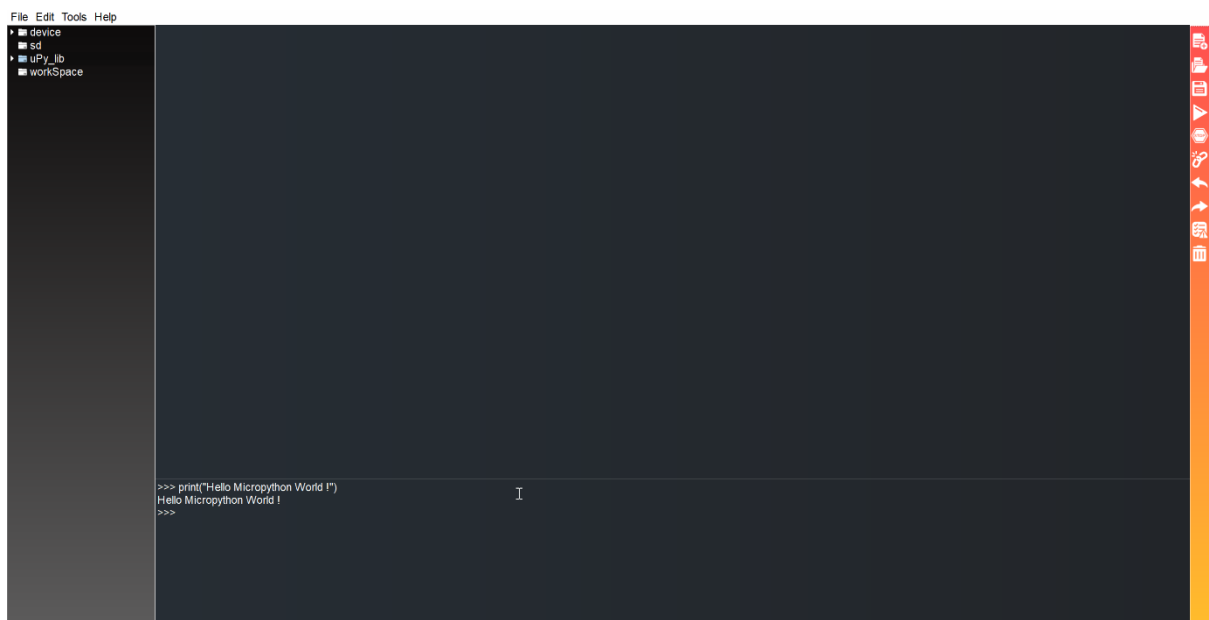
LAB 3(MP2)

LAB 1 GETTING STARTED WITH MICROPYTHON:



Installation of micropython, driver, and bin file are completed.

LAB 2 BASIC MICROPYTHON PROGRAMMING:



Arithmetic operation:

```
>>> 3+5
8
>>> 6-5
1
>>> 8*9
72
>>> 20/10
2.0
```

Boolean expression(logical operation):

```
>>> 2==5
False
>>> 4==4
True
>>> 69874 !=65
True
>>> 3>2
True
>>> |
```

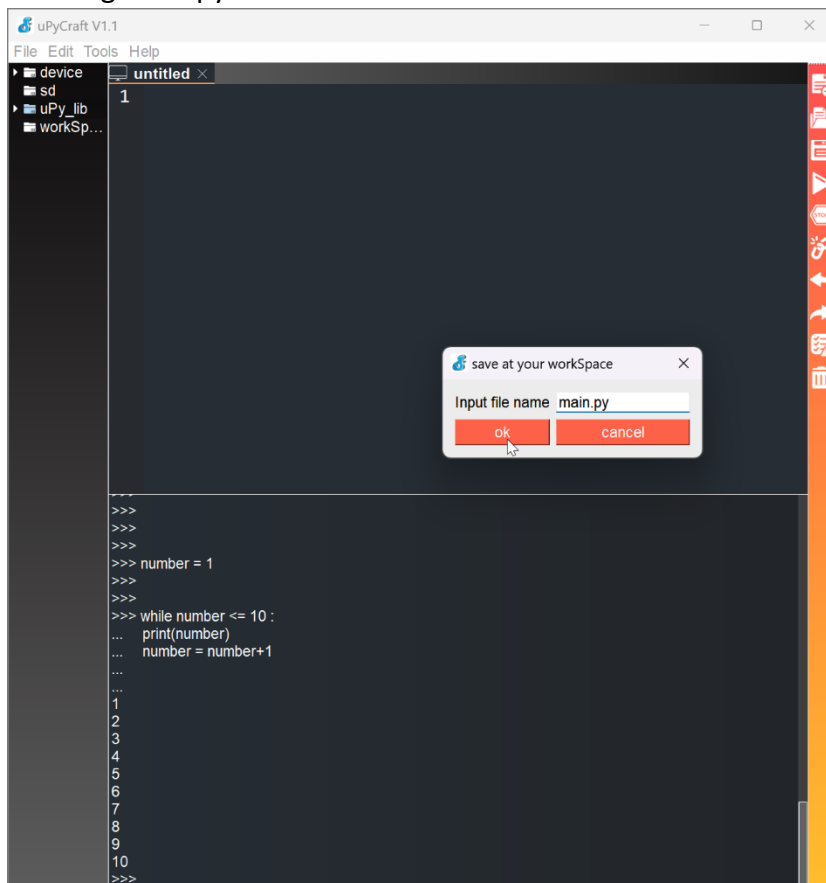
Assigning values and exploring micropython datatypes:

```
>>>
>>> a = 10
>>> b = 12
>>> c = 20.6
>>> text = 'abcdef'
>>> d = True
>>>
>>>
>>> type(a)
<class 'int'>
>>> type(b)
<class 'int'>
>>> type(b)
<class 'int'>
>>> type(c)
<class 'float'>
>>> type(text)
<class 'str'>
>>> type(d)
<class 'bool'>
>>>
```

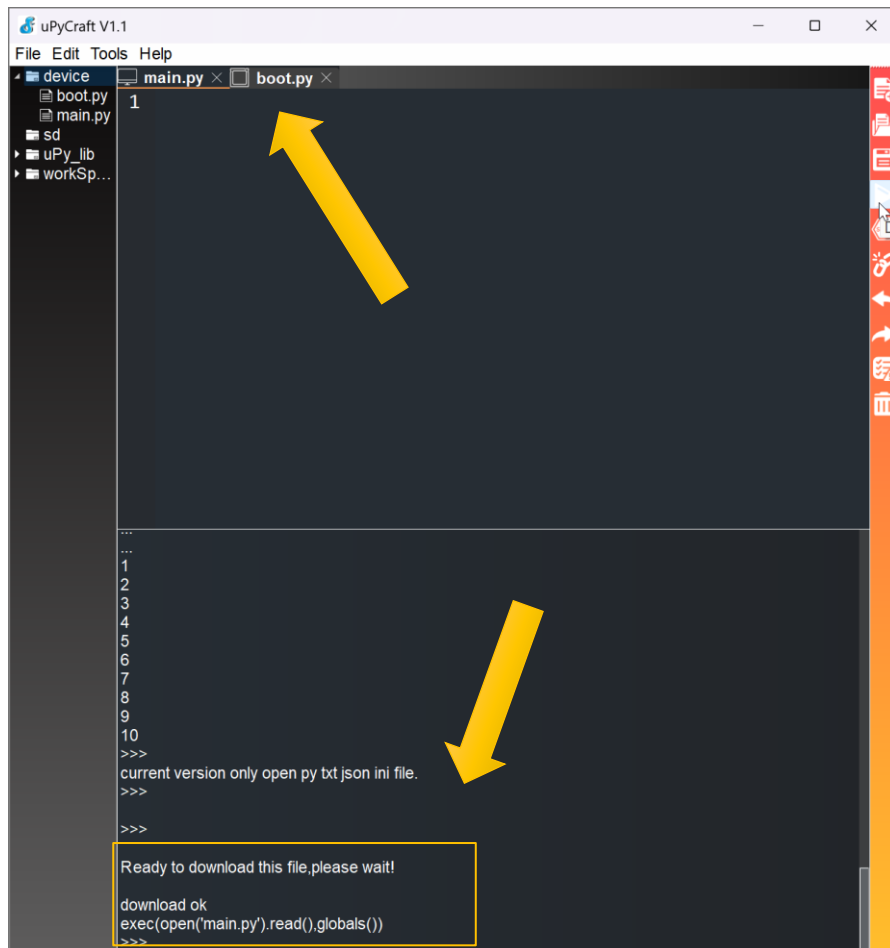
“while” loops in Micropython:

```
>>> number = 1
>>>
>>>
>>> while number <= 10 :
...   print(number)
...   number = number+1
...
...
1
2
3
4
5
6
7
8
9
10
>>>
```

Creating main.py in the board:



- After that, you should see the following in your uPyCraft IDE (the boot.py file in your device and a new tab with the main.py file)
- After clicking Download and run, the device directory should now load the main.py file. Your ESP has the file main.py stored.

