Hand-on IoT Specialization - IoT Communications course

Introduction

The present work is the continuation of the IoT Devices course where I assembled the Freenove Smart Car 4WD. Please refer to this short video: https://youtu.be/d5gMhNugcTo.

I will show how I have configured the different proposed aspects in the agenda of the course:

1. Set up the Bluetooth communication between the Raspberry PI and the desktop computer
2. Set up the wifi communication between the Raspberry PI and the desktop computer
3. Start up the web application that uses ElectronJS in the front-end and the Wifi python server in the backend

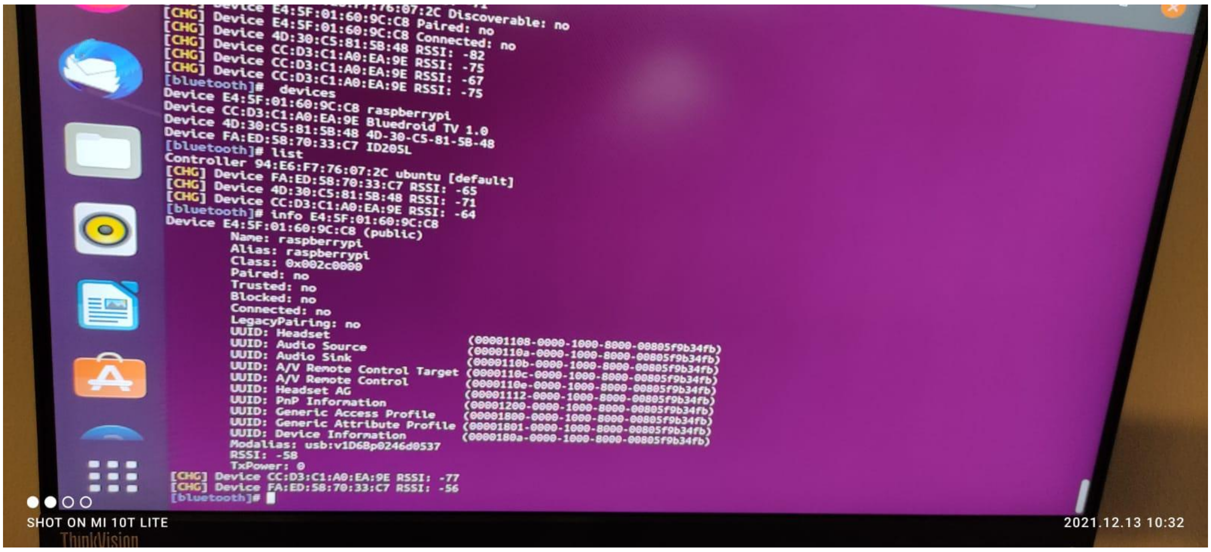
Setting up the Bluetooth connection

The devices used for these activities are a Windows 10 computer and the Raspberry PI. Initially, I installed the server program (bt_server.py) in the Raspberry and tried to do the same with the client one in Windows, but due to libraries problems that I didn't manage to solve I had to switch to other alternatives for the client operating system. I tried later running Ubuntu in VirtualBox and also VMWare (as was proposed) but it didn't work either, in both cases due to the virtualization tool didn't recognize the Bluetooth device of my desktop computer.
Due to all these limitations, I ran Ubuntu bare metal on the Desktop computer using a bootable flash USB stick, without installing the system, but configuring the package manager to install all the needed dependencies to be able to have Python, development libraries, and network utilities.
In such a way, I managed to run an operating system in the desktop computer able to run the python client using the pybluez library without problems and also recognizing the Bluetooth device (using bluetoothctl). And then I managed to run the client app connected to the server one in the Raspberry.

Something else to mention, it was necessary to pick some other port for the connection and configure properly the IDs of the server Bluetooth device in both sides. For this experiment, I chose port 10000 (and of course I had to allow the incoming connections from the firewall using ufw).

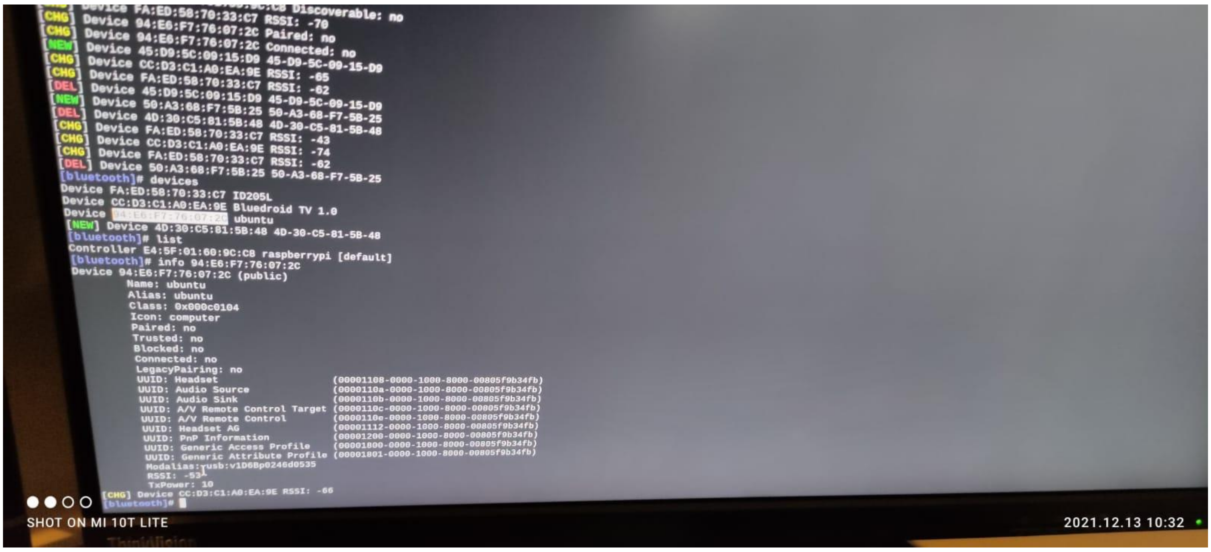
<u>Pictures:</u>

Running bluetoothctl on the desktop computer we can see that now the bluetooth device is recognized and we can also explore the near devices.



