WES 269 Co-design Hardware & Software – Pannuto	**Ricardo Lizárraga** rilizarraga@ucsd.edu PID: A69028483	**Submit to:** (Prof) **Patrick William Pannuto** (TA) **Anthony Tarbinian** (TA) **Samir Rashid**

Post-Lab 4: WiFi

What to submit?

Please use this document as a template, add your responses directly, and export it as a PDF to Gradescope. Each group should submit one post-lab.

Group name:	**Ricardo Lizarraga**
Team member names:	**Ricardo Lizarraga**
Link to GitHub repository:	https://github.com/RicardoUCSD/Postlab4_WiFi

5. Scan for WiFi networks

TASK: Demonstrate ability to scan networks

1. **Show me the terminal output from a scan**

```
Hard resetting via RTS pin...
--- Terminal on COM17 | 115200 8-N-1
--- Available filters and text transformations: colorize, debug, default, direct, esp32_exception_decoder, hexlify, log2file, nocontrol, printable, send_on_en
ter, time
--- More details at https://bit.ly/pio-monitor-filters
--- Quit: Ctrl+C | Menu: Ctrl+T | Help: Ctrl+T followed by Ctrl+H
Setup done
Scan start
Scan done
18 networks found
Nr | SSID                             | RSSI | CH | Encryption
 1 | tock_tutorial                    |  -45 |  5 | open
 2 | Roselab_2.4G                     |  -50 |  2 | WPA2
 3 | UCSD-DEVICE                      |  -59 |  1 | WPA2
 4 | UCSD-GUEST                       |  -59 |  1 | open
 5 | UCSD-PROTECTED                   |  -59 |  1 | WPA2-EAP
 6 | eduroam                          |  -60 |  1 | WPA2-EAP
 7 | eduroam                          |  -63 |  6 | WPA2-EAP
 8 | UCSD-DEVICE                      |  -63 |  6 | WPA2
 9 | UCSD-GUEST                       |  -63 |  6 | open
10 | UCSD-PROTECTED                   |  -63 |  6 | WPA2-EAP
11 | UCSD-DEVICE                      |  -67 | 11 | WPA2
12 | eduroam                          |  -67 | 11 | WPA2-EAP
13 | UCSD-GUEST                       |  -67 | 11 | open
14 | UCSD-PROTECTED                   |  -67 | 11 | WPA2-EAP
15 | eduroam                          |  -84 | 11 | WPA2-EAP
16 | UCSD-GUEST                       |  -84 | 11 | open
17 | MacbookPro's Wi-Fi Network       |  -86 |  6 | WPA2
18 | DIRECT-e0-HP M252 LaserJet       |  -88 |  6 | WPA2
```

2. **Commit your `wifi-scanner` code to your shared repo**

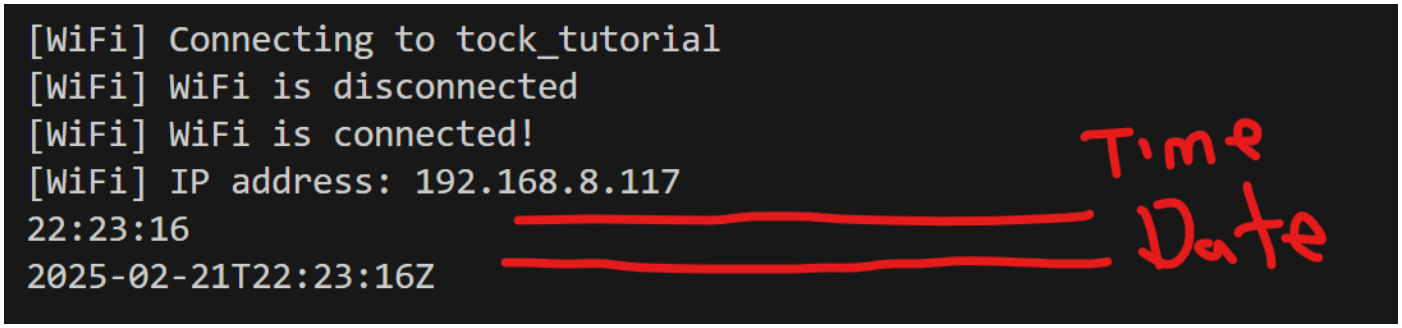
   https://github.com/RicardoUCSD/Postlab4_WiFi