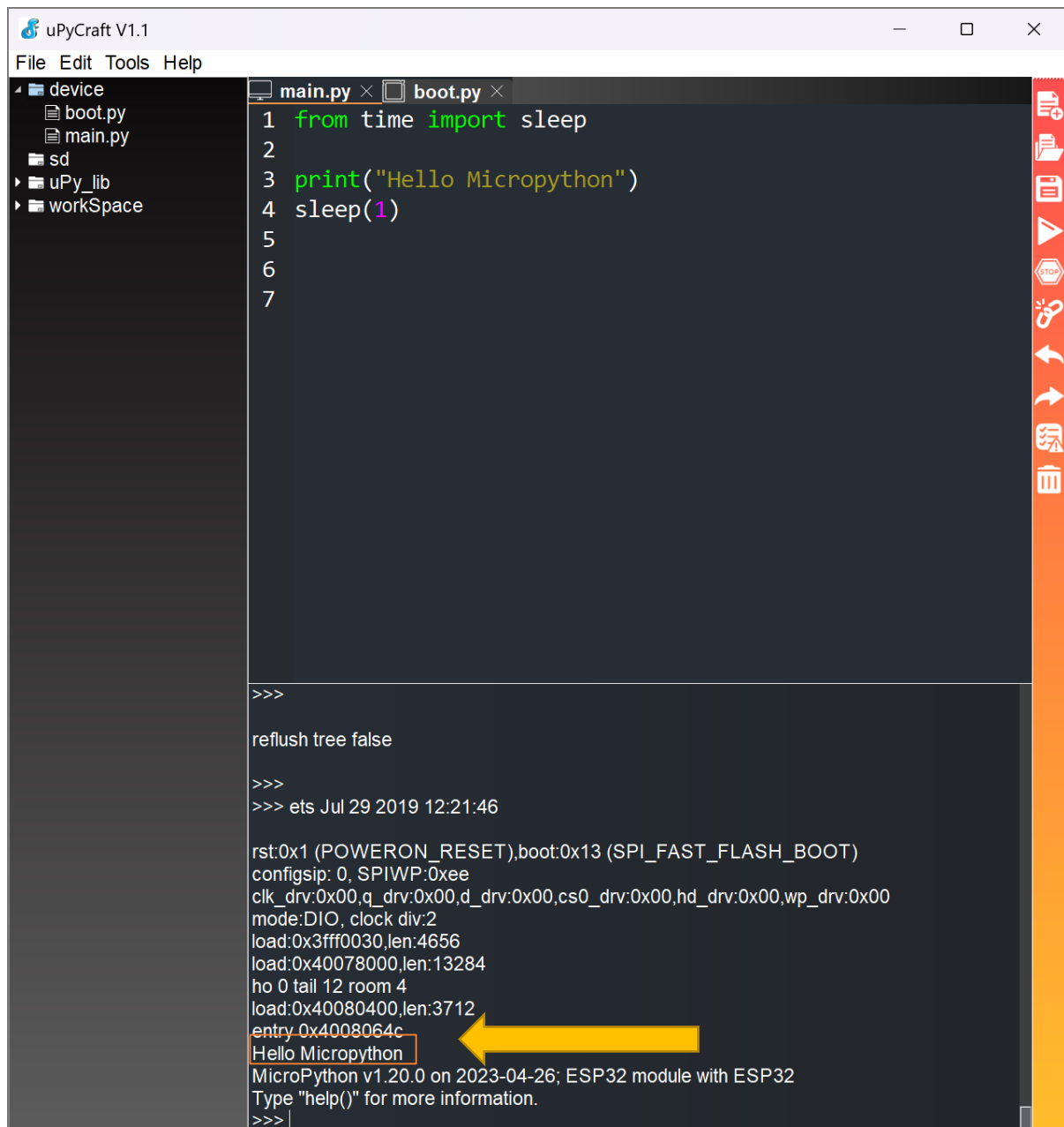


After clicking the reset button (EN) in the ESP32 the output looks like this.

```
>>> ets Jul 29 2019 12:21:46

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:4540
ho 0 tail 12 room 4
load:0x40078000,len:12344
ho 0 tail 12 room 4
load:0x40080400,len:4124
entry 0x40080680
MicroPython v1.19.1 on 2022-06-18; ESP32 module with ESP32
Type "help()" for more information.
>>>
>>>
```

Print and sleep methods:



The screenshot shows the uPyCraft V1.1 IDE interface. On the left is a file explorer with a tree view containing 'device', 'boot.py', 'main.py', 'sd', 'uPy_lib', and 'workSpace'. The main editor area has two tabs: 'main.py' and 'boot.py'. The 'main.py' tab is active and contains the following code:

```
1 from time import sleep
2
3 print("Hello Micropython")
4 sleep(1)
5
6
7
```

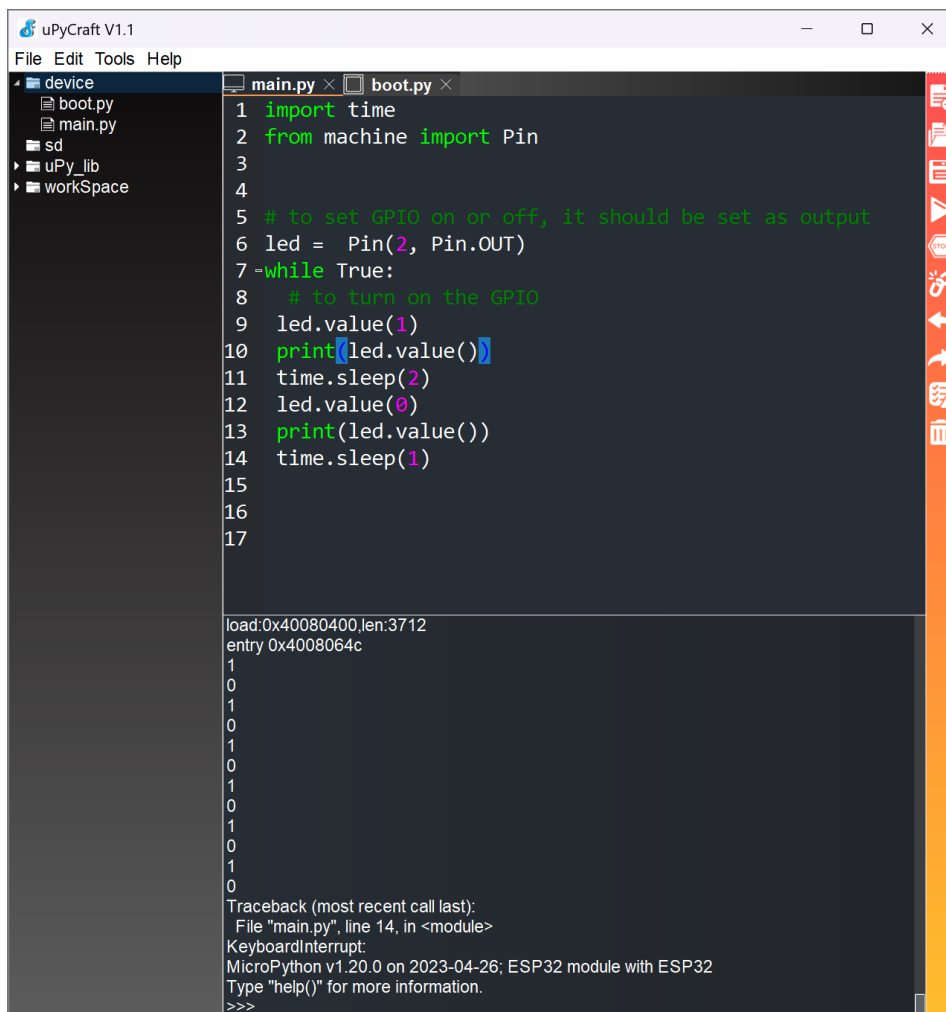
Below the code editor is a console window showing the execution output. The output includes system boot information and the printed message 'Hello Micropython', which is highlighted with a yellow box and a yellow arrow pointing to it from the right.

```
>>>
reflush tree false
>>>
>>> ets Jul 29 2019 12:21:46

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:4656
load:0x40078000,len:13284
ho 0 tail 12 room 4
load:0x40080400,len:3712
entry 0x4008064c
Hello Micropython
Micropython v1.20.0 on 2023-04-26; ESP32 module with ESP32
Type "help()" for more information.
>>>
```