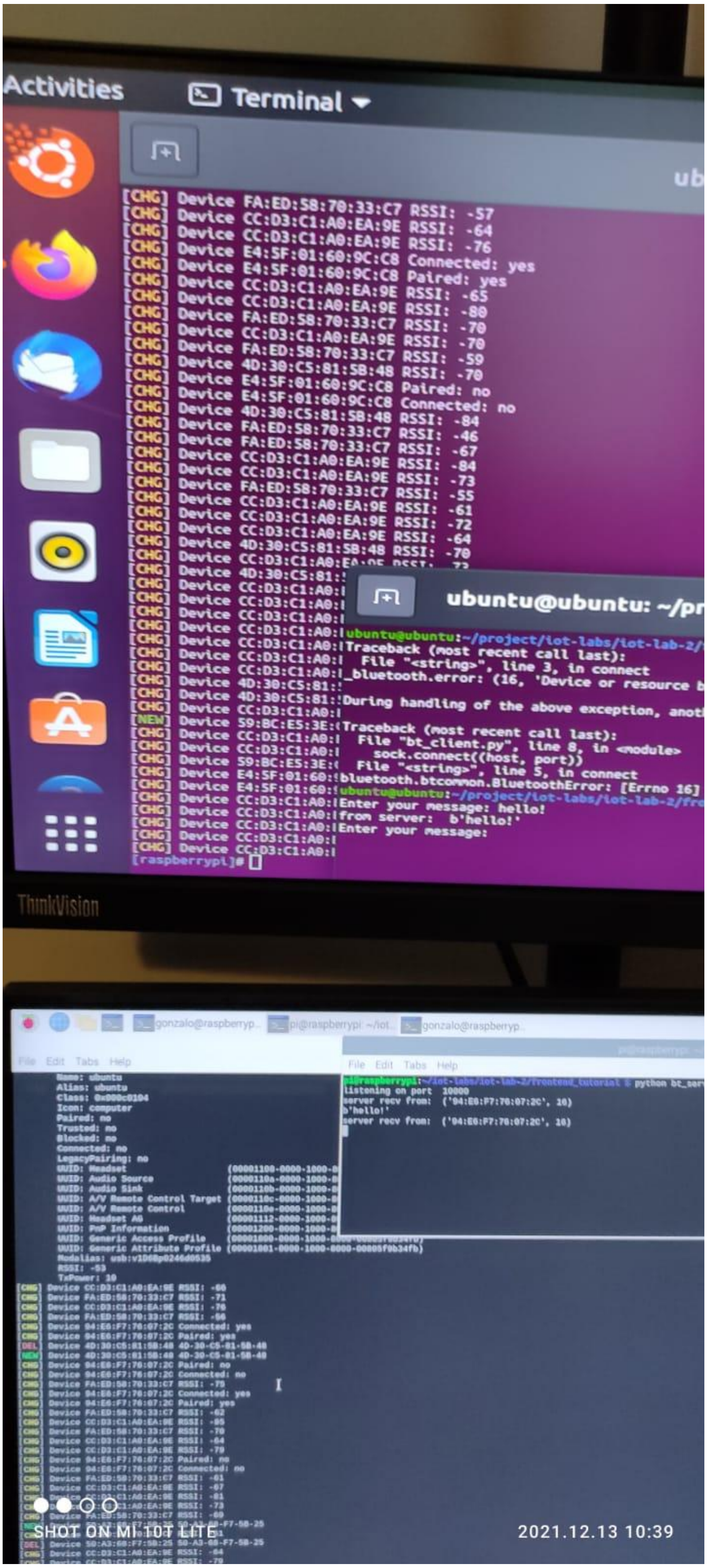
```
[CHG] Device E4:5F:01:60:9C:C8 Paired: no
[CHG] Device 4D:30:C5:81:5B:48 Connected: no
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -82
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -75
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -67
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -75
[bluetooth]#  devices
Device E4:5F:01:60:9C:C8 raspberrypi
Device CC:D3:C1:A0:EA:9E Bluedroid TV 1.0
Device 4D:30:C5:81:5B:48 4D-30-C5-81-5B-48
Device FA:ED:58:70:33:C7 ID205L
[bluetooth]# list
Controller 94:E6:F7:76:07:2C ubuntu [default]
[CHG] Device FA:ED:58:70:33:C7 RSSI: -65
[CHG] Device 4D:30:C5:81:5B:48 RSSI: -71
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -64
[bluetooth]# info E4:5F:01:60:9C:C8
Device E4:5F:01:60:9C:C8 (public)
        Name: raspberrypi
        Alias: raspberrypi
        Class: 0x002c0000
        Paired: no
        Trusted: no
        Blocked: no
        Connected: no
        LegacyPairing: no
        UUID: Headset                   (00001108-0000-1000-8000-00805f9b34fb)
        UUID: Audio Source              (0000110a-0000-1000-8000-00805f9b34fb)
        UUID: Audio Sink                (0000110b-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control Target (0000110c-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control        (0000110e-0000-1000-8000-00805f9b34fb)
        UUID: Headset AG                (00001112-0000-1000-8000-00805f9b34fb)
        UUID: PnP Information           (00001200-0000-1000-8000-00805f9b34fb)
        UUID: Generic Access Profile    (00001800-0000-1000-8000-00805f9b34fb)
        UUID: Generic Attribute Profile (00001801-0000-1000-8000-00805f9b34fb)
        UUID: Device Information        (0000180a-0000-1000-8000-00805f9b34fb)
        Modalias: usb:v1D6Bp0246d0537
        RSSI: -58
        TxPower: 0
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -77
[CHG] Device FA:ED:58:70:33:C7 RSSI: -56
[bluetooth]# 
```

On the other hand, we can see the Ubuntu bluetooth device on the Raspberry PI



```
[CHG] Device 94:E6:F7:76:07:2C RSSI: -70
[CHG] Device 94:E6:F7:76:07:2C Paired: no
[NEW] Device 45:D9:5C:09:15:D9 Connected: no
[CHG] Device CC:D3:C1:A0:EA:9E 45-D9-5C-09-15-D9
[CHG] Device FA:ED:58:70:33:C7 RSSI: -65
[DEL] Device 45:D9:5C:09:15:D9 RSSI: -62
[NEW] Device 50:A3:68:F7:5B:25 45-D9-5C-09-15-D9
[DEL] Device 4D:30:C5:81:5B:48 50-A3-68-F7-5B-25
[CHG] Device FA:ED:58:70:33:C7 4D-30-C5-81-5B-48
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -43
[CHG] Device FA:ED:58:70:33:C7 RSSI: -74
[DEL] Device 50:A3:68:F7:5B:25 RSSI: -62
                                50-A3-68-F7-5B-25
[bluetooth]# devices
Device FA:ED:58:70:33:C7 ID205L
Device CC:D3:C1:A0:EA:9E Bluedroid TV 1.0
Device 94:E6:F7:76:07:2C ubuntu
[NEW] Device 4D:30:C5:81:5B:48 4D-30-C5-81-5B-48
[bluetooth]# list
Controller E4:5F:01:60:9C:C8 raspberrypi [default]
[bluetooth]# info 94:E6:F7:76:07:2C
Device 94:E6:F7:76:07:2C (public)
        Name: ubuntu
        Alias: ubuntu
        Class: 0x000c0104
        Icon: computer
        Paired: no
        Trusted: no
        Blocked: no
        Connected: no
        LegacyPairing: no
        UUID: Headset                   (00001108-0000-1000-8000-00805f9b34fb)
        UUID: Audio Source              (0000110a-0000-1000-8000-00805f9b34fb)
        UUID: Audio Sink                (0000110b-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control Target (0000110c-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control        (0000110e-0000-1000-8000-00805f9b34fb)
        UUID: Headset AG                (00001112-0000-1000-8000-00805f9b34fb)
        UUID: PnP Information           (00001200-0000-1000-8000-00805f9b34fb)
        UUID: Generic Access Profile    (00001800-0000-1000-8000-00805f9b34fb)
        UUID: Generic Attribute Profile (00001801-0000-1000-8000-00805f9b34fb)
        Modalias: usb:v1D6Bp0246d0535
        RSSI: -53
        TxPower: 10
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -66
[bluetooth]# 
```

Then, as required, we can run the server on the Raspberry and the client on the Ubuntu desktop…