   wifi-scanner _main.cpp

6. Connect to the Internet

TASK: Demonstrate ability to get time through the Internet

1. **Show me the terminal output printing the current date AND time**



2. **Commit your `wifi-client` code to your shared repo**

https://github.com/RicardoUCSD/Postlab4_WiFi

wifi-client _main.cpp

7. Scan Promiscuously

MAC ADDRESS of the client device:

[WiFi] Connecting to tock_tutorial
[WiFi] WiFi is disconnected
[WiFi] WiFi is connected!
[WiFi] IP address: 192.168.8.117
[WiFi] MAC Address: **34:CD:B0:3D:F7:40**
23:14:18
2025-02-21T23:14:18Z

TASK: Demonstrate ability to capture packets from your other device

1. **Show me the terminal output printing some packet metadata**

```
[WiFi] IP address: 192.168.8.117
[WiFi] MAC Address: 34:CD:B0:3D:F7:40
23:31:34
2025-02-21T23:31:34Z
[WiFi] IP address: 192.168.8.117
[WiFi] MAC Address: 34:CD:B0:3D:F7:40
23:31:35
2025-02-21T23:31:35Z
[WiFi] IP address: 192.168.8.117
[WiFi] MAC Address: 34:CD:B0:3D:F7:40
23:31:36
2025-02-21T23:31:36Z
[WiFi] IP address: 192.168.8.117
[WiFi] MAC Address: 34:CD:B0:3D:F7:40
23:31:37
2025-02-21T23:31:37Z
[WiFi] IP address: 192.168.8.117
[WiFi] MAC Address: 34:CD:B0:3D:F7:40
23:31:38
2025-02-21T23:31:38Z
[WiFi] IP address: 192.168.8.117
[WiFi] MAC Address: 34:CD:B0:3D:F7:40
23:31:39
2025-02-21T23:31:39Z
```

```
PACKET TYPE=MGMT, CHAN=05, RSSI=-51, ADDR1=50:41:1c:8a:c0:e8, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=DATA, CHAN=05, RSSI=-60, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3b:e6:3c, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-49, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=DATA, CHAN=05, RSSI=-60, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3b:e6:3c, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=DATA, CHAN=05, RSSI=-44, ADDR1=34:cd:b0:3b:e6:3c, ADDR2=7a:d3:b8:29:05:84, ADDR3=94:83:c4:58:bf:91
PACKET TYPE=MGMT, CHAN=05, RSSI=-51, ADDR1=90:de:80:46:14:41, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-50, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-89, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=90:de:80:45:e6:cf, ADDR3=ff:ff:ff:ff:ff:ff
PACKET TYPE=MGMT, CHAN=05, RSSI=-88, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=90:de:80:45:e6:cf, ADDR3=ff:ff:ff:ff:ff:ff
PACKET TYPE=DATA, CHAN=05, RSSI=-61, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3b:e6:3c, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-52, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-64, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=90:de:80:46:07:f1, ADDR3=ff:ff:ff:ff:ff:ff
PACKET TYPE=MGMT, CHAN=05, RSSI=-66, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=90:de:80:46:07:f1, ADDR3=ff:ff:ff:ff:ff:ff
PACKET TYPE=MGMT, CHAN=05, RSSI=-71, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=5e:b3:32:65:2e:b2, ADDR3=ff:ff:ff:ff:ff:ff
PACKET TYPE=MGMT, CHAN=05, RSSI=-71, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=5e:b3:32:65:2e:b2, ADDR3=ff:ff:ff:ff:ff:ff
PACKET TYPE=MGMT, CHAN=05, RSSI=-50, ADDR1=ec:2e:98:1c:b8:e1, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-50, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-53, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-53, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-55, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-55, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-52, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=DATA, CHAN=05, RSSI=-46, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3d:f7:40, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=DATA, CHAN=05, RSSI=-50, ADDR1=34:cd:b0:3d:f7:40, ADDR2=7a:d3:b8:29:05:84, ADDR3=94:83:c4:58:bf:91
PACKET TYPE=MGMT, CHAN=05, RSSI=-52, ADDR1=34:cd:b0:3d:f7:40, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-52, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-52, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-52, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-52, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
```

```
PACKET TYPE=DATA, CHAN=05, RSSI=-60, ADDR1=7a:d3:b8:29:05:84, ADDR2=2c:cf:67:6d:c9:40, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-49, ADDR1=40:ec:99:85:97:01, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-51, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-51, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=DATA, CHAN=05, RSSI=-65, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3b:b1:90, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=DATA, CHAN=05, RSSI=-68, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3b:b1:90, ADDR3=94:83:c4:58:bf:91
PACKET TYPE=DATA, CHAN=05, RSSI=-64, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3b:c0:f8, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-50, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=DATA, CHAN=05, RSSI=-64, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3b:c0:f8, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=DATA, CHAN=05, RSSI=-59, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3b:e6:3c, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-49, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-49, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-62, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=90:de:80:46:07:f1, ADDR3=ff:ff:ff:ff:ff:ff
PACKET TYPE=MGMT, CHAN=05, RSSI=-63, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=90:de:80:46:07:f1, ADDR3=ff:ff:ff:ff:ff:ff
PACKET TYPE=MGMT, CHAN=05, RSSI=-51, ADDR1=34:cd:b0:3d:f7:40, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
Found!!!
PACKET TYPE=MGMT, CHAN=05, RSSI=-51, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3d:f7:40, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-52, ADDR1=34:cd:b0:3d:f7:40, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
Found!!!
PACKET TYPE=MGMT, CHAN=05, RSSI=-51, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3d:f7:40, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=MGMT, CHAN=05, RSSI=-51, ADDR1=ff:ff:ff:ff:ff:ff, ADDR2=7a:d3:b8:29:05:84, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=DATA, CHAN=05, RSSI=-31, ADDR1=7a:d3:b8:29:05:84, ADDR2=34:cd:b0:3d:f7:40, ADDR3=7a:d3:b8:29:05:84
PACKET TYPE=DATA, CHAN=05, RSSI=-50, ADDR1=34:cd:b0:3d:f7:40, ADDR2=7a:d3:b8:29:05:84, ADDR3=94:83:c4:58:bf:91
```