LAB 3 ESP32 PROGRAMMING:

Turning on the LED in the ESP32 board(internal)



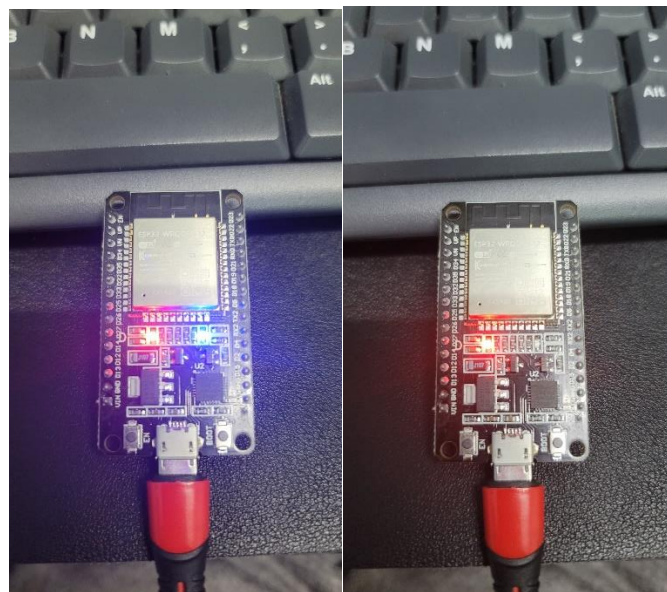
```
uPyCraft V1.1
File Edit Tools Help

device
├── boot.py
├── main.py
├── sd
├── uPy_lib
└── workspace

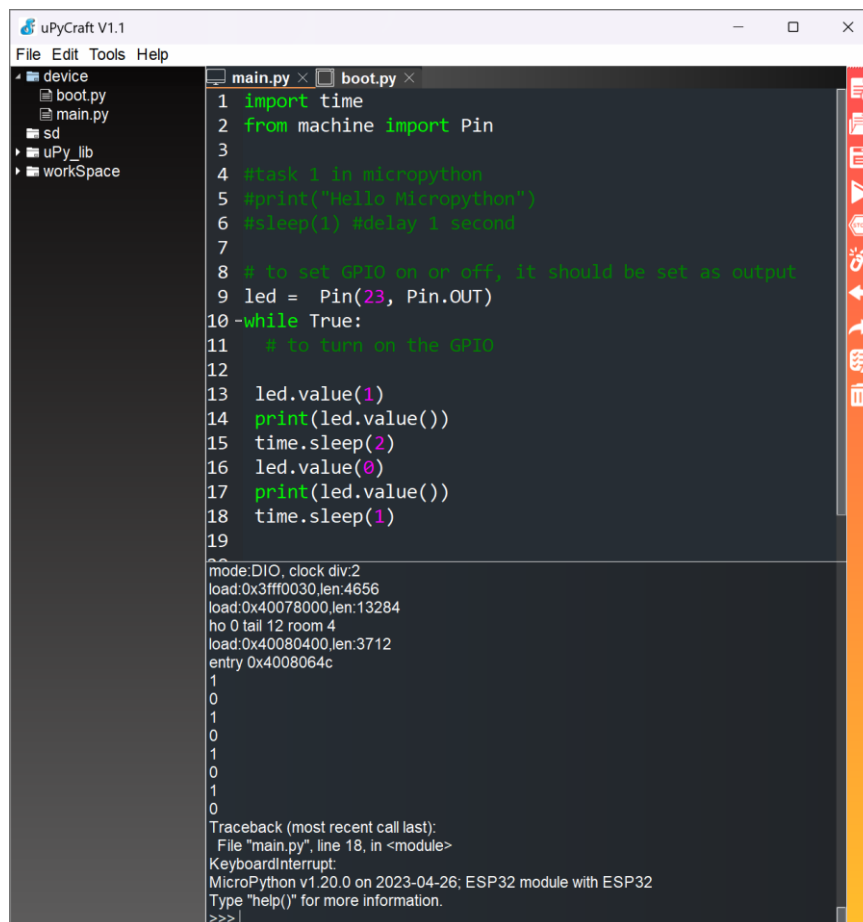
main.py x boot.py x
1 import time
2 from machine import Pin
3
4
5 # to set GPIO on or off, it should be set as output
6 led = Pin(2, Pin.OUT)
7 while True:
8     # to turn on the GPIO
9     led.value(1)
10    print(led.value())
11    time.sleep(2)
12    led.value(0)
13    print(led.value())
14    time.sleep(1)
15
16
17

load:0x40080400,len:3712
entry 0x4008064c
1
0
1
0
1
0
1
0
1
0
1
0
0
Traceback (most recent call last):
  File "main.py", line 14, in <module>
KeyboardInterrupt:
MicroPython v1.20.0 on 2023-04-26; ESP32 module with ESP32
Type "help()" for more information.
>>>
```

Output



Blinking LED (external):

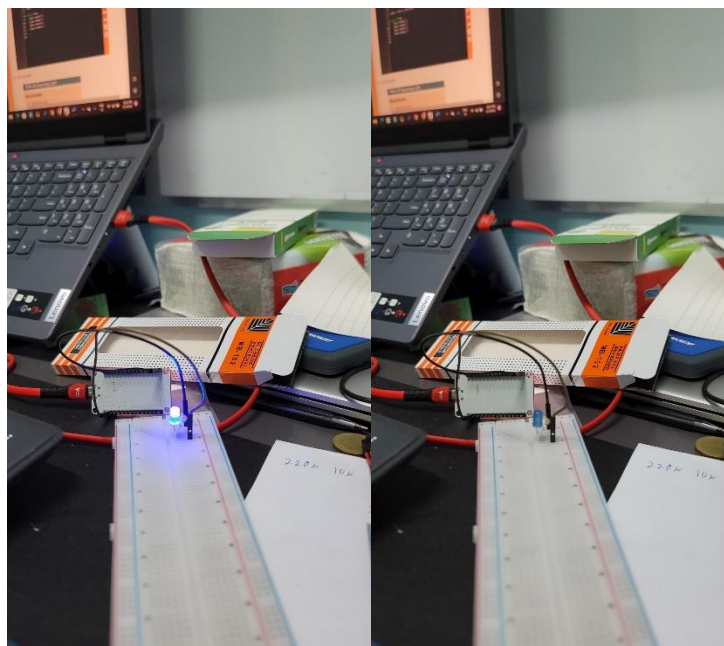


```
uPyCraft V1.1
File Edit Tools Help

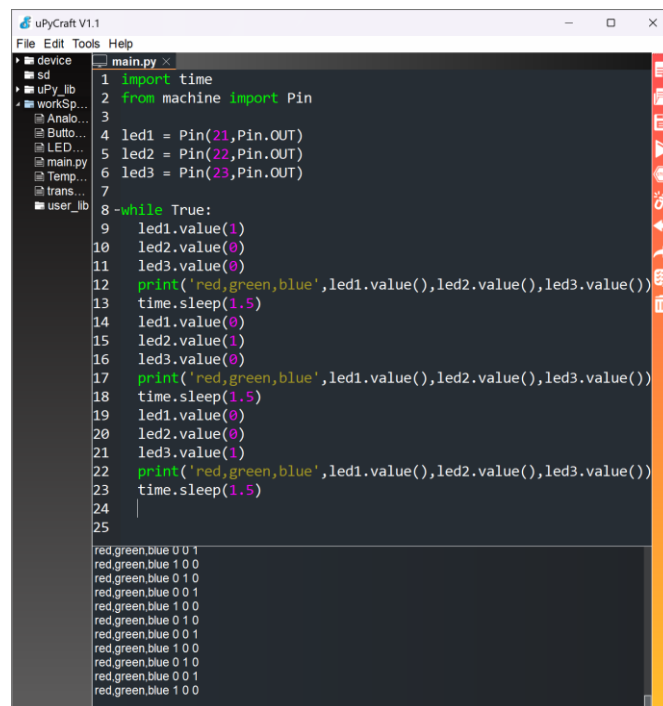
device
├─ boot.py
├─ main.py
├─ sd
├─ uPy_lib
└─ workSpace

main.py x boot.py x
1 import time
2 from machine import Pin
3
4 #task 1 in micropython
5 #print("Hello Micropython")
6 #sleep(1) #delay 1 second
7
8 # to set GPIO on or off, it should be set as output
9 led = Pin(23, Pin.OUT)
10 -while True:
11     # to turn on the GPIO
12
13     led.value(1)
14     print(led.value())
15     time.sleep(2)
16     led.value(0)
17     print(led.value())
18     time.sleep(1)
19
mode:DIO, clock div:2
load:0x3fff0030,len:4656
load:0x40078000,len:13284
ho 0 tail 12 room 4
load:0x40080400,len:3712
entry 0x4008064c
1
0
1
0
1
0
1
0
0
Traceback (most recent call last):
  File "main.py", line 18, in <module>
KeyboardInterrupt:
MicroPython v1.20.0 on 2023-04-26; ESP32 module with ESP32
Type "help()" for more information.
>>>
```