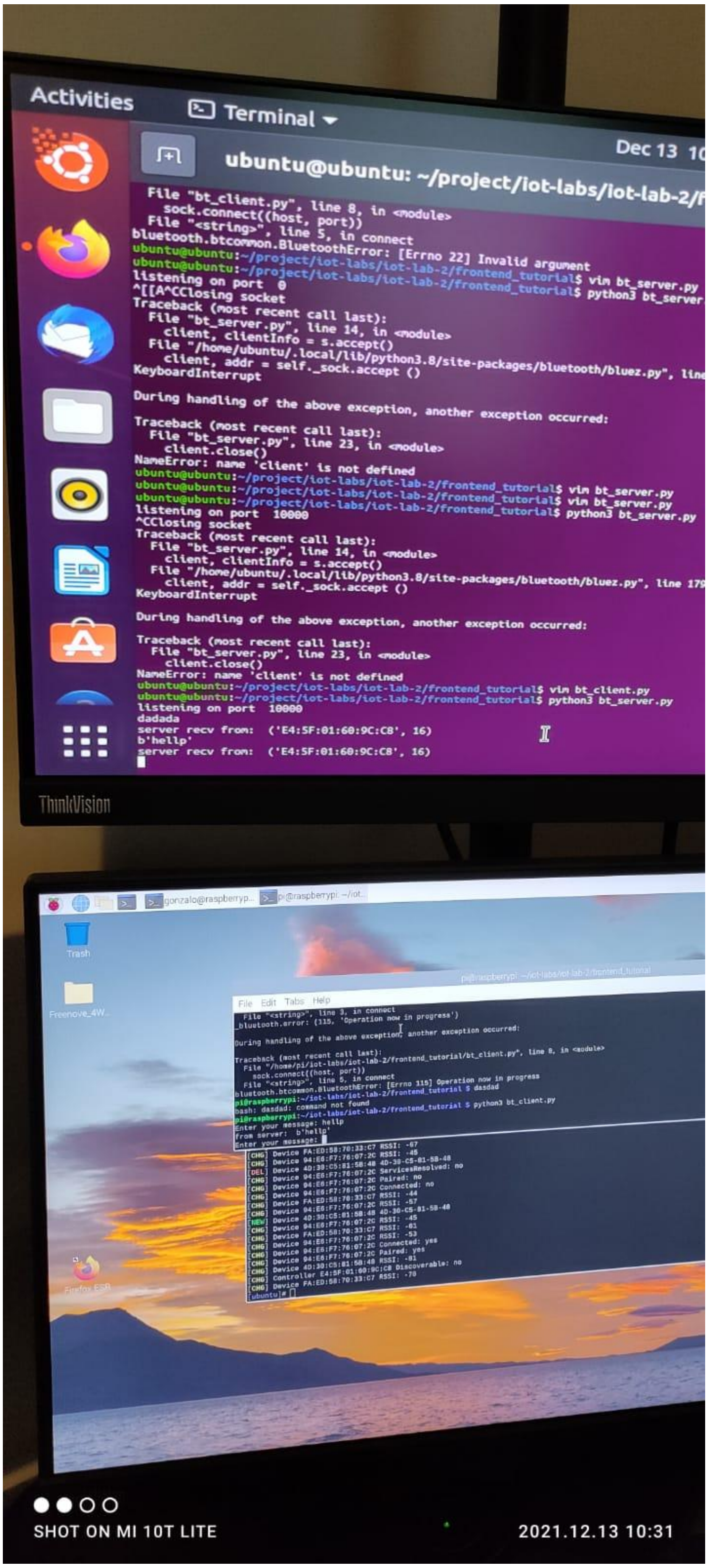
```
[CHG] Device FA:ED:58:70:33:C7 RSSI: -57
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -64
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -76
[CHG] Device E4:5F:01:60:9C:C8 Connected: yes
[CHG] Device E4:5F:01:60:9C:C8 Paired: yes
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -65
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -80
[CHG] Device FA:ED:58:70:33:C7 RSSI: -70
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -70
[CHG] Device FA:ED:58:70:33:C7 RSSI: -59
[CHG] Device 4D:30:C5:81:5B:48 RSSI: -70
[CHG] Device E4:5F:01:60:9C:C8 Paired: no
[CHG] Device E4:5F:01:60:9C:C8 Connected: no
[CHG] Device 4D:30:C5:81:5B:48 RSSI: -84
[CHG] Device FA:ED:58:70:33:C7 RSSI: -46
[CHG] Device FA:ED:58:70:33:C7 RSSI: -67
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -84
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -73
[CHG] Device FA:ED:58:70:33:C7 RSSI: -55
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -61
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -72
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -64
[CHG] Device 4D:30:C5:81:5B:48 RSSI: -70
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -73
[CHG] Device 4D:30:C5:81:
[CHG] Device CC:D3:C1:A0:
[CHG] Device CC:D3:C1:A0:
[CHG] Device CC:D3:C1:A0:
[CHG] Device CC:D3:C1:A0:
[CHG] Device CC:D3:C1:A0:
[CHG] Device CC:D3:C1:A0:
[CHG] Device CC:D3:C1:A0:
[CHG] Device 4D:30:C5:81:
[CHG] Device 4D:30:C5:81:
[CHG] Device CC:D3:C1:A0:
[NEW] Device 59:BC:E5:3E:
[CHG] Device CC:D3:C1:A0:
[CHG] Device CC:D3:C1:A0:
[CHG] Device 59:BC:E5:3E:
[CHG] Device E4:5F:01:60:
[CHG] Device E4:5F:01:60:
[CHG] Device CC:D3:C1:A0:
[CHG] Device CC:D3:C1:A0:
[CHG] Device CC:D3:C1:A0:
[CHG] Device CC:D3:C1:A0:
[CHG] Device CC:D3:C1:A0:
[raspberrypi]#
```

```
ubuntu@ubuntu:~/project/iot-labs/iot-lab-2/
Traceback (most recent call last):
  File "<string>", line 3, in connect
_bluetooth.error: (16, 'Device or resource b

During handling of the above exception, anot

Traceback (most recent call last):
  File "bt_client.py", line 8, in <module>
    sock.connect((host, port))
  File "<string>", line 5, in connect
bluetooth.btcommon.BluetoothError: [Errno 16]
ubuntu@ubuntu:~/project/iot-labs/iot-lab-2/fro
Enter your message: hello!
from server:  b'hello!'
Enter your message:
```

```
        Name: ubuntu
        Alias: ubuntu
        Class: 0x000c0104
        Icon: computer
        Paired: no
        Trusted: no
        Blocked: no
        Connected: no
        LegacyPairing: no
        UUID: Headset                   (00001108-0000-1000-8
        UUID: Audio Source              (0000110a-0000-1000-8
        UUID: Audio Sink                (0000110b-0000-1000-8
        UUID: A/V Remote Control Target (0000110c-0000-1000-8
        UUID: A/V Remote Control        (0000110e-0000-1000-8
        UUID: Headset AG                (00001112-0000-1000-8
        UUID: PnP Information           (00001200-0000-1000-8
        UUID: Generic Access Profile    (00001800-0000-1000-
        UUID: Generic Attribute Profile (00001801-0000-1000-8000-00805f9b34fb)
        Modalias: usb:v1D6Bp0246d0535
        RSSI: -53
        TxPower: 10
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -66
[CHG] Device FA:ED:58:70:33:C7 RSSI: -71
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -76
[CHG] Device FA:ED:58:70:33:C7 RSSI: -56
[CHG] Device 94:E6:F7:76:07:2C Connected: yes
[CHG] Device 94:E6:F7:76:07:2C Paired: yes
[DEL] Device 4D:30:C5:81:5B:48 4D-30-C5-81-5B-48
[NEW] Device 4D:30:C5:81:5B:48 4D-30-C5-81-5B-48
[CHG] Device 94:E6:F7:76:07:2C Paired: no
[CHG] Device 94:E6:F7:76:07:2C Connected: no
[CHG] Device FA:ED:58:70:33:C7 RSSI: -79
[CHG] Device 94:E6:F7:76:07:2C Connected: yes
[CHG] Device 94:E6:F7:76:07:2C Paired: yes
[CHG] Device FA:ED:58:70:33:C7 RSSI: -62
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -85
[CHG] Device FA:ED:58:70:33:C7 RSSI: -70
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -64
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -79
[CHG] Device 94:E6:F7:76:07:2C Paired: no
[CHG] Device 94:E6:F7:76:07:2C Connected: no
[CHG] Device FA:ED:58:70:33:C7 RSSI: -61
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -67
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -81
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -73
[CHG] Device FA:ED:58:70:33:C7 RSSI: -69
[NEW] Device 50:A3:68:F7:5B:25 50-A3-68-F7-5B-25
[CHG] [illegible]
[DEL] Device 50:A3:68:F7:5B:25 50-A3-68-F7-5B-25
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -84
[CHG] Device CC:D3:C1:A0:EA:9E RSSI: -79
```

```
pi@raspberrypi:~/iot-labs/iot-lab-2/frontend_tutorial $ python bt_serv
listening on port  10000
server recv from:  ('94:E6:F7:76:07:2C', 10)
b'hello!'
server recv from:  ('94:E6:F7:76:07:2C', 10)
```