2. **Commit your `wifi-promiscuous` code to your shared repo**

   https://github.com/RicardoUCSD/Postlab4_WiFi
   wifi-promiscuous_main.cpp

TASK: Determine some information about how the ESP32 client is connected

3. **Which WiFi PHY protocol is it using to connect to the WiFi network? (e.g., 802.11b, 802.11g, 802.11n, etc.)**

802.11g

Extracting the protocol version

```
const wifi_promiscuous_pkt_t *pkt2 = (wifi_promiscuous_pkt_t *)buf;
const uint8_t *payload = pkt2->payload;
if (payload == NULL) return;
uint8_t frame_control = payload[0];
uint8_t protocol_version = frame_control & 0x03;
printf("Protocol Version: %d ", protocol_version);
// Determine the standard based on packet information
if (pkt2->rx_ctrl.rate < 4) {
    printf("(802.11b)\n");
} else if (pkt2->rx_ctrl.sig_mode == 0) {
    printf("(802.11g)\n");
} else {
    printf("(802.11n)\n");
}
```

```
PACKET TYPE=DATA, CHAN=08, RSSI=-33, ADDR1=c0:d7:aa:dd:a1:d1, ADDR2=34:cd:b0:3d:f7:40, ADDR3=c0:d7:aa:dd:a1
Protocol Version: 0 (802.11g)
```

4. **Which channel is it connected on?**

   Is connected to channel #5 (This was during lab practice, but at home, after reconfiguring the device for my local WiFi LAN, it was connected to channel 8)

   20 MHz

HERE, I resume this section at **home**, then I reconfigured devices for my local wifi network:

Nr	SSID	RSSI	CH	Encryption
1	KOAyLANI	-55	8	WPA2
2	KOAyLANI_2.4GEXT	-65	8	WPA2
3	KOAyLANI_NetgearN300_EXT	-66	8	WPA2
4	Dadusethisone	-78	11	WPA2
5	NETGEAR25	-81	2	WPA2
6	SETUP-D4CB	-82	6	WPA2
7	SETUP-9237	-87	1	WPA2
8	Martinezhousehold	-89	1	WPA2
9	ATTMmvGGDS	-89	2	WPA2
10	ATT6mUY4x7	-93	1	WPA2

5. **How much bandwidth is it using on that channel, 20 MHz or 40 MHz?**

```
//To Determine version & BW
const wifi_promiscuous_pkt_t *pkt2 = (wifi_promiscuous_pkt_t *)buf;
const uint8_t *payload = pkt2->payload;
if (payload == NULL) return;
uint8_t frame_control = payload[0];
uint8_t protocol_version = frame_control & 0x03;
printf("Protocol Version: %d ", protocol_version);
// Determine the standard based on packet information
if (pkt2->rx_ctrl.rate < 4) {
    printf("(802.11b)\n");
} else if (pkt2->rx_ctrl.sig_mode == 0) {
    printf("(802.11g)\n");
} else {
    printf("(802.11n)\n");
}

// To determine BW,  either 20 or 40MHz
printf("bandwidth: %s\n", pkt2->rx_ctrl.cwb ? "40MHz" : "20MHz");
```

BW

```
PACKET TYPE=DATA, CHAN=08, RSSI=-49, ADDR1=c0:d7:aa:dd:a1:d1, ADDR2=4e:f2:cf:7a:0b:7f, ADDR3=c0:d7:aa:dd:a1:d1
Protocol Version: 0 (802.11b)
bandwidth: 20MHz
```

8. Host a WiFi network and connect to it

C:_RLS_GoogleDrive\UCSDWESClass_WES 269 - Co-design Hardware & Software - Pannuto [WI25]\Lab 4 Wifi\wifi-starter-main\wifi-starter-main\wifi-access-point\src

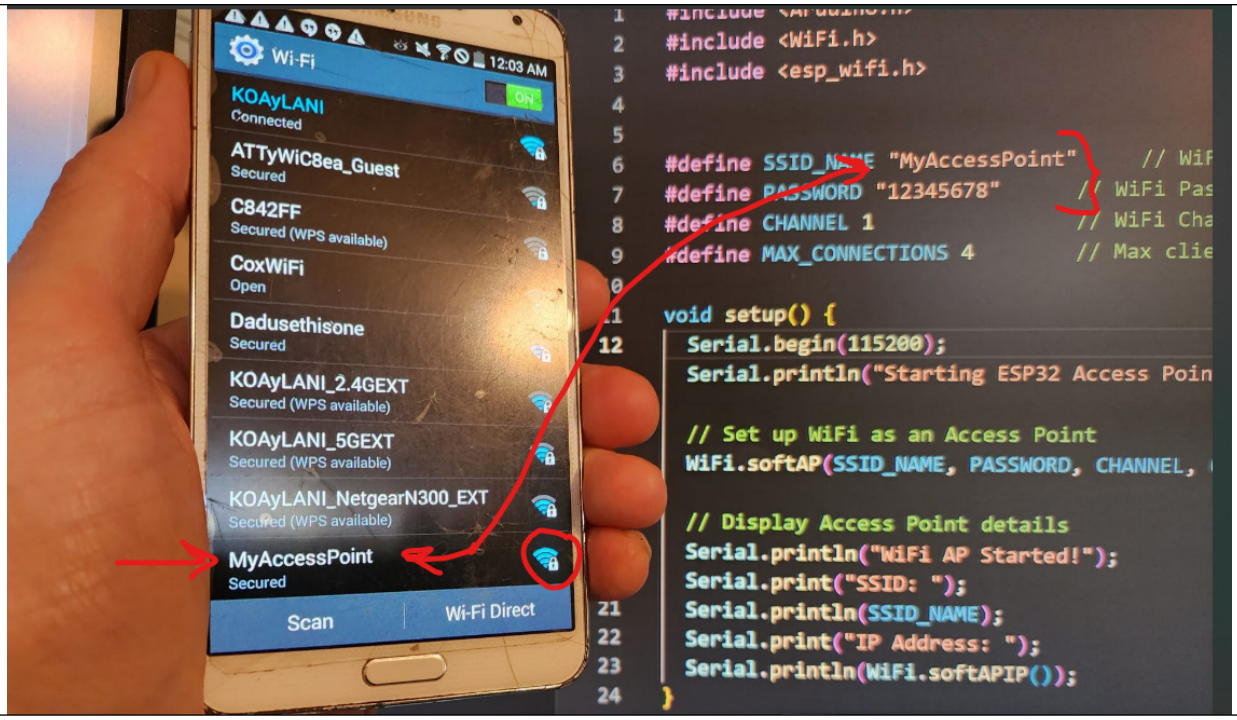
TASK: Demonstrate creation of an Access Point