Output:



RGB LED:

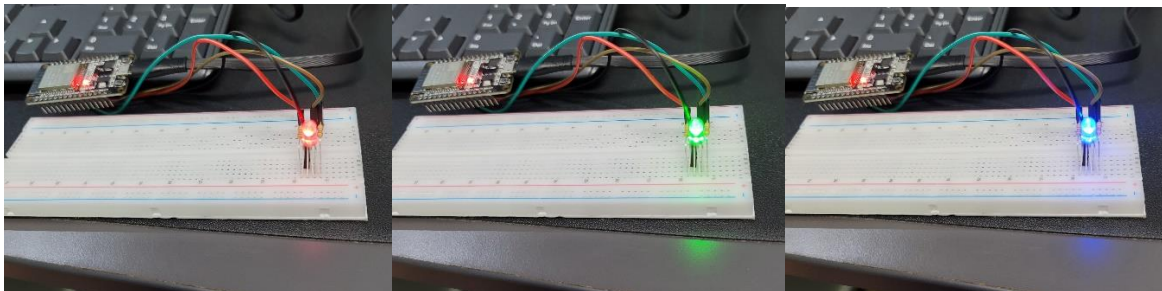


```
uPyCraft V1.1
File Edit Tools Help

main.py x
1 import time
2 from machine import Pin
3
4 led1 = Pin(21,Pin.OUT)
5 led2 = Pin(22,Pin.OUT)
6 led3 = Pin(23,Pin.OUT)
7
8 -while True:
9     led1.value(1)
10    led2.value(0)
11    led3.value(0)
12    print('red,green,blue',led1.value(),led2.value(),led3.value())
13    time.sleep(1.5)
14    led1.value(0)
15    led2.value(1)
16    led3.value(0)
17    print('red,green,blue',led1.value(),led2.value(),led3.value())
18    time.sleep(1.5)
19    led1.value(0)
20    led2.value(0)
21    led3.value(1)
22    print('red,green,blue',led1.value(),led2.value(),led3.value())
23    time.sleep(1.5)
24
25

red,green,blue 0 0 1
red,green,blue 1 0 0
red,green,blue 0 1 0
red,green,blue 0 0 1
red,green,blue 1 0 0
red,green,blue 0 1 0
red,green,blue 0 0 1
red,green,blue 1 0 0
red,green,blue 0 1 0
red,green,blue 0 0 1
red,green,blue 1 0 0
```

Output



Button interaction with ESP32:

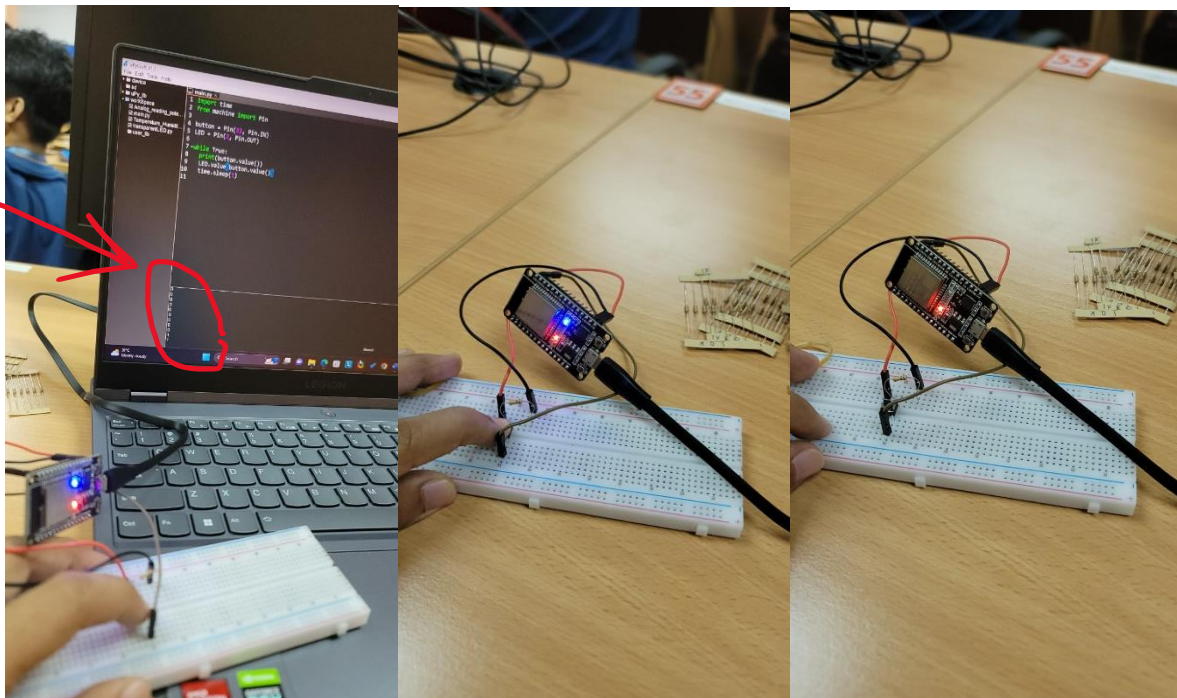
```
uPyCraft V1.1
File Edit Tools Help

device
├── sd
├── uPy_lib
└── workspace
    ├── Analog_reading_pote...
    ├── ButtonDigitalInput.py
    ├── LEDButton.py
    ├── main.py
    ├── Temperature_Humidit...
    ├── transparentLED.py
    └── user_lib

ButtonDigitalInput.py
1 import time
2 from machine import Pin
3
4 button = Pin(23, Pin.IN)
5 LED = Pin(2, Pin.OUT)
6
7 while True:
8     print(button.value())
9     LED.value(button.value())
10    time.sleep(1)
11
12
```

Output

When the button is clicked the value = 1, otherwise 0, plus if it 1 the internal LED will blink as it shown in the picture below.



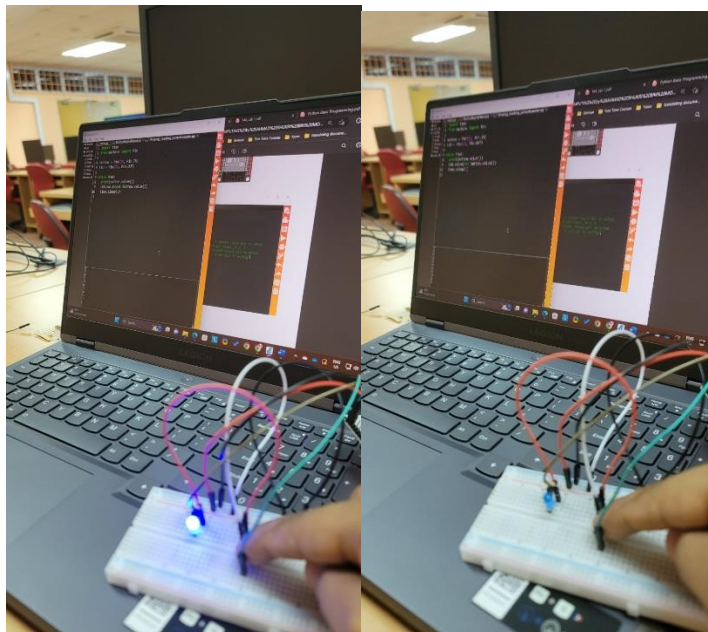
Button interaction to light LED in ESP32:

```
uPyCraft V1.1
File Edit Tools Help
└─ device
  └─ sd
  └─ uPy_lib
  └─ workSpace
    └─ Analog_rea...
    └─ ButtonDigita...
    └─ LEDButton.py
    └─ main.py
    └─ RGB_LED...
    └─ Temperatur...
    └─ transparent...
    └─ user_lib

main.py × ButtonDigitalInput.py × LEDButton.py ×
1 import time
2 from machine import Pin
3
4 button = Pin(23, Pin.IN)
5 LED = Pin(22, Pin.OUT)
6
7 while True:
8     print(button.value())
9     LED.value(not button.value())
10    time.sleep(1)
11
12
```

Output

When the button is clicked the LED any color will light up, otherwise it will be down.