Or in the other way round, the client on the Raspberry and the server on the Ubuntu desktop..



```
ubuntu@ubuntu: ~/project/iot-labs/iot-lab-2/f
  File "bt_client.py", line 8, in <module>
    sock.connect((host, port))
  File "<string>", line 5, in connect
bluetooth.btcommon.BluetoothError: [Errno 22] Invalid argument
ubuntu@ubuntu:~/project/iot-labs/iot-lab-2/frontend_tutorial$ vim bt_server.py
ubuntu@ubuntu:~/project/iot-labs/iot-lab-2/frontend_tutorial$ python3 bt_server
listening on port  0
^[[A^CClosing socket
Traceback (most recent call last):
  File "bt_server.py", line 14, in <module>
    client, clientInfo = s.accept()
  File "/home/ubuntu/.local/lib/python3.8/site-packages/bluetooth/bluez.py", line
    client, addr = self._sock.accept ()
KeyboardInterrupt

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "bt_server.py", line 23, in <module>
    client.close()
NameError: name 'client' is not defined
ubuntu@ubuntu:~/project/iot-labs/iot-lab-2/frontend_tutorial$ vim bt_server.py
ubuntu@ubuntu:~/project/iot-labs/iot-lab-2/frontend_tutorial$ vim bt_server.py
ubuntu@ubuntu:~/project/iot-labs/iot-lab-2/frontend_tutorial$ python3 bt_server.py
listening on port  10000
^CClosing socket
Traceback (most recent call last):
  File "bt_server.py", line 14, in <module>
    client, clientInfo = s.accept()
  File "/home/ubuntu/.local/lib/python3.8/site-packages/bluetooth/bluez.py", line 179
    client, addr = self._sock.accept ()
KeyboardInterrupt

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "bt_server.py", line 23, in <module>
    client.close()
NameError: name 'client' is not defined
ubuntu@ubuntu:~/project/iot-labs/iot-lab-2/frontend_tutorial$ vim bt_client.py
ubuntu@ubuntu:~/project/iot-labs/iot-lab-2/frontend_tutorial$ python3 bt_server.py
listening on port  10000
dadada
server recv from:  ('E4:5F:01:60:9C:C8', 16)
b'hellp'
server recv from:  ('E4:5F:01:60:9C:C8', 16)
```

```
pi@raspberrypi: ~/iot-labs/iot-lab-2/frontend_tutorial
File  Edit  Tabs  Help
  File "<string>", line 3, in connect
_bluetooth.error: (115, 'Operation now in progress')

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/home/pi/iot-labs/iot-lab-2/frontend_tutorial/bt_client.py", line 8, in <module>
    sock.connect((host, port))
  File "<string>", line 5, in connect
bluetooth.btcommon.BluetoothError: [Errno 115] Operation now in progress
pi@raspberrypi:~/iot-labs/iot-lab-2/frontend_tutorial $ dasdad
bash: dasdad: command not found
pi@raspberrypi:~/iot-labs/iot-lab-2/frontend_tutorial $ python3 bt_client.py
Enter your message: hellp
from server:  b'hellp'
Enter your message: 
```

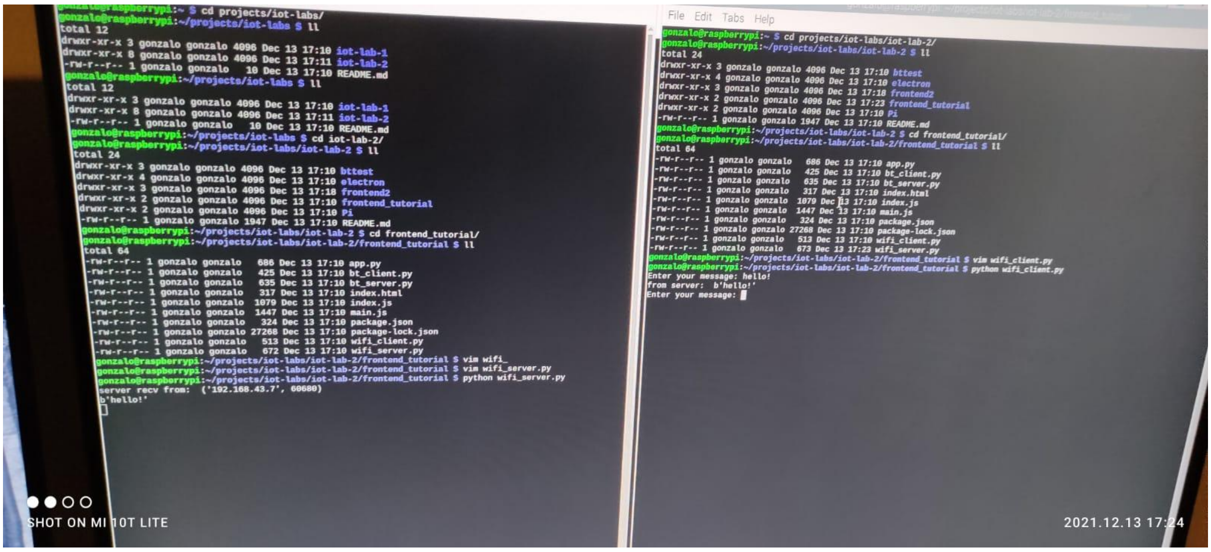
```
[CHG] Device FA:ED:58:70:33:C7 RSSI: -67
[CHG] Device FA:ED:58:70:33:C7 RSSI: -45
[CHG] Device 94:E6:F7:76:07:2C 4D-30-C5-81-5B-48
[DEL] Device 4D:30:C5:81:5B:48 ServicesResolved: no
[CHG] Device 94:E6:F7:76:07:2C Paired: no
[CHG] Device 94:E6:F7:76:07:2C Connected: no
[CHG] Device 94:E6:F7:76:07:2C RSSI: -44
[CHG] Device FA:ED:58:70:33:C7 RSSI: -57
[CHG] Device 94:E6:F7:76:07:2C 4D-30-C5-81-5B-48
[CHG] Device 4D:30:C5:81:5B:48 RSSI: -45
[NEW] Device 94:E6:F7:76:07:2C RSSI: -81
[CHG] Device FA:ED:58:70:33:C7 RSSI: -53
[CHG] Device 94:E6:F7:76:07:2C Connected: yes
[CHG] Device 94:E6:F7:76:07:2C Paired: yes
[CHG] Device 4D:30:C5:81:5B:48 RSSI: -81
[CHG] Device 4D:30:C5:81:60:9C:C8 Discoverable: no
[CHG] Controller E4:5F:01:60:9C:C8 RSSI: -78
[CHG] Device FA:ED:58:70:33:C7
[ubuntu]# 
```

Wireless connection

For setting up the socket communication over the TCP-IP WiFi connection I just had to install the socket library on Windows and Raspbian, configure the right IP addresses in the python code, and allow the incoming connections to the port. Other than that the connection was quite straightforward.