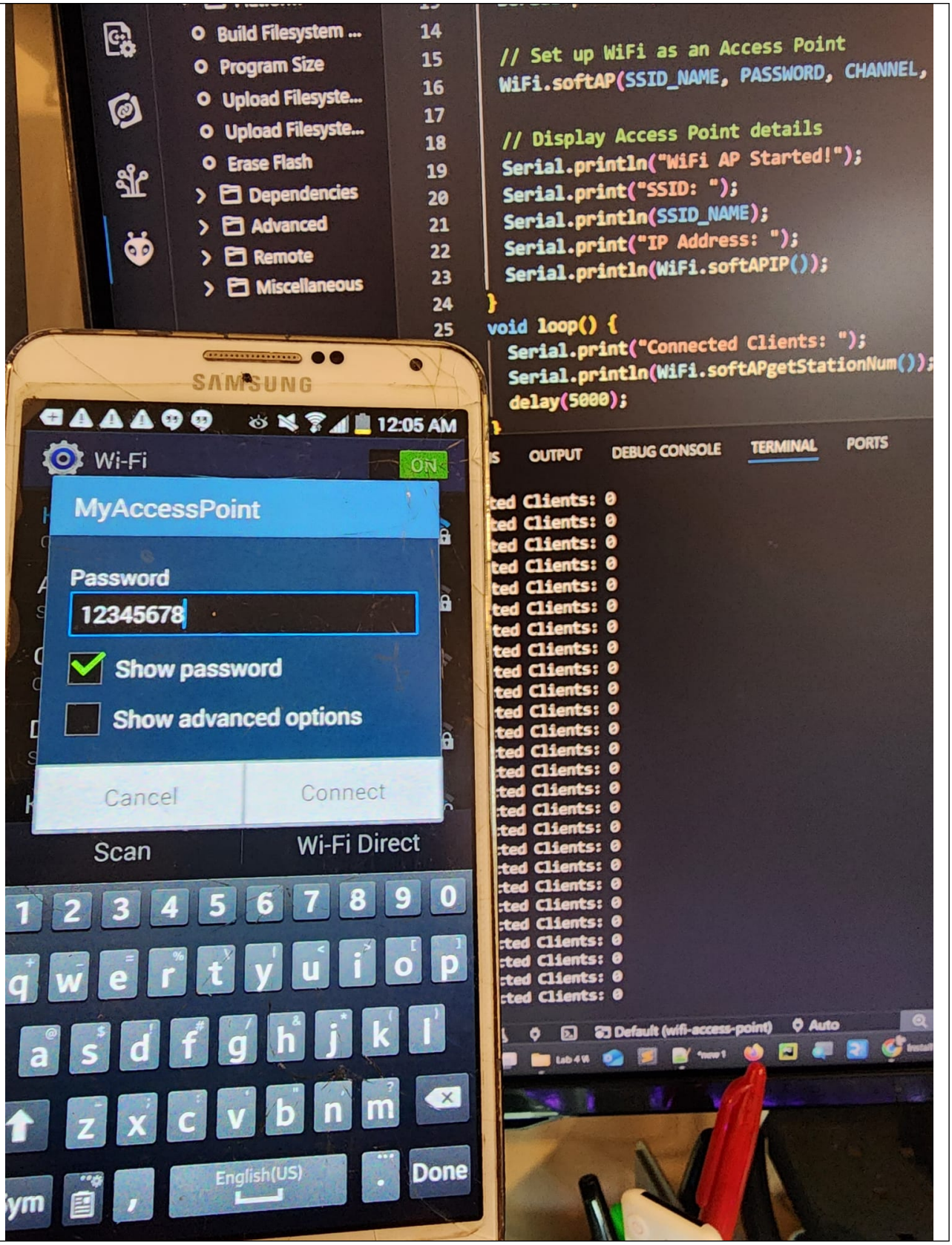
```
#define SSID_NAME "MyAccessPoint”// WiFi SSID
#define PASSWORD "12345678"      // WiFi Password (Minimum 8 characters)
#define CHANNEL 1                // WiFi Channel
#define MAX_CONNECTIONS 4        // Max clients allowed
```

