Final Project - Suhuan Pan

https://github.com/Suhuan-Pan/CS2600_Final

https://youtu.be/xxZzeYqJqLo

```
// part(1)
#include <WiFi.h>
#include <PubSubClient.h>

// part (2)
#include <ESP32Servo.h>
#include <UltrasonicSensor.h>

#define LED 2


// part(1): global constant variables for local WiFi
const char *ssid = "Be Cool Honey Bunny"; // WiFi name
const char *password = "7608144989";  // WiFi password

// part(1): global constant variables for MQTT Broker(server)
const char *mqtt_broker = "192.168.1.2"; // WiFi local IP
address
const char *topic = "Suhuan_FinalProject";
const char *mqtt_username = "emqx";
const char *mqtt_password = "public";
const int mqtt_port = 1883;

// part (2)
int posVal = 0; // variable to store the servo position
int servoPin = 15; // Servo motor pin


// part(1)
WiFiClient espClient;
PubSubClient client(espClient);

// part(2)
UltrasonicSensor ultrasonic(13, 14);
Servo myservo;  // create servo object to control a servo
// 16 servo objects can be created on the ESP3s

// part(1)
void connectingToWiFi() {
```

```
  // keep trying every half second
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    // Serial.println("Connecting to WiFi..");
  }

  Serial.println("Connected to the WiFi network.");
}


// part(1)
void connectingToBroker() {
  while (!client.connected()) {

    String client_id = "[esp32-client]";

    // esp connects to MQTT broker succeed
    if (client.connect(client_id.c_str(), mqtt_username,
mqtt_password)) {
      Serial.println("Public emqx mqtt broker connected.");
    }

    // failed
    else {
      Serial.print("failed with state ");
      Serial.print(client.state());
      delay(2000);
    }
  }

}


void setup() {

  // 1. Set software serial baud to 115200;
  Serial.begin(115200);

  // part(2) initialization
  int temperature = 22;
  ultrasonic.setTemperature(temperature);

  // initialize output
  pinMode(LED, OUTPUT);

  // 2. esp is connecting to local WiFi network
```

```
  WiFi.begin(ssid, password);
  connectingToWiFi();

  // 3a. esp32 is connecting to a mqtt broker(server)
  client.setServer(mqtt_broker, mqtt_port);

  // 3b. message send back to esp32 (output in serial monitor
under same baud)
  client.setCallback(callback);


  // 4. make esp32 to send / receive message via MQTT broker
  connectingToBroker();


  // 5. publish message and subscribe the same topic to receive
message from MQTT server
  client.publish(topic, "Greetings from ESP32 board.");
  client.subscribe(topic);


}



// 4 what you will see in serial monitor
void callback(char *topic, byte *payload, unsigned int length) {

  Serial.print("Message arrived in topic: ");
  Serial.println(topic);

  String s = "";
  for (int i = 0; i < length; i++) {
    // Serial.print((char) payload[i]);
    s += (char) payload[i];
  }


  if (s == "1") {
    digitalWrite(LED, HIGH);  // high volt means lights on
    delay(2000);
    Serial.println("Press 1 to turn on LED.");

  }

  else if (s == "2") {
    digitalWrite(LED, LOW);  // low volt means lights off
```

```
    delay(2000);
    Serial.println("Press 2 to turn off LED.");
  }

  else if (s == "3") {
    int distance = ultrasonic.distanceInCentimeters();
    Serial.printf("Press 3 to measure distance = %dcm.\n",
distance);
    delay(2000);

    Serial.println("Adjust servo position with the same
value.");
    // part (2)
    myservo.setPeriodHertz(50); // standard 50 hz servo
    myservo.attach(servoPin, 500, 2500);

    myservo.write(distance);
    delay(15);  // wait for 15 ms to reach the position

    myservo.detach();
  }

  else {
    delay(2000);
  }

  Serial.println();
  Serial.println("-----------------------");
}



// 7. run the loop
void loop() {
  client.loop();
}
```

Step 0, go to the director, start MQTT server
```
cd /usr/local/etc/mosquitto
brew services restart mosquitto
```

Step 1. create a MQTT-client subscriber the topic 'Suhuan_FinalProject'
```
mosquitto_sub -h "192.168.1.2" -v -t 'Suhuan_FinalProject' -d
```

Step 2, create a MQTT-client publisher with the same topic to take user input

```
mosquitto_pub -h "192.168.1.2" -t 'Suhuan_FinalProject' -m 'Initial greetings.'
```

Step 3, Arduino script

Step 4, press three to measure distance
```
mosquitto_pub -h "192.168.1.11" -t 'Suhuan_FinalProject' -m '1'

mosquitto_pub -h "192.168.1.2" -t 'Suhuan_FinalProject' -m '2'

mosquitto_pub -h "192.168.1.2" -t 'Suhuan_FinalProject' -m '3'
```

Checking output on MQTT-client publisher
Checking MQTT-client subscriber terminal:

Meanwhile, Checking Arduino Serial Monitor

open the serial monitor
put hands close to ultrasonic to measure distance

Finally, stop service on publisher terminal

```
brew services stop mosquitto
```