<u>Pictures</u>

First I ran both programs on the Raspberry PI to do a quick test…



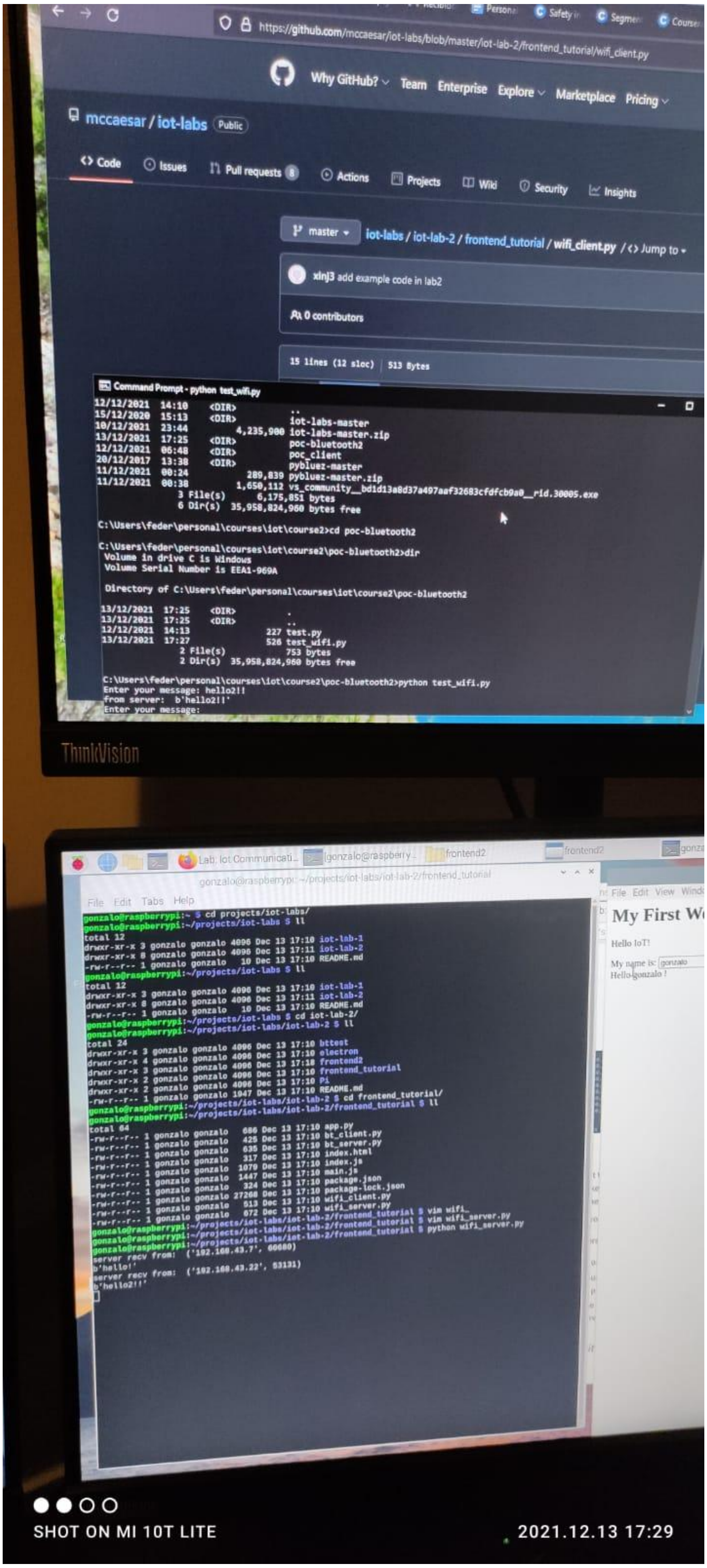
```
gonzalo@raspberrypi:~ $ cd projects/iot-labs/
gonzalo@raspberrypi:~/projects/iot-labs $ ll
total 12
drwxr-xr-x 3 gonzalo gonzalo 4096 Dec 13 17:10 iot-lab-1
drwxr-xr-x 8 gonzalo gonzalo 4096 Dec 13 17:11 iot-lab-2
-rw-r--r-- 1 gonzalo gonzalo   10 Dec 13 17:10 README.md
gonzalo@raspberrypi:~/projects/iot-labs $ ll
total 12
drwxr-xr-x 3 gonzalo gonzalo 4096 Dec 13 17:10 iot-lab-1
drwxr-xr-x 8 gonzalo gonzalo 4096 Dec 13 17:11 iot-lab-2
-rw-r--r-- 1 gonzalo gonzalo   10 Dec 13 17:10 README.md
gonzalo@raspberrypi:~/projects/iot-labs $ cd iot-lab-2/
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2 $ ll
total 24
drwxr-xr-x 3 gonzalo gonzalo 4096 Dec 13 17:10 bttest
drwxr-xr-x 4 gonzalo gonzalo 4096 Dec 13 17:10 electron
drwxr-xr-x 3 gonzalo gonzalo 4096 Dec 13 17:18 frontend2
drwxr-xr-x 2 gonzalo gonzalo 4096 Dec 13 17:10 frontend_tutorial
drwxr-xr-x 2 gonzalo gonzalo 4096 Dec 13 17:10 Pi
-rw-r--r-- 1 gonzalo gonzalo 1947 Dec 13 17:10 README.md
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2 $ cd frontend_tutorial/
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ ll
total 64
-rw-r--r-- 1 gonzalo gonzalo   686 Dec 13 17:10 app.py
-rw-r--r-- 1 gonzalo gonzalo   425 Dec 13 17:10 bt_client.py
-rw-r--r-- 1 gonzalo gonzalo   635 Dec 13 17:10 bt_server.py
-rw-r--r-- 1 gonzalo gonzalo   317 Dec 13 17:10 index.html
-rw-r--r-- 1 gonzalo gonzalo  1079 Dec 13 17:10 index.js
-rw-r--r-- 1 gonzalo gonzalo  1447 Dec 13 17:10 main.js
-rw-r--r-- 1 gonzalo gonzalo   324 Dec 13 17:10 package.json
-rw-r--r-- 1 gonzalo gonzalo 27268 Dec 13 17:10 package-lock.json
-rw-r--r-- 1 gonzalo gonzalo   513 Dec 13 17:10 wifi_client.py
-rw-r--r-- 1 gonzalo gonzalo   672 Dec 13 17:10 wifi_server.py
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ vim wifi_
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ vim wifi_server.py
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ python wifi_server.py
server recv from:  ('192.168.43.7', 60680)
b'hello!'
```

```
gonzalo@raspberrypi:~ $ cd projects/iot-labs/iot-lab-2/
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2 $ ll
total 24
drwxr-xr-x 3 gonzalo gonzalo 4096 Dec 13 17:10 bttest
drwxr-xr-x 4 gonzalo gonzalo 4096 Dec 13 17:10 electron
drwxr-xr-x 3 gonzalo gonzalo 4096 Dec 13 17:18 frontend2
drwxr-xr-x 2 gonzalo gonzalo 4096 Dec 13 17:23 frontend_tutorial
drwxr-xr-x 2 gonzalo gonzalo 4096 Dec 13 17:10 Pi
-rw-r--r-- 1 gonzalo gonzalo 1947 Dec 13 17:10 README.md
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2 $ cd frontend_tutorial/
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ ll
total 64
-rw-r--r-- 1 gonzalo gonzalo   686 Dec 13 17:10 app.py
-rw-r--r-- 1 gonzalo gonzalo   425 Dec 13 17:10 bt_client.py
-rw-r--r-- 1 gonzalo gonzalo   635 Dec 13 17:10 bt_server.py
-rw-r--r-- 1 gonzalo gonzalo   317 Dec 13 17:10 index.html
-rw-r--r-- 1 gonzalo gonzalo  1079 Dec 13 17:10 index.js
-rw-r--r-- 1 gonzalo gonzalo  1447 Dec 13 17:10 main.js
-rw-r--r-- 1 gonzalo gonzalo   324 Dec 13 17:10 package.json
-rw-r--r-- 1 gonzalo gonzalo 27268 Dec 13 17:10 package-lock.json
-rw-r--r-- 1 gonzalo gonzalo   513 Dec 13 17:10 wifi_client.py
-rw-r--r-- 1 gonzalo gonzalo   673 Dec 13 17:23 wifi_server.py
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ vim wifi_client.py
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ python wifi_client.py
Enter your message: hello!
from server:  b'hello!'
Enter your message:
```