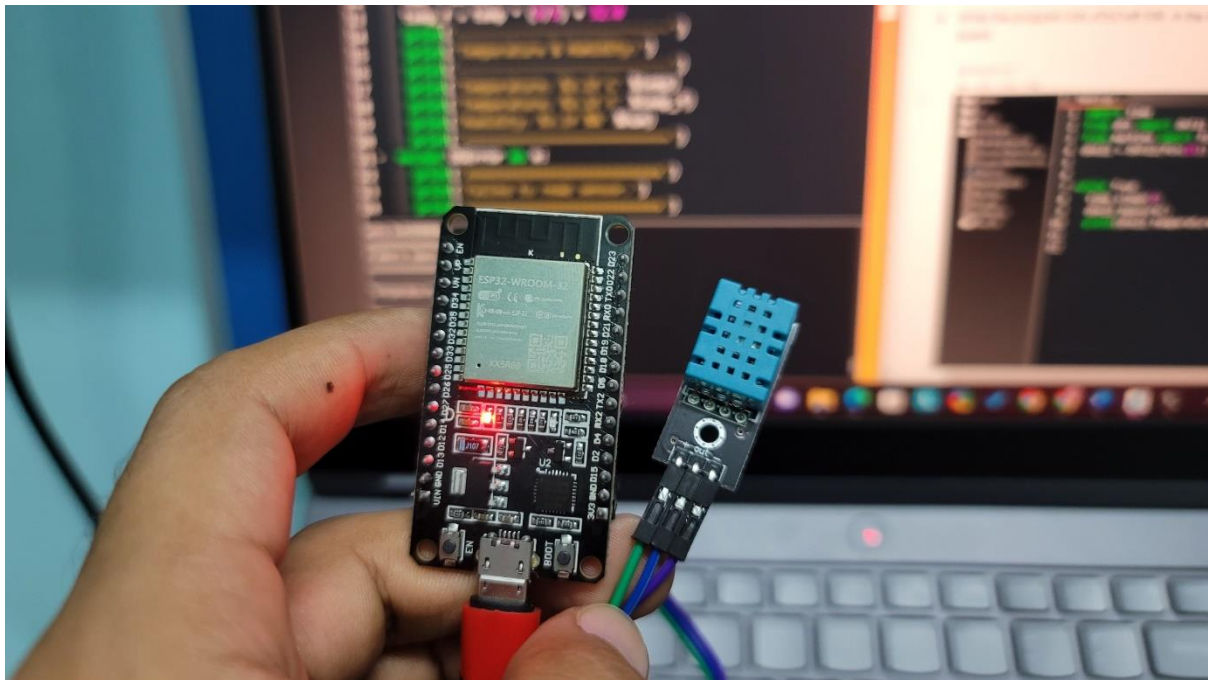


Temperature and Humidity measurement:

```
File Edit Tools Help
main.py x
1 from machine import Pin
2 from time import sleep
3 import dht
4
5 sensor = dht.DHT11(Pin(14))
6
7 -while True:
8 -   try:
9       sleep(3)
10      sensor.measure()
11      temp = sensor.temperature()
12      hum = sensor.humidity()
13      temp_f = temp * (9/5) + 32.0
14      print('-----')
15      print('Temperature & Humidity:')
16      print('-----')
17      print('Temperature: %3.1f C' %temp)
18      print('Temperature: %3.1f F' %temp_f)
19      print('Humidity: %3.1f %%' %hum)
20      print('=====')
21 -   except OSError as e:
22       print('xxxxxxxxxxxxxxxxxxxxxxxxxxxx')
23       print('Failed to read sensor.')
24       print('xxxxxxxxxxxxxxxxxxxxxxxxxxxx')

entry 0x4008064c
xxxxxxxxxxxxxxxxxxxxxxxxxxxx
Failed to read sensor.
xxxxxxxxxxxxxxxxxxxxxxxxxxxx

-----
Temperature & Humidity:
-----
Temperature: 31.0 C
Temperature: 87.8 F
Humidity: 75.0 %
=====
Temperature & Humidity:
```



LED fade:

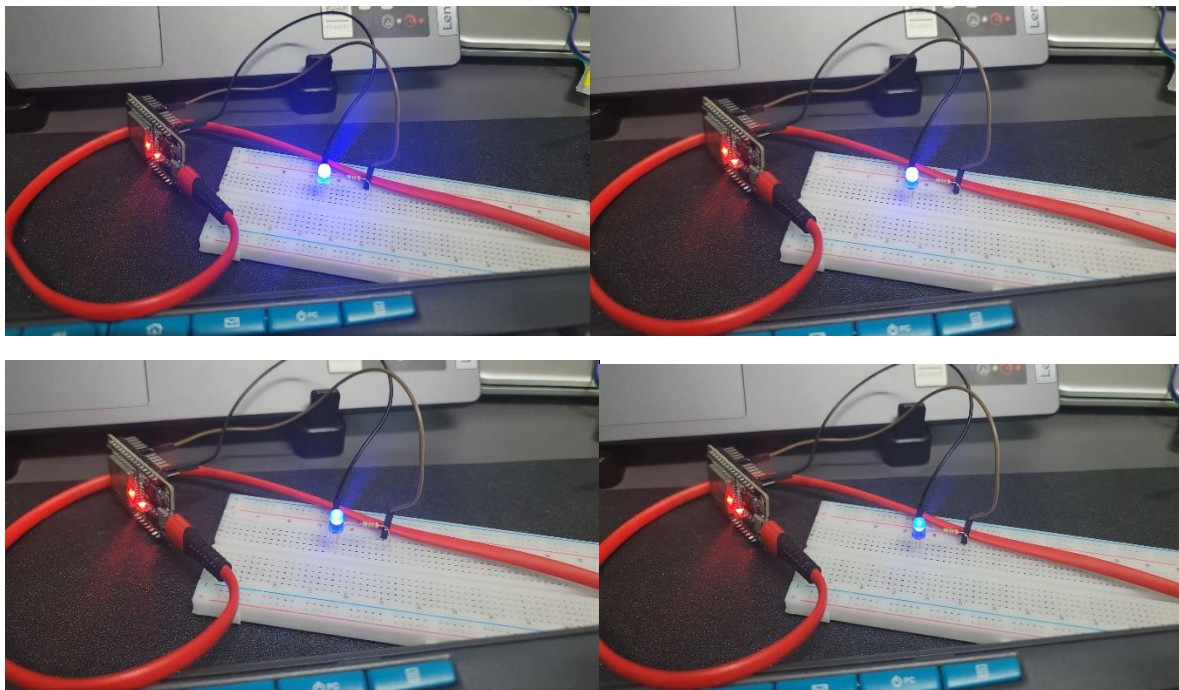
```
File Edit Tools Help
main.py x
device
sd
uPy_lib
workSp...
Analo...
main.py
Temp...
user_lib

1 from machine import Pin, PWM
2 from time import sleep
3
4 frequency = 5000
5 led = PWM(Pin(18), frequency)
6
7 -while True:
8 -   for duty_cycle in range(0,1024):
9       led.duty(duty_cycle)
10      sleep(0.005)

Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<string>", line 10, in <module>
KeyboardInterrupt:
>>>
>>>
>>> ets Jul 29 2019 12:21:46

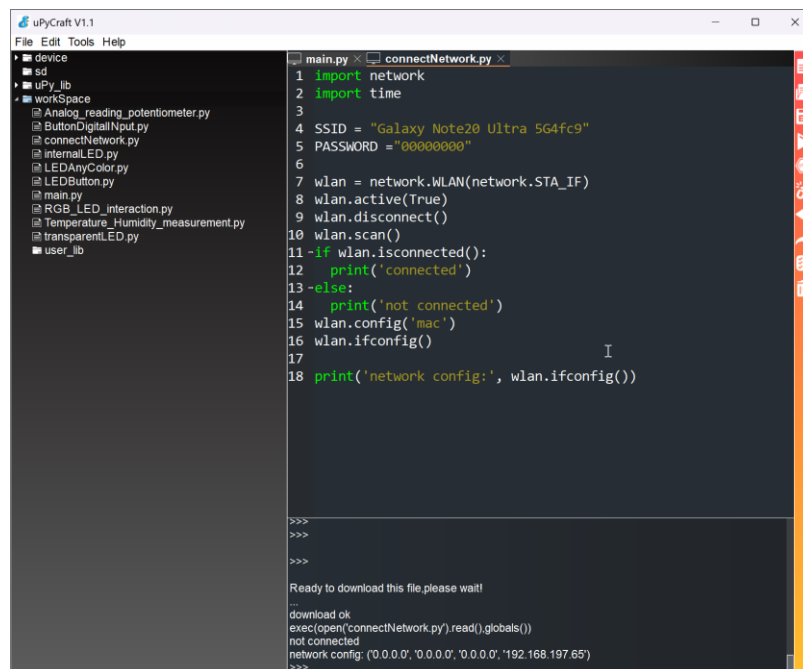
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:4656
load:0x40078000,len:13284
ho 0 tail 12 room 4
load:0x40080400,len:3712
entry 0x4008064c
```

Output(max, medium-high. medium-low, low)



LAB 2 IOT PROGRAMMING:

ESP32 connect to Wi-Fi.