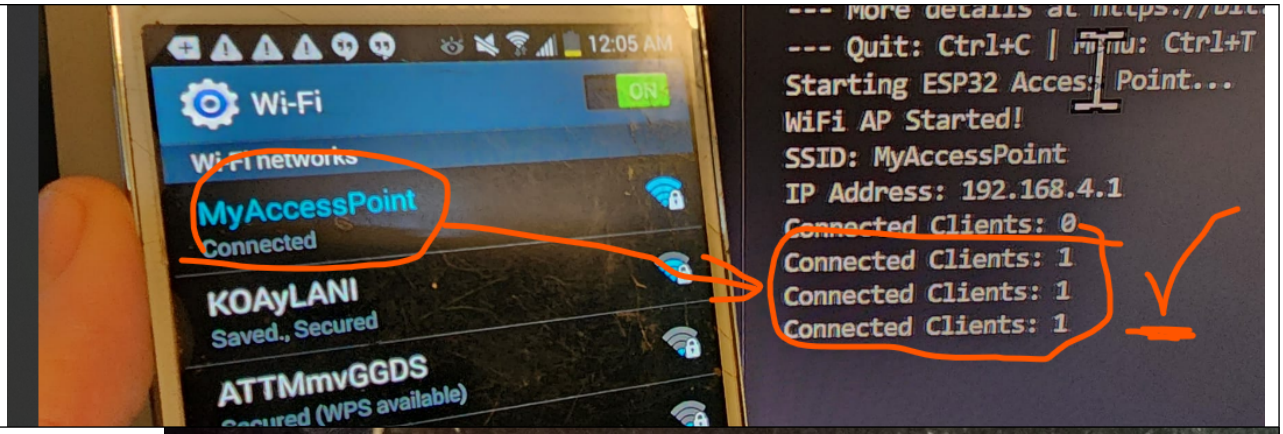
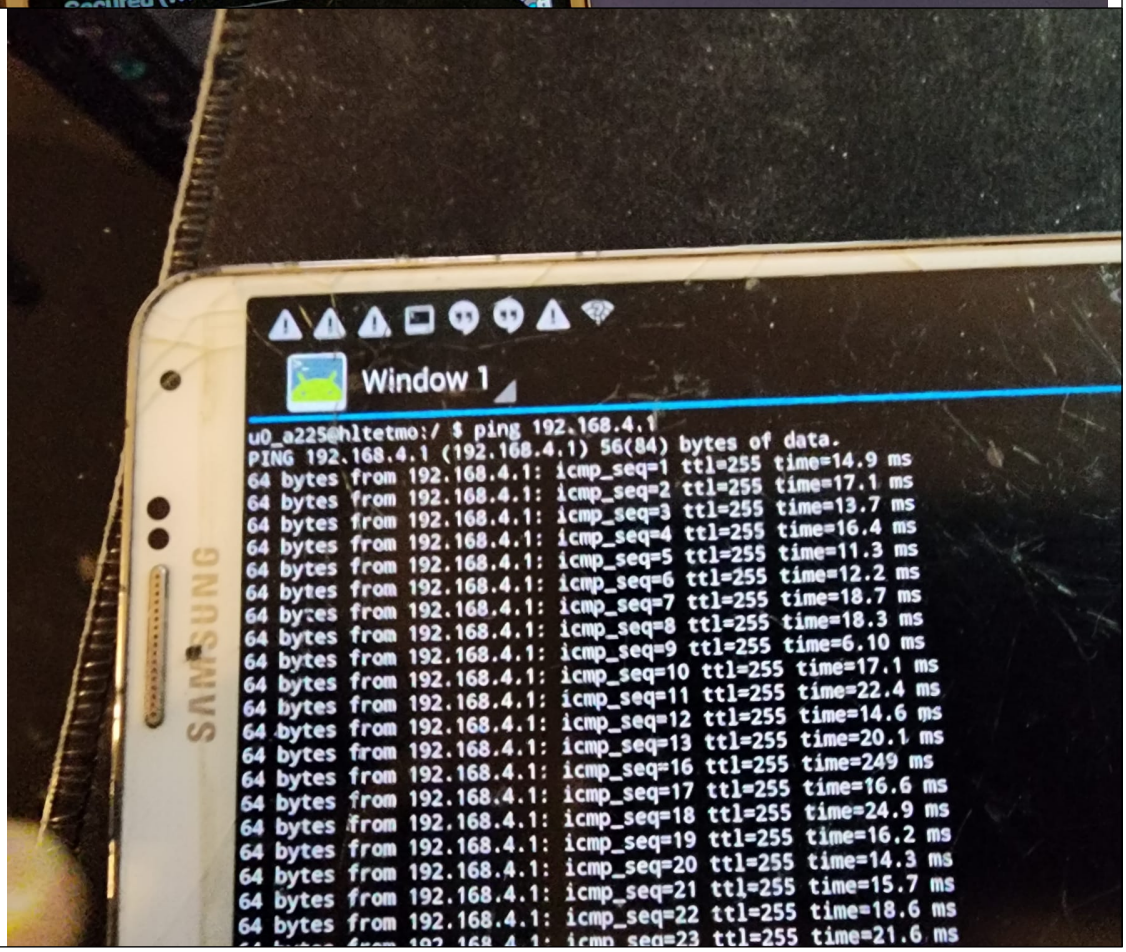
```
#define SSID_NAME "MyAccessPoint"     // WiFi SSID
#define PASSWORD "12345678"      // WiFi Password (Minimum 8 characters)
#define CHANNEL 1                // WiFi Channel
#define MAX_CONNECTIONS 4        // Max clients allowed
```