SHOT ON MI 10T LITE

2021.12.13 17:24

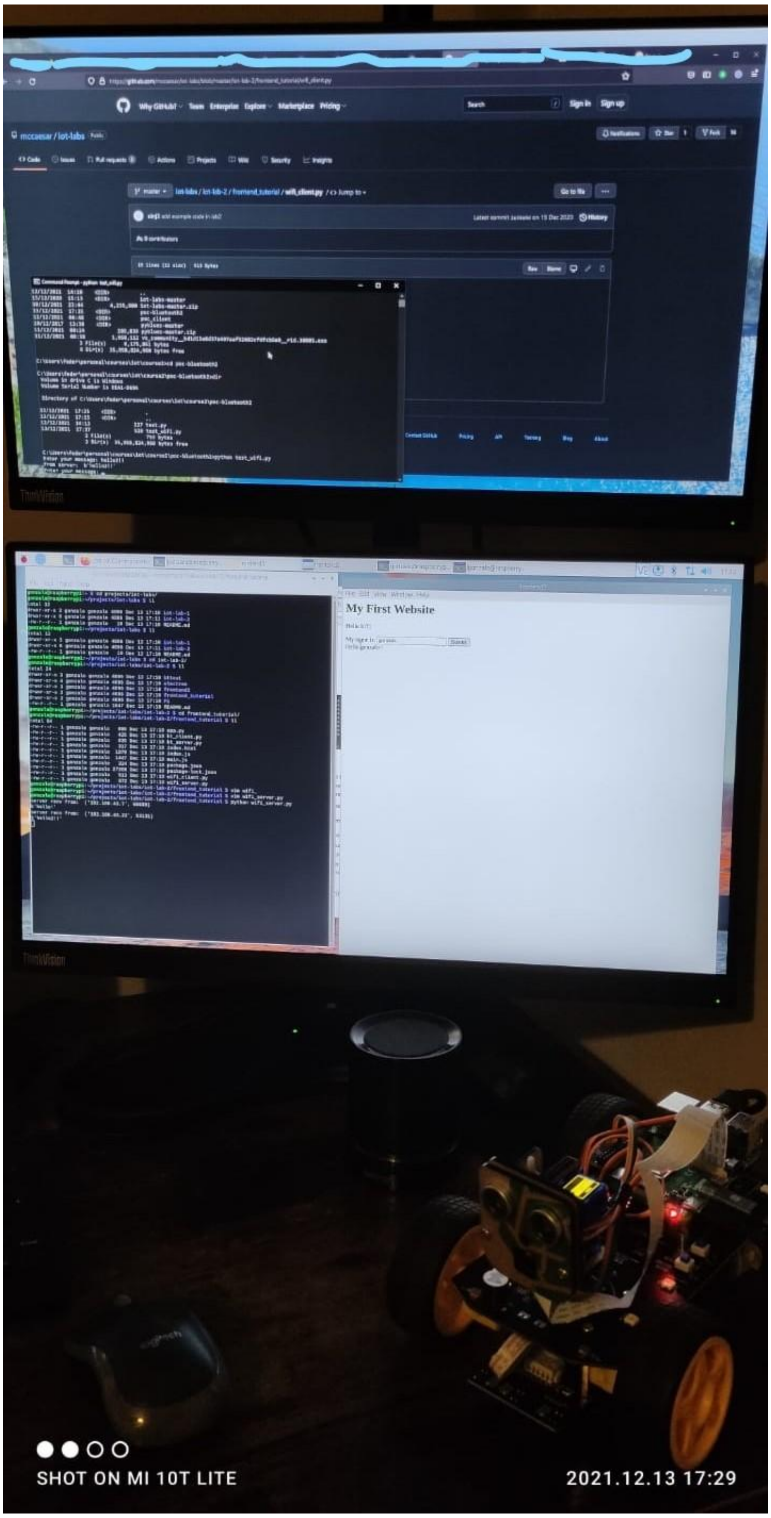
And later I ran the client one on Windows..


https://github.com/mccaesar/iot-labs/blob/master/iot-lab-2/frontend_tutorial/wifi_client.py
Why GitHub?
Team
Enterprise
Explore
Marketplace
Pricing
mccaesar / iot-labs
Public
Code
Issues
Pull requests 8
Actions
Projects
Wiki
Security
Insights
master
iot-labs / iot-lab-2 / frontend_tutorial / wifi_client.py
Jump to
xinj3 add example code in lab2
0 contributors
15 lines (12 sloc) | 513 Bytes
Command Prompt - python test_wifi.py
ThinkVision

```
12/12/2021  14:10    <DIR>          ..
15/12/2020  15:13    <DIR>          iot-labs-master
10/12/2021  23:44         4,235,900 iot-labs-master.zip
13/12/2021  17:25    <DIR>          poc-bluetooth2
12/12/2021  06:48    <DIR>          poc_client
20/12/2017  13:38    <DIR>          pybluez-master
11/12/2021  00:24           289,839 pybluez-master.zip
11/12/2021  00:38         1,650,112 vs_community__bd1d13a8d37a497aaf32683cfdfcb9a0__rid.30005.exe
               3 File(s)      6,175,851 bytes
               6 Dir(s)  35,958,824,960 bytes free

C:\Users\feder\personal\courses\iot\course2>cd poc-bluetooth2

C:\Users\feder\personal\courses\iot\course2\poc-bluetooth2>dir
 Volume in drive C is Windows
 Volume Serial Number is EEA1-969A

 Directory of C:\Users\feder\personal\courses\iot\course2\poc-bluetooth2

13/12/2021  17:25    <DIR>          .
13/12/2021  17:25    <DIR>          ..
12/12/2021  14:13               227 test.py
13/12/2021  17:27               526 test_wifi.py
               2 File(s)            753 bytes
               2 Dir(s)  35,958,824,960 bytes free

C:\Users\feder\personal\courses\iot\course2\poc-bluetooth2>python test_wifi.py
Enter your message: hello2!!
from server:  b'hello2!!'
Enter your message:
```

```
gonzalo@raspberrypi:~ $ cd projects/iot-labs/
gonzalo@raspberrypi:~/projects/iot-labs $ ll
total 12
drwxr-xr-x 3 gonzalo gonzalo 4096 Dec 13 17:10 iot-lab-1
drwxr-xr-x 8 gonzalo gonzalo 4096 Dec 13 17:11 iot-lab-2
-rw-r--r-- 1 gonzalo gonzalo   10 Dec 13 17:10 README.md
gonzalo@raspberrypi:~/projects/iot-labs $ ll
total 12
drwxr-xr-x 3 gonzalo gonzalo 4096 Dec 13 17:10 iot-lab-1
drwxr-xr-x 8 gonzalo gonzalo 4096 Dec 13 17:11 iot-lab-2
-rw-r--r-- 1 gonzalo gonzalo   10 Dec 13 17:10 README.md
gonzalo@raspberrypi:~/projects/iot-labs $ cd iot-lab-2/
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2 $ ll
total 24
drwxr-xr-x 3 gonzalo gonzalo 4096 Dec 13 17:10 bttest
drwxr-xr-x 4 gonzalo gonzalo 4096 Dec 13 17:10 electron
drwxr-xr-x 3 gonzalo gonzalo 4096 Dec 13 17:18 frontend2
drwxr-xr-x 2 gonzalo gonzalo 4096 Dec 13 17:10 frontend_tutorial
drwxr-xr-x 2 gonzalo gonzalo 4096 Dec 13 17:10 Pi
-rw-r--r-- 1 gonzalo gonzalo 1947 Dec 13 17:10 README.md
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2 $ cd frontend_tutorial/
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ ll
total 64
-rw-r--r-- 1 gonzalo gonzalo   686 Dec 13 17:10 app.py
-rw-r--r-- 1 gonzalo gonzalo   425 Dec 13 17:10 bt_client.py
-rw-r--r-- 1 gonzalo gonzalo   635 Dec 13 17:10 bt_server.py
-rw-r--r-- 1 gonzalo gonzalo   317 Dec 13 17:10 index.html
-rw-r--r-- 1 gonzalo gonzalo  1079 Dec 13 17:10 index.js
-rw-r--r-- 1 gonzalo gonzalo  1447 Dec 13 17:10 main.js
-rw-r--r-- 1 gonzalo gonzalo   324 Dec 13 17:10 package.json
-rw-r--r-- 1 gonzalo gonzalo 27268 Dec 13 17:10 package-lock.json
-rw-r--r-- 1 gonzalo gonzalo   513 Dec 13 17:10 wifi_client.py
-rw-r--r-- 1 gonzalo gonzalo   672 Dec 13 17:10 wifi_server.py
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ vim wifi_
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ vim wifi_server.py
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ python wifi_server.py
server recv from:  ('192.168.43.7', 60680)
b'hello!'
server recv from:  ('192.168.43.22', 53131)
b'hello2!!'
```

And another picture with my lovely Smart Car ^^ (isn't it cute?)

