# Documentation

*Embedded & Beyond*
*23.06.2022*

## *1.* Team members

Arsal Abbasi
Charles Okere
Muhammad Umer Bin Yaqoob

## 2. Introduction

Many people desire indoor plants, and the number of people purchasing indoor plants increases daily. Indoor plants can be challenging to maintain and difficult to predict when they will pass away. According to the National Gardening Association, people from all walks of life engage in gardening. Indoor plants are extremely popular with people over the age of 20. Germany imported 213.75 million indoor plants in bloom or with buds in 2021. Comparing this to the previous years, there was an increase. To raise and maintain the health of indoor plants, several factors must be monitored, including soil moisture, temperature, and humidity. It is challenging to keep track of all these variables manually. Based on the care of the indoor plants, this paper proposed a low-cost smart pot using some components (Raspberry Pi, Arduino, DC Motor, Light Sensor, Temperature Sensor) that performs some functions like (i) keeping track of the pot's temperature (ii) keeping track of the humidity of the environment. The results of these readings will be sent to the user on android phone and raspberry pi using mqtt. The user will be able to change the plant`s location through one message sent by android phone to the suitable location where it can get required temperature and humidity for its growth. The project's primary goal is to relieve the user of daily maintenance responsibilities for the plant and improve living conditions for optimal plant growth while minimizing equipment costs.

The primary goal of this investigation is to assist people who keep plants inside with the care of their plants. Our research focuses on two primary domains.

- Integrating intelligent solutions into the chamber will allow the user to monitor the plant environment.
- An intelligent application for the user that will let user control the plant’s location just by sending one message.

As a result of the motivation, we specify certain requirements with component-level specifications. Following is a description of the primary component of this project, along with its requirement specifications in Figure 2.

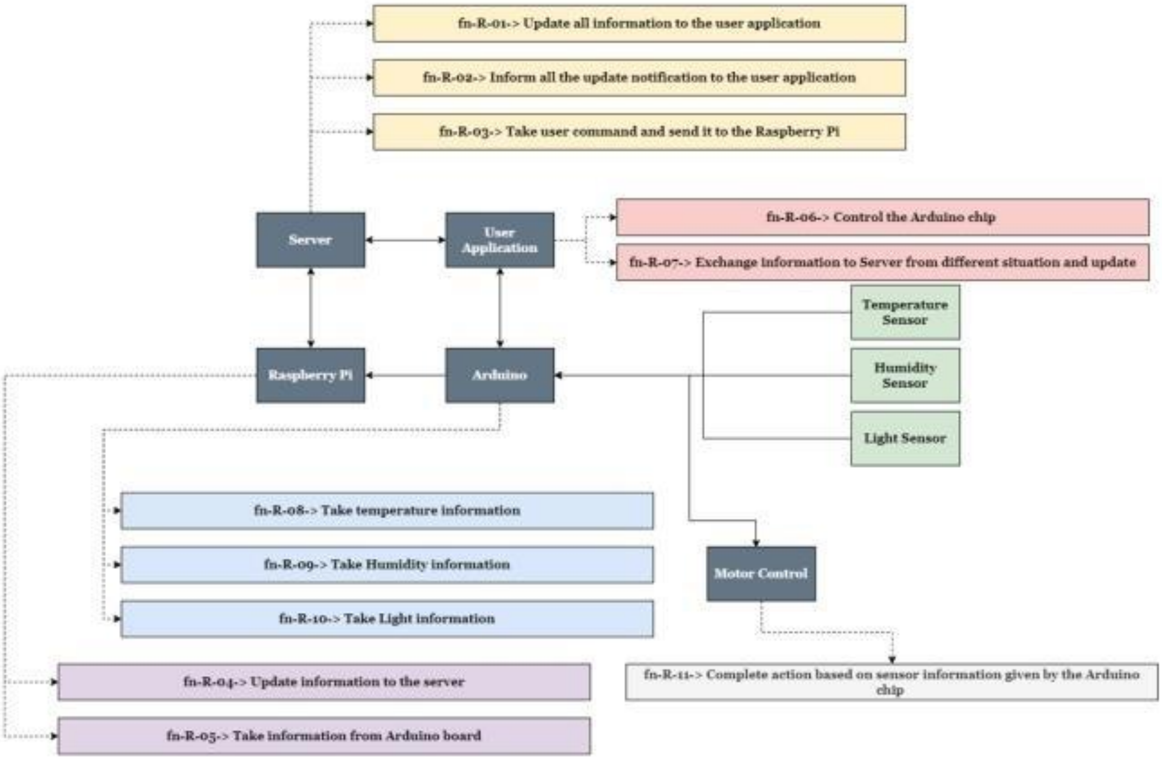

fn-R-01-> Update all information to the user application
fn-R-02-> Inform all the update notification to the user application
fn-R-03-> Take user command and send it to the Raspberry Pi
Server
User Application
fn-R-06-> Control the Arduino chip
fn-R-07-> Exchange information to Server from different situation and update
Temperature Sensor
Humidity Sensor
Light Sensor
Raspberry Pi
Arduino
fn-R-08-> Take temperature information
fn-R-09-> Take Humidity information
fn-R-10-> Take Light information
Motor Control
fn-R-04-> Update information to the server
fn-R-11-> Complete action based on sensor information given by the Arduino chip
fn-R-05-> Take information from Arduino board