1. **Show me the terminal output printing data about a connected client**

Access point ready to grant remote client connections	`--- Terminal on COM17 | 115200 8-N-1`
	`--- Available filters and text transformations: colorize, debug, default, dire`
	`exception_decoder, hexlify, log2file, nocontrol, printable, send_on_enter, tim`
	`--- More details at https://bit.ly/pio-monitor-filters`
	`--- Quit: Ctrl+C | Menu: Ctrl+T | Help: Ctrl+T followed by Ctrl+H`
	`Starting ESP32 Access Point...`
	`WiFi AP Started!`
	`SSID: MyAccessPoint`
	`IP Address: 192.168.4.1`
	`Connected Clients: 0`

Right before connecting	

Enter password (see below zero clients connected up to this point)	

Successful Connection to access point	
Pinging IP address from client	

2. **Commit your `wifi-access-point` code to your shared repo**
   https://github.com/RicardoUCSD/Postlab4_WiFi
   wifi-access-point_main.cpp

9. Use MQTT across multiple devices

- `wifi-access-point-broker`
  - **C:_RLS\wifi-starter-main\wifi-access-point-broker**
- `wifi-client-publisher`
  - **C:_RLS\wifi-starter-main\wifi-client-publisher**
- `wifi-client-subscriber`
  - **C:_RLS\wifi-starter-main\wifi-client-subscriber**