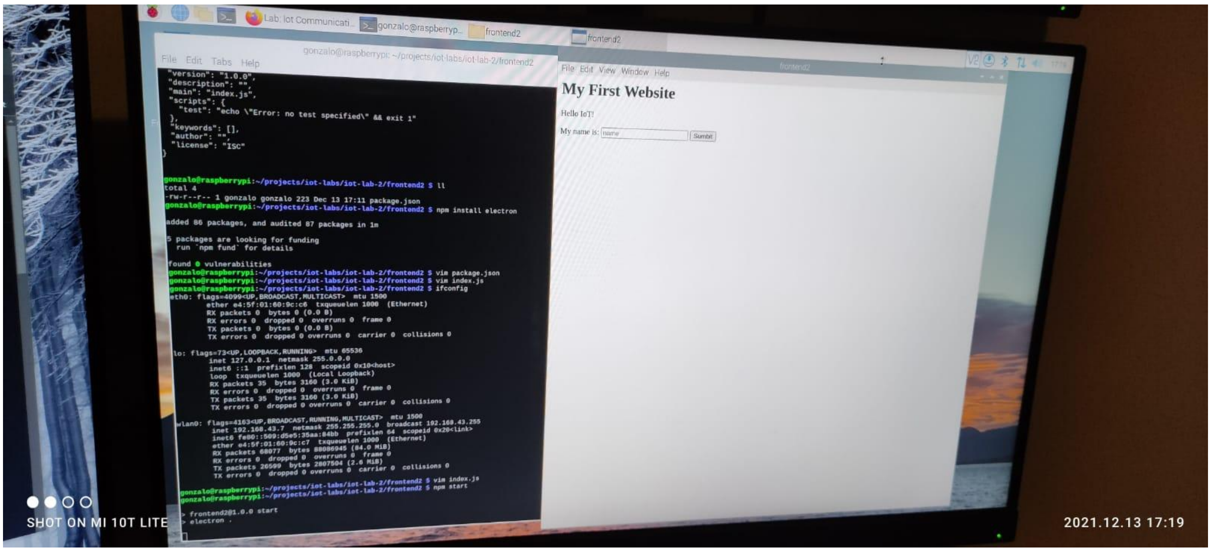


Web Frontend using Node.js and ElectronJs

To start up the web frontend I first cloned the indicated repository and ran the commands for installing the node dependencies and starting up the server (npm install & npm start). This didn’t work quite well then I followed the steps for creating the app from scratch but adding the files of the mentioned repository.
Then I ran:

```
$ npm init -y
$ npm install electron
$ npm start
```

After adding the files of the iot-labs-2 folder in my npm project I managed to start up the web app (the hello world one).



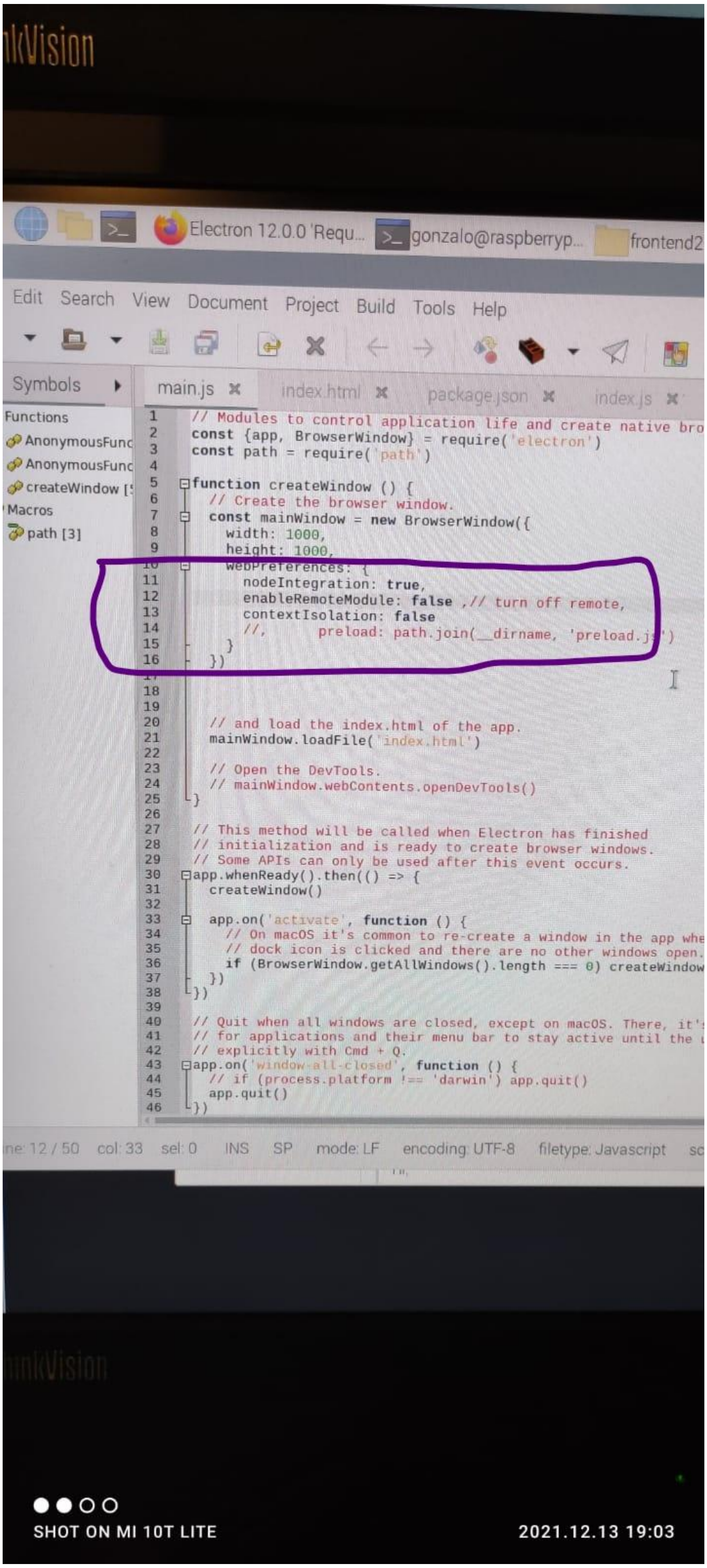
```
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}


gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend2 $ ll
total 4
-rw-r--r-- 1 gonzalo gonzalo 223 Dec 13 17:11 package.json
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend2 $ npm install electron

added 86 packages, and audited 87 packages in 1m

5 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend2 $ vim package.json
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend2 $ vim index.js
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend2 $ ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether e4:5f:01:60:9c:c6  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 35  bytes 3160 (3.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 35  bytes 3160 (3.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.43.7  netmask 255.255.255.0  broadcast 192.168.43.255
        inet6 fe80::509:d5e5:35aa:84bb  prefixlen 64  scopeid 0x20<link>
        ether e4:5f:01:60:9c:c7  txqueuelen 1000  (Ethernet)
        RX packets 68077  bytes 88086945 (84.0 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 26599  bytes 2807504 (2.6 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend2 $ vim index.js
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend2 $ npm start

> frontend2@1.0.0 start
> electron .
```

My First Website

Hello IoT!