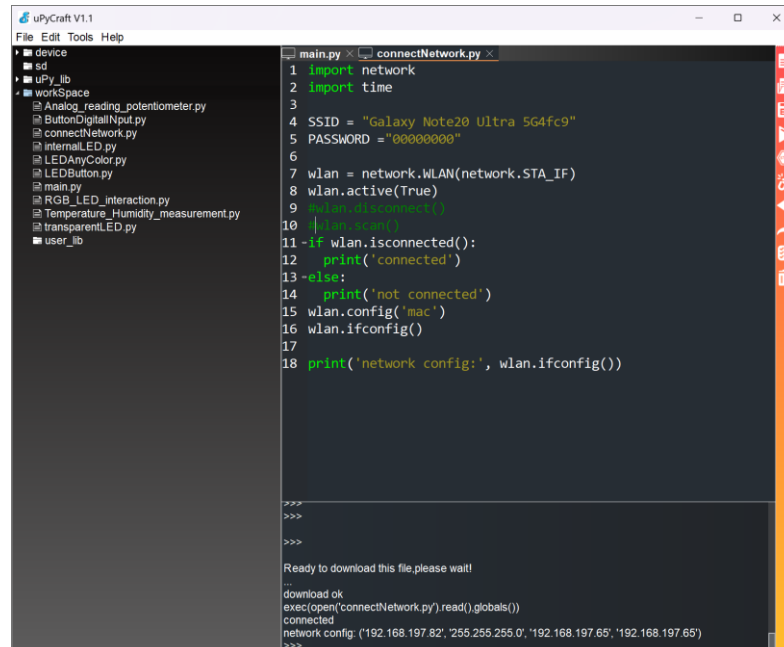
```
uPyCraft V1.1
File Edit Tools Help
device
sd
uPy_lib
workspace
  Analog_reading_potentiometer.py
  ButtonDigitalNput.py
  connectNetwork.py
  internalLED.py
  LEDAnyColor.py
  LEDButton.py
  main.py
  RGB_LED_interaction.py
  Temperature_Humidity_measurement.py
  transparentLED.py
  user_lib

main.py x connectNetwork.py x
1 import network
2 import time
3
4 SSID = "Galaxy Note20 Ultra 5G4fc9"
5 PASSWORD = "00000000"
6
7 wlan = network.WLAN(network.STA_IF)
8 wlan.active(True)
9 wlan.disconnect()
10 wlan.scan()
11 -if wlan.isconnected():
12     print('connected')
13 -else:
14     print('not connected')
15 wlan.config('mac')
16 wlan.ifconfig()
17
18 print('network config:', wlan.ifconfig())

>>>
>>>
>>>

Ready to download this file, please wait!
...
download ok
exec(open('connectNetwork.py').read()).globals()
not connected
network config: ('0.0.0.0', '0.0.0.0', '0.0.0.0', '192.168.197.65')
>>>
```

If we comment the `wlan.disconnect()` and `wlan.scan()`, we can get more parameters value such as subnet mask and other.



```
uPyCraft V1.1
File Edit Tools Help
device
sd
uPy_lib
workspace
  Analog_reading_potentiometer.py
  ButtonDigitalNput.py
  connectNetwork.py
  internalLED.py
  LEDAnyColor.py
  LEDButton.py
  main.py
  RGB_LED_interaction.py
  Temperature_Humidity_measurement.py
  transparentLED.py
  user_lib

main.py x connectNetwork.py x
1 import network
2 import time
3
4 SSID = "Galaxy Note20 Ultra 5G4fc9"
5 PASSWORD = "00000000"
6
7 wlan = network.WLAN(network.STA_IF)
8 wlan.active(True)
9 #wlan.disconnect()
10 #wlan.scan()
11 -if wlan.isconnected():
12     print('connected')
13 -else:
14     print('not connected')
15 wlan.config('mac')
16 wlan.ifconfig()
17
18 print('network config:', wlan.ifconfig())

>>>
>>>
>>>

Ready to download this file, please wait!
...
download ok
exec(open('connectNetwork.py').read()).globals()
connected
network config: ('192.168.197.82', '255.255.255.0', '192.168.197.65', '192.168.197.65')
>>>
```

ESP32 MQTT

Registration Completed Successfully.

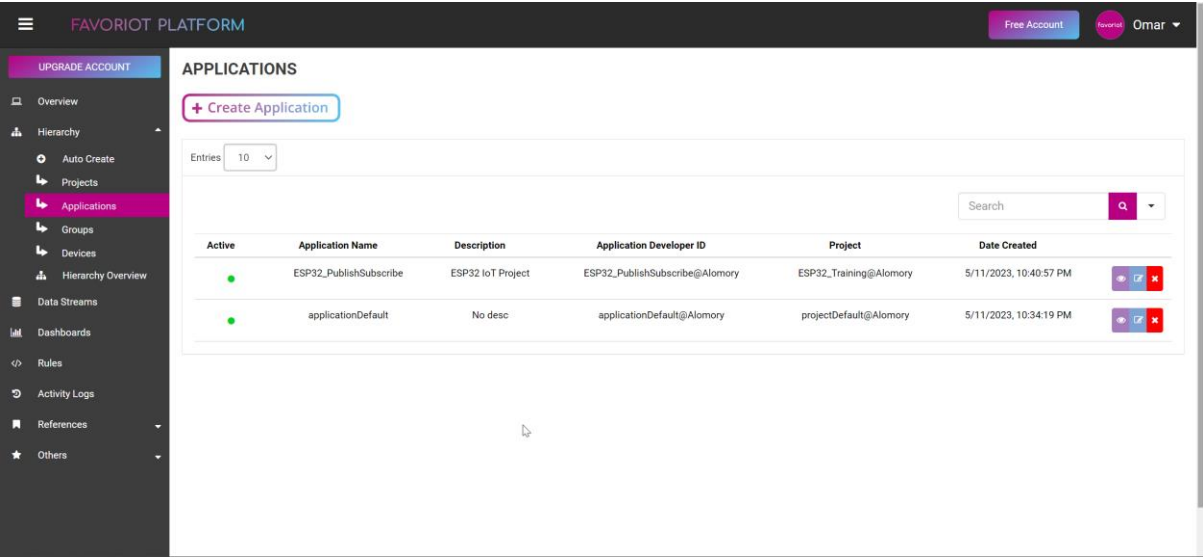
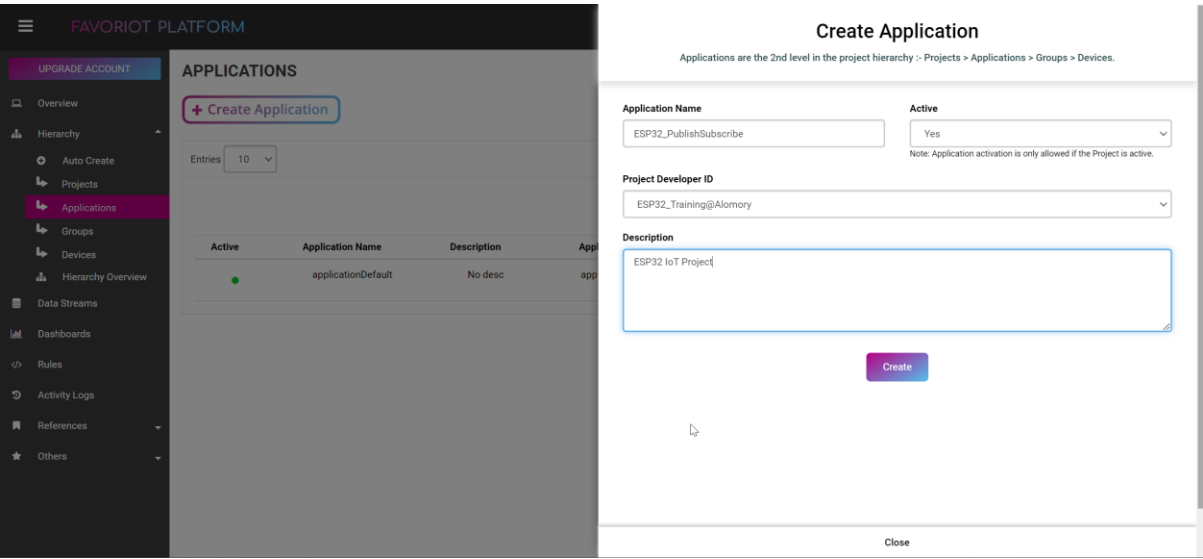
The screenshot shows the 'OVERVIEW' page of the FAVORIOT PLATFORM. The left sidebar contains navigation links: Overview, Hierarchy, Data Streams, Dashboards, Rules, Activity Logs, References, and Others. The main content area displays several metrics: 1 Project, 1 Applications, 1 Groups, and 1 Devices. Below these, there are sections for 'Device Details' (showing 1), 'Total Data Stream' (showing 0), and 'Latest Data Stream' (showing 'No data stream has been recorded'). On the right, there are sections for 'API Limit' (showing 0% used, 0 out of 500) and 'SMS Service' (showing 0% Not Subscribed with a 'SUBSCRIBE NOW' button).

adding new project.