Fig.2 SmartPot requirement diagram

**Sever:** The Server will maintain the information flow and provide monitoring notifications to the system. It will also receive user command information and send it to the Raspberry Pi.

**User application:** The User Application can control the Arduino chip and send information and complete actions to the Server via the data exchange process.

**Raspberry Pi:** The Raspberry Pi will send all of the system sensor data to the Server that was collected from the Arduino chip. Arduino: The User Application can control it, and it will also send sensor data to the Raspberry Pi.

- **Temperature Sensor**
- **Humidity Sensor**

**Motor Control:** The User Application will control the motor based on sensor data.

## 3. Concept description

The hardware that will be used for this project includes; Arduino Uno Wifi, Raspberry Pi, DHT sensor and DC motor. DHT sensor will be used to take readings of the temperature and humidity. Raspberry pi , Arduino uno wifi, and android phone will be connected using mqtt. It will be acting as mqtt subscriber and publisher. When the readings will be taken the readings will be published on raspberry pi and androind phone. Android phone will also be used as a publisher. When the user is notified about the readings, he will have an option to send a message to Arduino which will turn on the DC motor and the location of the plant will be changed to somewhere where it can get suitable temperature and humidity.
The basic operating principle of this procedure depends on the data collected by DHT sensor inside the smart pot. Finally, the input is examined, and action is taken. As a result, maintain the plant's health to complete the process successfully. Basic functions of our smartphone are summed up in the block diagram below represented in Fig.1.

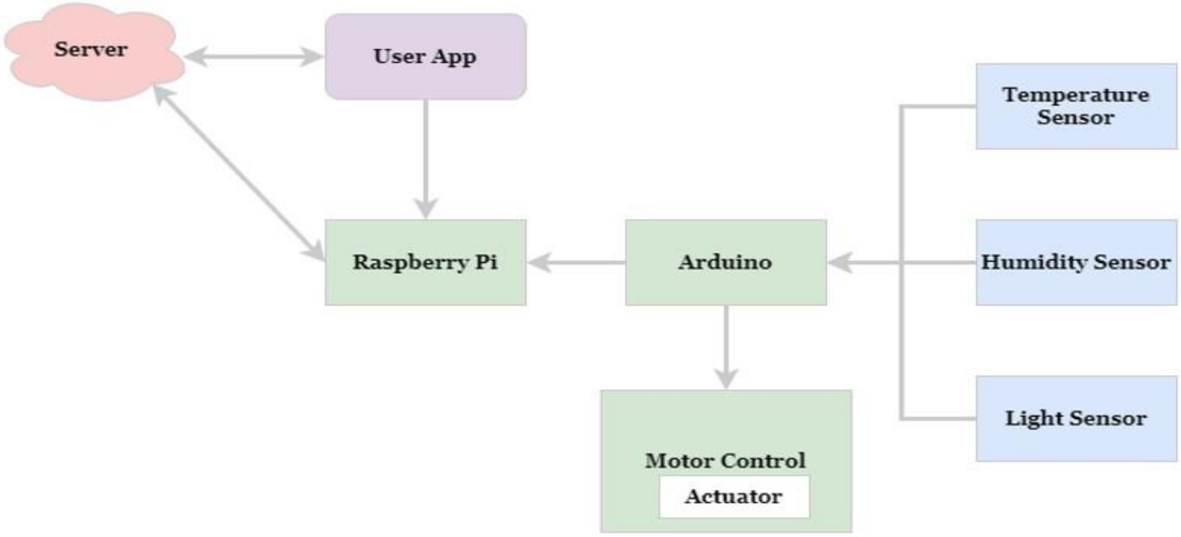


*Fig.1 Block Diagram*

*This smartpot is designed for the people who want to take care of their plats and monitor the environment to choose what's best of their plants. For this application sensors used are; DHT sensor and Light sensor, the hardware used will be Raspberry pi, Arduino uno wifi and a DC motor.*

## 4. Project/Team management

*Initially we came up with different ideas and after some research we decided for smart pot. We worked on this project, did research and brainstormed with different ideas. We had problems with coordinating as a team, but after some corporation we were able to overcome the problems. We helped each other and worked on the tasks.*

***Charles Okere***

- ***Worked on Block Diagram, requirement Diagram and Class Diagram.***
- ***Introduction in Documentation***
- ***Concept description in Documentation***