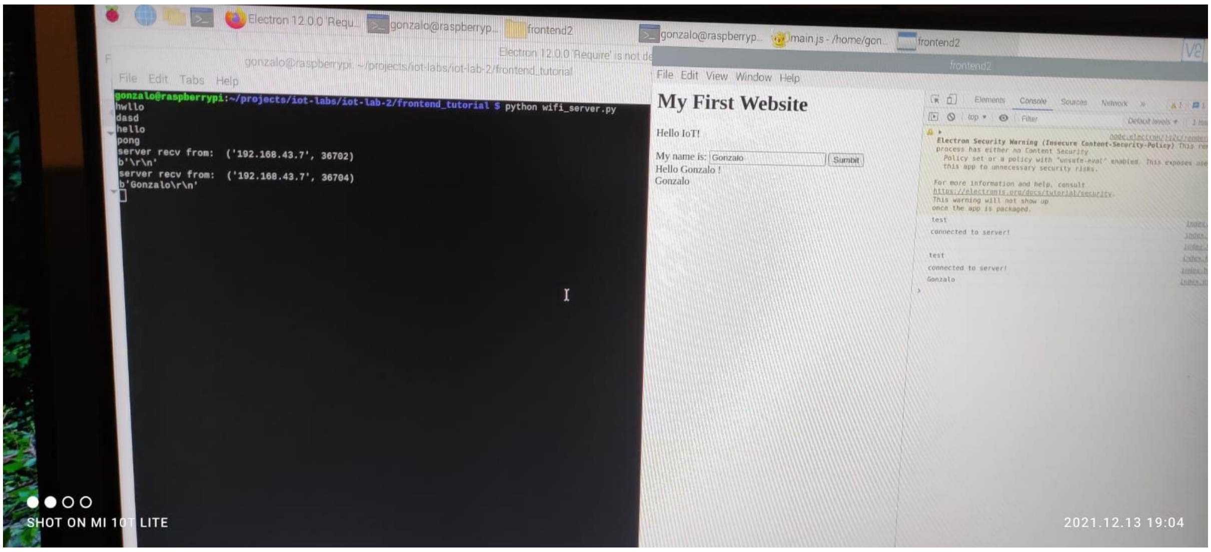
My name is: name Sumbit

Then I configured the right IP addresses in the index.js and wifi_server.py files to establish the backend socket communication.
I had some problems at this point due to some runtime behavior of ElectronJS. I haven’t used this framework before then I had to research a bit. The problem was that even in node is quite common to use the verb ‘require’ to include libraries and other modules, it just runs in the node server (the backend). Here in this application, it seems that ElectronJS is a front-end framework that is embedding the web application in sort of desktop application without using the browser for the renderization (just the engine of the browser) and it also uses the ‘require’ verb everywhere. Anyway, even this is normal for ElectronJS the renderization engine was throwing an error that ‘require is not defined’.
After googling a bit and trying different alternatives I solved just including one line of code in the configuration of Electron setting ‘contextIsolation’ to ‘false’.



```
// Modules to control application life and create native bro
const {app, BrowserWindow} = require('electron')
const path = require('path')

function createWindow () {
  // Create the browser window.
  const mainWindow = new BrowserWindow({
    width: 1000,
    height: 1000,
    webPreferences: {
      nodeIntegration: true,
      enableRemoteModule: false ,// turn off remote,
      contextIsolation: false
      //,       preload: path.join(__dirname, 'preload.js')
    }
  })



  // and load the index.html of the app.
  mainWindow.loadFile('index.html')

  // Open the DevTools.
  // mainWindow.webContents.openDevTools()
}

// This method will be called when Electron has finished
// initialization and is ready to create browser windows.
// Some APIs can only be used after this event occurs.
app.whenReady().then(() => {
  createWindow()

  app.on('activate', function () {
    // On macOS it's common to re-create a window in the app whe
    // dock icon is clicked and there are no other windows open.
    if (BrowserWindow.getAllWindows().length === 0) createWindow
  })
})

// Quit when all windows are closed, except on macOS. There, it'
// for applications and their menu bar to stay active until the
// explicitly with Cmd + Q.
app.on('window-all-closed', function () {
  // if (process.platform !== 'darwin') app.quit()
  app.quit()
})
```

After doing that, the web app started up without problems and I managed to conclude the communication between the hello world page and the python server app.



```
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend_tutorial $ python wifi_server.py
hwllo
dasd
hello
pong
server recv from:  ('192.168.43.7', 36702)
b'\r\n'
server recv from:  ('192.168.43.7', 36704)
b'Gonzalo\r\n'
```

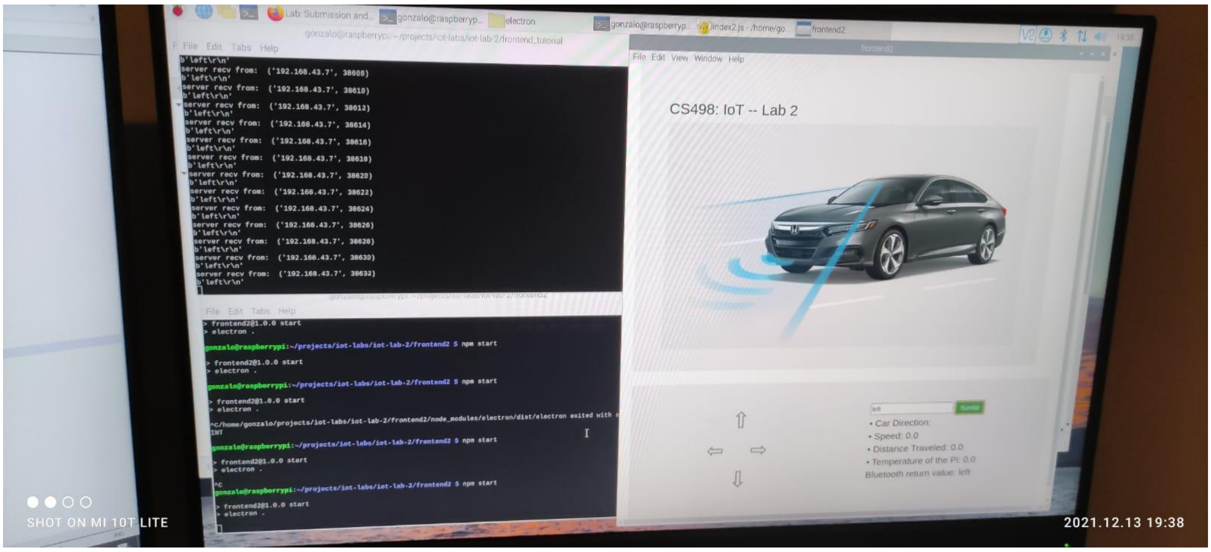
My First Website

Hello IoT!

My name is: Gonzalo Sumbit
Hello Gonzalo !
Gonzalo

```
Electron Security Warning (Insecure Content-Security-Policy) This renderer
process has either no Content Security
  Policy set or a policy with "unsafe-eval" enabled. This exposes users of
  this app to unnecessary security risks.

For more information and help, consult
https://electronjs.org/docs/tutorial/security.
This warning will not show up
once the app is packaged.
test
connected to server!

test
connected to server!
Gonzalo
```

Finally for starting up the proper Car Web frontend and send instructions to the car I just moved the files from the electron folder to the current one, configured the IP addresses again and again ran npm start and then in the new frontend just indicating the directions to execute ‘left’, ‘right’, etc, that was sent to the python backend that is able to communicate with the Raspberry PI drivers and execute it on the car.



```
b'left\r\n'
server recv from:  ('192.168.43.7', 38608)
b'left\r\n'
server recv from:  ('192.168.43.7', 38610)
b'left\r\n'
server recv from:  ('192.168.43.7', 38612)
b'left\r\n'
server recv from:  ('192.168.43.7', 38614)
b'left\r\n'
server recv from:  ('192.168.43.7', 38616)
b'left\r\n'
server recv from:  ('192.168.43.7', 38618)
b'left\r\n'
server recv from:  ('192.168.43.7', 38620)
b'left\r\n'
server recv from:  ('192.168.43.7', 38622)
b'left\r\n'
server recv from:  ('192.168.43.7', 38624)
b'left\r\n'
server recv from:  ('192.168.43.7', 38626)
b'left\r\n'
server recv from:  ('192.168.43.7', 38628)
b'left\r\n'
server recv from:  ('192.168.43.7', 38630)
b'left\r\n'
server recv from:  ('192.168.43.7', 38632)
b'left\r\n'
```

```
> frontend2@1.0.0 start
> electron .

gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend2 $ npm start

> frontend2@1.0.0 start
> electron .

gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend2 $ npm start

> frontend2@1.0.0 start
> electron .

^C/home/gonzalo/projects/iot-labs/iot-lab-2/frontend2/node_modules/electron/dist/electron exited with s
INT
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend2 $ npm start

> frontend2@1.0.0 start
> electron .

^C
gonzalo@raspberrypi:~/projects/iot-labs/iot-lab-2/frontend2 $ npm start

> frontend2@1.0.0 start
> electron .
```

CS498: IoT -- Lab 2

left Sumbit

- Car Direction:
- Speed: 0.0
- Distance Traveled: 0.0
- Temperature of the Pi: 0.0

Bluetooth return value: left

Conclusions

The Bluetooth part was a bit challenging essentially due to the problems of the pybluez libraries in Windows and the lack of an easy way of running a virtual operating system that just recognizes the Bluetooth device of the host. Other than that, the Bluetooth protocol didn’t present great challenges, it was just a matter of running the client and server programs

a few times -and maybe struggling a bit with the bluetoothctl tool to pair, connect, disconnect, etc- till find the point where the connection is established.
The wifi part wasn’t challenging at all. And finally the web frontend part it was a bit tricky essentially due to the behavior of ElectronJS.