The screenshot shows the 'PROJECTS' page of the FAVORIOT PLATFORM. The left sidebar is the same as the overview page. The main content area shows a table with columns: Active, Project Name, and Description. The table has one row with a green dot in the Active column, 'projectDefault' in the Project Name column, and 'No desc' in the Description column. A '+ Create Project' button is visible. To the right, the 'Create Project' modal is open, showing a form with fields for Project Name (ESP32_Training), Active (Yes), and Description (IoT Project). A 'Create' button is at the bottom of the modal.

The screenshot shows the 'PROJECTS' page of the FAVORIOT PLATFORM after a successful creation. A green banner at the top right says 'Success Project successfully created'. The main content area shows a table with columns: Active, Project Name, Description, Project Developer Id, and Date Created. The table has two rows: one for 'ESP32_Training' (IoT Project, ESP32_Training@Alomory, 5/11/2023, 10:38:29 PM) and one for 'projectDefault' (No desc, projectDefault@Alomory, 5/11/2023, 10:34:19 PM). A '+ Create Project' button is visible. A search bar is also present.

Create application.



Create Group.

FAVORIOT PLATFORM

UPGRADE ACCOUNT

Overview

Hierarchy

- Auto Create
- Projects
- Applications
- Groups
- Devices
- Hierarchy Overview

Data Streams

Dashboards

Rules

Activity Logs

References

Others

GROUPS

Create Group

Entries 10

Active	Group Name	Description	Group Developer ID
	groupDefault	No desc	groupDefault@Alomory

Create Group

Groups are the 3rd level in the project hierarchy :- Projects > Applications > Groups > Devices.

Group Name

ESP32_Training

Active

Yes

Application Developer ID

ESP32_PublishSubscribe@Alomory

Description

ESP32 IoT Project

Create

Close

FAVORIOT PLATFORM

UPGRADE ACCOUNT

Overview

Hierarchy

- Auto Create
- Projects
- Applications
- Groups
- Devices
- Hierarchy Overview

Data Streams

Dashboards

Rules

Activity Logs

References

Others

GROUPS

Create Group

Entries 10

Search

Active	Group Name	Description	Group Developer ID	Application	Date Created	
	ESP32_Training	ESP32 IoT Project	ESP32_Training@Alomory	ESP32_PublishSubscribe@Alomory	5/11/2023, 10:43:07 PM	
	groupDefault	No desc	groupDefault@Alomory	applicationDefault@Alomory	5/11/2023, 10:34:19 PM	

Success
Group successfully created

Create Device.

FAVORIOT PLATFORM

UPGRADE ACCOUNT

Overview

Hierarchy

Auto Create

Projects

Applications

Groups

Devices

Hierarchy Overview

Data Streams

Dashboards

Rules

Activity Logs

References

Others

DEVICES

+ Create Device

View on Map

Entries 10

Active	Connected	Device Name	Description
<div></div>	<div></div>	deviceDefault	No desc

Create Device

Devices are the 4th level in the project hierarchy > Projects > Applications > Groups > Devices.

Device Name

ESP32

Active

Yes

Note: Device activation is only allowed if the Group is active.

Group Developer ID

ESP32_Training@Alomory

Description

ESP32 IoT Project

Device Type

Other

Sensor Type (Separate with comma for multiple sensors)

Other


Time zone

Kuala Lumpur, Singapore

Move marker or enter coordinate to select location (optional)

Map

Satellite



Close

FAVORIOT PLATFORM

UPGRADE ACCOUNT

- Overview
- Hierarchy
- Auto Create
- Projects
- Applications
- Groups
- Devices**
- Hierarchy Overview
- Data Streams
- Dashboards
- Rules
- Activity Logs
- References
- Others

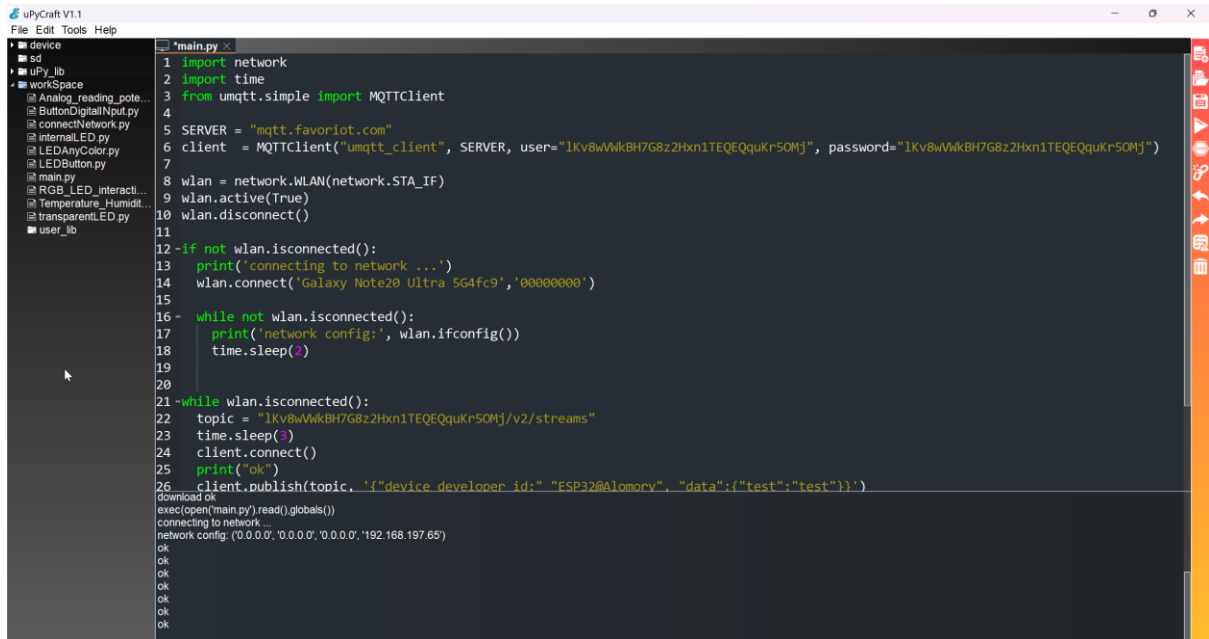
DEVICES

+ Create Device
View on Map ☐

Entries 10
Search

Active	Connected	Device Name	Description	Device Developer ID	Group	Date Created
●	●	ESP32	ESP32 IoT Project	ESP32@Alomory	ESP32_Training@Alomory	5/11/2023, 10:46:57 PM

Program your ESP32 as below and edit the program. After that download the program into ESP32.