CHECKOFF: Demonstrate the full working MQTT app across three devices

1. **Show us your MQTT application working.**

```
/** Basic Mqtt Broker
  *
  *  +------------------------------+
  *  | ESP                          |
  *  |       +--------+             |
  *  |       | broker |             | 1883 <--- External client/s
  *  |       +--------+             |
  *  |                              |
  *  +------------------------------+
  *
  *  Your ESP will become a MqttBroker.
       *  You can test it with any client such as mqtt-spy for example
       *
       * Messages are retained *only* if retain > 0
       *
  */
```

```
.Connected to NIWC-Guest, IP address: 192.168.57.65
Broker ready : 192.168.57.65 on port 1883
```

```
/** Simple Client (The simplest configuration, client only sends topics)
  *
  *
  *                  +--------+
  *           +------>| broker |<--- <  Other client
  *           |      +--------+
  *           |
  *   +-----------------+
  *   | ESP   |         |
  *   | +----------+    |
  *   | | internal |    |
  *   | | client   |    |
  *   | +----------+    |
  *   |                 |
  *   +-----------------+
  *
       * 1 - change the ssid/password
       * 2 - change BROKER values (or keep emqx.io test broker)
       * 3 - you can use mqtt-spy to connect to the same broker and
       *     see the sensor/temperature updated by the client.
       *
       * pro  - small memory footprint (both ram and flash)
       *      - very simple to setup and use
       *
       * cons - cannot work without wifi connection
  *     - stop working if broker is down
       *      - local publishes takes more time (because they go outside)
       */
```

```
Simple clients with wifi
....Connected to NIWC-GuestIP address: 192.168.57.87
--> Publishing a new sensor/temperature value: 18.90
--> Publishing a new sensor/temperature value: 18.80
--> Publishing a new sensor/temperature value: 18.80
--> Publishing a new sensor/temperature value: 18.70
--> Publishing a new sensor/temperature value: 18.70
--> Publishing a new sensor/temperature value: 18.70
--> Publishing a new sensor/temperature value: 18.70
```

```
/** TinyMQTT allows a disconnected mode:
  *
  *  +-----------------------------+
  *  | ESP                         |
  *  |                 +--------+  |
  *  |       +-------->| broker |  |
  *  |       |         +--------+  |
  *  |       |             ^       |
  *  |       |             |       |
  *  |       v             v       |
  *  | +----------+  +----------+  |
  *  | | internal |  | internal |  |
  *  | | client   |  | client   |  |
  *  | +----------+  +----------+  |
  *  |                             |
  *  +-----------------------------+
  *
  *  In this example, local clients A and B are talking together, no need to be connected.
  *
  *  A single ESP can use this to be able to comunicate with itself with the power
  *  of MQTT, and once connected still continue to work with others.
  *
  *  The broker may still be conected if wifi is on.
  *
  */
```

```
B is publishing sensor/temperature
--> Client A received msg on topic sensor/temperature, sent by B
--> Client B received msg on topic sensor/temperature, sent by B
A is publishing sensor/temperature
B is publishing sensor/temperature
--> Client A received msg on topic sensor/temperature, sent by B
--> Client B received msg on topic sensor/temperature, sent by B
A is publishing sensor/temperature
--> Client A received msg on topic sensor/temperature, sent by A
--> Client B received msg on topic sensor/temperature, sent by A
B is publishing sensor/temperature
--> Client A received msg on topic sensor/temperature, sent by B
--> Client B received msg on topic sensor/temperature, sent by B
B is publishing sensor/temperature
--> Client A received msg on topic sensor/temperature, sent by B
--> Client B received msg on topic sensor/temperature, sent by B
A is publishing sensor/temperature
--> Client A received msg on topic sensor/temperature, sent by A
--> Client B received msg on topic sensor/temperature, sent by A
```

2. **Write a couple of sentences on what you did and how it worked and show relevant terminal output from devices.**

   Having the **TinyMqtt** dependencies install, it didn't work after the first try, after rebooting the computer, dependencies worked.
   This example exercises publishing and getting read backs from a simulated sensor via the Broker

3. **Commit code for all three applications to your shared repo**

   https://github.com/RicardoUCSD/Postlab4_WiFi

- **MQTT_wifi-access-point-broker_main.cpp**
- **MQTT_wifi-client-publisher_main.cpp**
- **MQTT_wifi-client-subscriber_main.cpp**