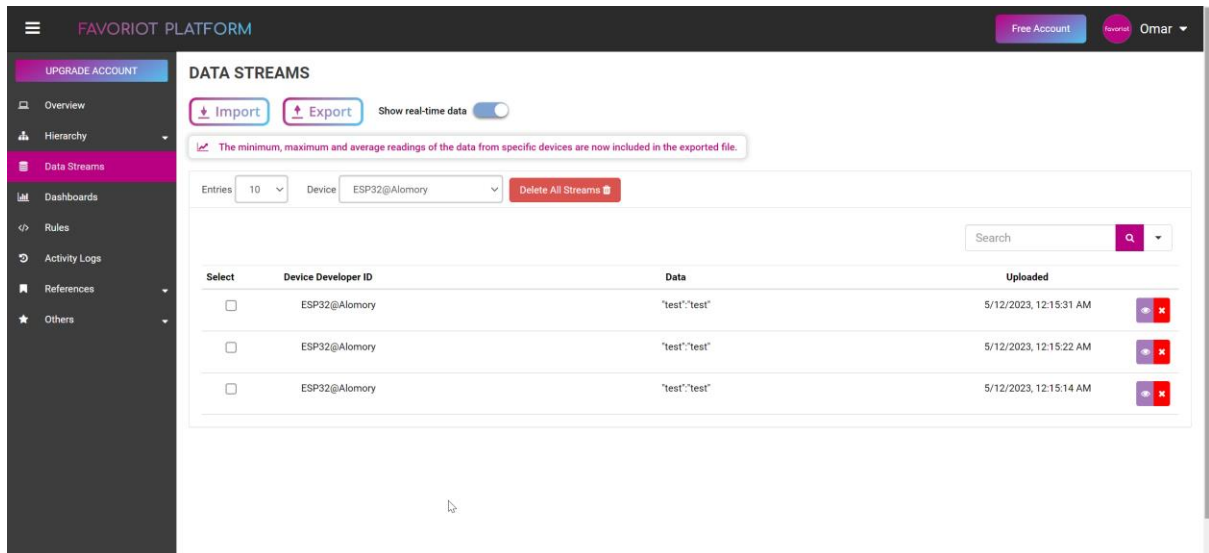


```
uPyCraft V1.1
File Edit Tools Help

device
sd
uPy lib
workspace
  Analog_reading_pote...
  ButtonDigitalInput.py
  connectNetwork.py
  internal_LED.py
  LEDAnyColor.py
  LEDButton.py
  main.py
  RGB_LED_interacti...
  Temperature_Humidit...
  transparentLED.py
  user_lib

main.py
1 import network
2 import time
3 from umqtt.simple import MQTTClient
4
5 SERVER = "mqtt.favotiot.com"
6 client = MQTTClient("umqtt_client", SERVER, user="lkv8wVwKbH7G8z2Hxn1TEQEQuKr5OMj", password="lkv8wVwKbH7G8z2Hxn1TEQEQuKr5OMj")
7
8 wlan = network.WLAN(network.STA_IF)
9 wlan.active(True)
10 wlan.disconnect()
11
12 if not wlan.isconnected():
13     print('connecting to network ...')
14     wlan.connect('Galaxy Note20 Ultra 5G4fc9','00000000')
15
16 while not wlan.isconnected():
17     print('network config:', wlan.ifconfig())
18     time.sleep(2)
19
20
21 while wlan.isconnected():
22     topic = "lkv8wVwKbH7G8z2Hxn1TEQEQuKr5OMj/v2/streams"
23     time.sleep(3)
24     client.connect()
25     print("ok")
26     client.publish(topic, '{"device_developer_id": "ESP32@Alomory", "data":{"test":"test"}}')
27
28 download ok
29 exec(open('main.py').read()).globals()
30 connecting to network ...
31 network config: ('0.0.0.0', '0.0.0.0', '0.0.0.0', '192.168.197.65')
32 ok
33 ok
34 ok
35 ok
36 ok
37 ok
```

The dummy “test” data will be published to MQTT Broker Favotiot every 5 seconds. You can see your data at Data Stream tab



FAVORIOT PLATFORM

Free Account Omar

UPGRADE ACCOUNT

Overview

Hierarchy

Data Streams

Dashboards

Rules

Activity Logs

References

Others

DATA STREAMS

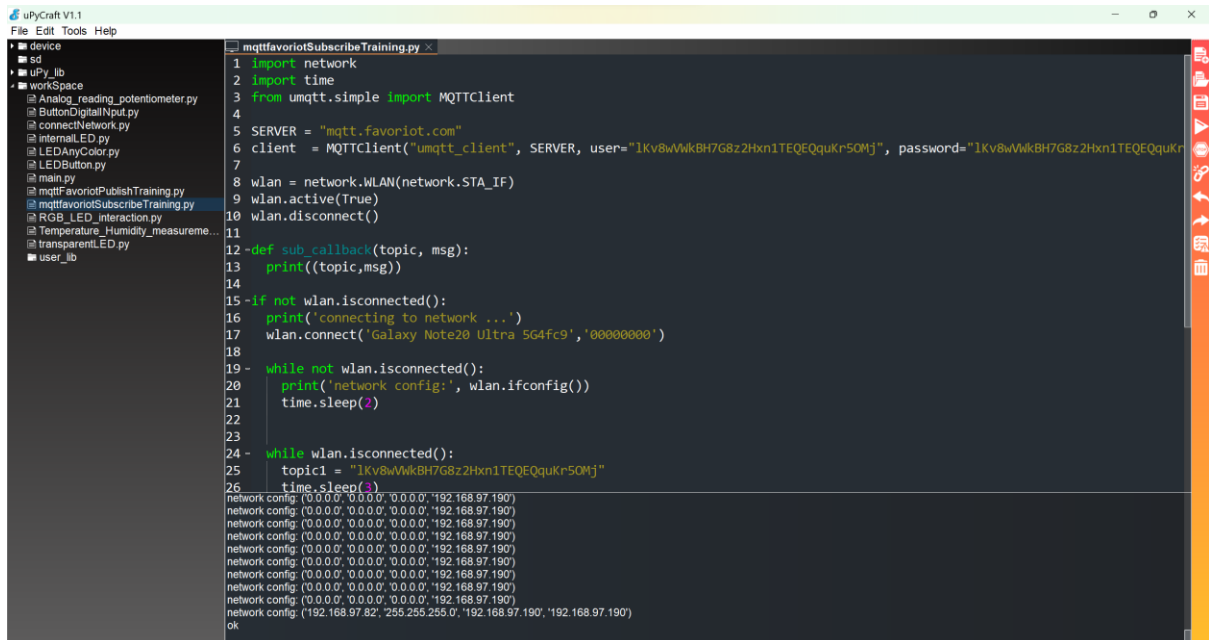
Import Export Show real-time data

The minimum, maximum and average readings of the data from specific devices are now included in the exported file.

Entries 10 Device ESP32@Alomory Delete All Streams

Select	Device Developer ID	Data	Uploaded
<input type="checkbox"/>	ESP32@Alomory	"test":"test"	5/12/2023, 12:15:31 AM
<input type="checkbox"/>	ESP32@Alomory	"test":"test"	5/12/2023, 12:15:22 AM
<input type="checkbox"/>	ESP32@Alomory	"test":"test"	5/12/2023, 12:15:14 AM

Subscriber file



```
uPyCraft V1.1
File Edit Tools Help

device
sd
uPyLib
workspace
Analog_reading_potentiometer.py
ButtonDigitalInput.py
connectNetwork.py
internalLED.py
LEDAnyColor.py
LEDButton.py
main.py
mqttFavoriotPublishTraining.py
mqttFavoriotSubscribeTraining.py
RGB_LED_interaction.py
Temperature_Humidity_measureme...
transparentLED.py
user_lib

mqttFavoriotSubscribeTraining.py
1 import network
2 import time
3 from umqtt.simple import MQTTClient
4
5 SERVER = "mqtt.favoriot.com"
6 client = MQTTClient("umqtt_client", SERVER, user="1Kv8wVwKbH7G8z2Hxn1TEQEQuKr5OMj", password="1Kv8wVwKbH7G8z2Hxn1TEQEQuKr5OMj")
7
8 wlan = network.WLAN(network.STA_IF)
9 wlan.active(True)
10 wlan.disconnect()
11
12 -def sub_callback(topic, msg):
13     print((topic,msg))
14
15 -if not wlan.isconnected():
16     print('connecting to network ...')
17     wlan.connect('Galaxy Note20 Ultra 5G4fc9','00000000')
18
19 - while not wlan.isconnected():
20     print('network config:', wlan.ifconfig())
21     time.sleep(2)
22
23
24 - while wlan.isconnected():
25     topic1 = "1Kv8wVwKbH7G8z2Hxn1TEQEQuKr5OMj"
26     time.sleep(2)
27
28 network config: ('0.0.0.0', '0.0.0.0', '0.0.0.0', '192.168.97.190')
29 network config: ('0.0.0.0', '0.0.0.0', '0.0.0.0', '192.168.97.190')
30 network config: ('0.0.0.0', '0.0.0.0', '0.0.0.0', '192.168.97.190')
31 network config: ('0.0.0.0', '0.0.0.0', '0.0.0.0', '192.168.97.190')
32 network config: ('0.0.0.0', '0.0.0.0', '0.0.0.0', '192.168.97.190')
33 network config: ('0.0.0.0', '0.0.0.0', '0.0.0.0', '192.168.97.190')
34 network config: ('0.0.0.0', '0.0.0.0', '0.0.0.0', '192.168.97.190')
35 network config: ('0.0.0.0', '0.0.0.0', '0.0.0.0', '192.168.97.190')
36 network config: ('192.168.97.82', '255.255.255.0', '192.168.97.190', '192.168.97.190')
37 ok
```

As we can see the connection is successful ("ok") and its waiting whenever data is published to the server it will be retrieved from here.