- ***Implementation in Documentation***
- ***Use Case Diagram***

***Arsal Abbasi***

- ***Code for Arduino Uno wifi***
- ***Code for Mqtt***
- ***Established connection between Arduino, raspberry pi and android phone***
- ***Established the hardware as mqtt subscriber and publisher***
- ***Solved the problems with initial code***
- ***Implementation of the project***
- ***Enabling mosquito***
- ***Enabling mosquito***
- ***Concept description in documentation***
- ***Technologies in documentation***
- ***Implementation in documentation***
- ***Use case in documentation***

***Muhammad Umer Bin Yaqoob***

- ***Code for Arduino Uno wifi***
- ***Worked on Block Diagram***
- ***Code for Mqtt***
- ***Established connection between Arduino, raspberry pi and android phone***
- ***Established the hardware as mqtt subscriber and publisher***
- ***Solved the problems with initial code***
- ***Implementation of the project***
- ***Enabling mosquito***
- ***Enabling mosquito***
- ***Concept description in documentation***
- ***Technologies in documentation***
- ***Implementation in documentation***
- ***Use case in documentation***

*Due to unavailability of Light sensor in university, it`s not included in the implementation.*

## 5. Technologies

We are using the following for our project

- Arduino Uno Wi-Fi Rev 2
- DHT 11 sensor (Temperature and humidity)
- Raspberry Pi
- DC Motor

The communication protocol that we are using is MQTT protocol which is a open messaging protocol that provides resource-constrained network clients with a simple way to distribute messages in a publish/subscribe communication pattern. The MQTT client establishes a TCP connection to the broker. MQTT has some major advantages over competing protocol for example efficient data transmission and easy to setup. Has a low network usage and efficient distribution of data.
The programing language that we are using is C++.

## 6.Implementation

*The static structure of the environment is;*

- Arduino schedules and manages the process of fetching sensor data.
- The DHT sensor and light sensor run to gather the data values.
- As soon as the Arduino uno wifi receives the data, it sends the utilizing the Raspberry Pi to upload data values to the Server to help establish a connection.
- An application on the user's smart device downloads or receives data from through mqtt. For example, the Android application is linked to the Smart Pot's plant database, and the Arduino is connected to the Raspberry Pi through a server to analyze the app's data.
- Taking action will cause the Arduino to run the motor and carry out the work.
- If any problems arise, users can also control the motor manually by sending the message using android phone.

The SMART POT project's complete concept, which centers on the idea of controlling the entire system through a user application in which the user can fill out all of the system specifications, is used to create the class diagram in Figure 3.

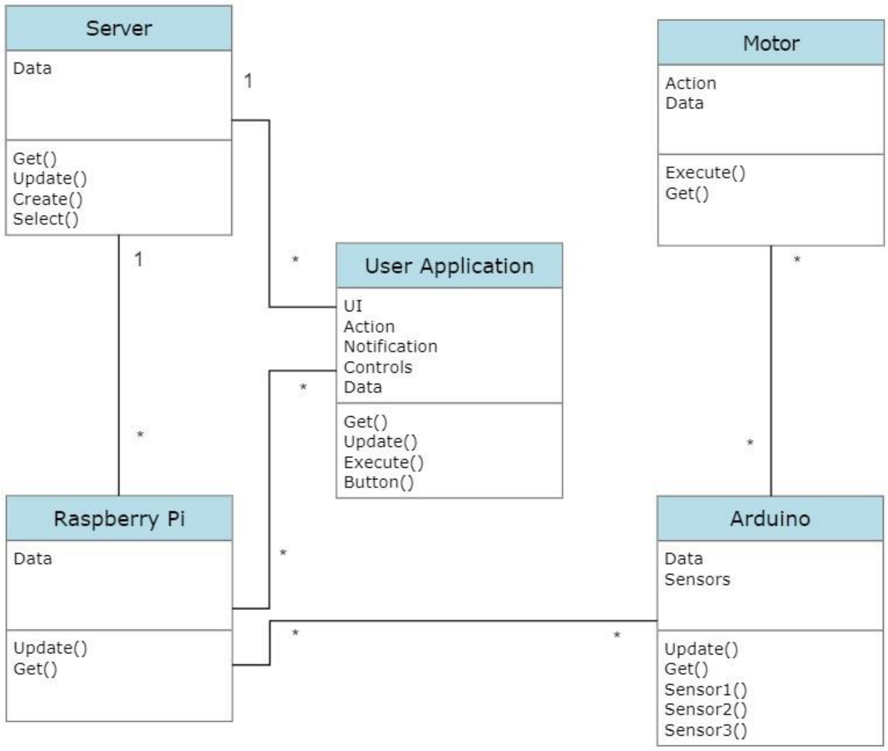
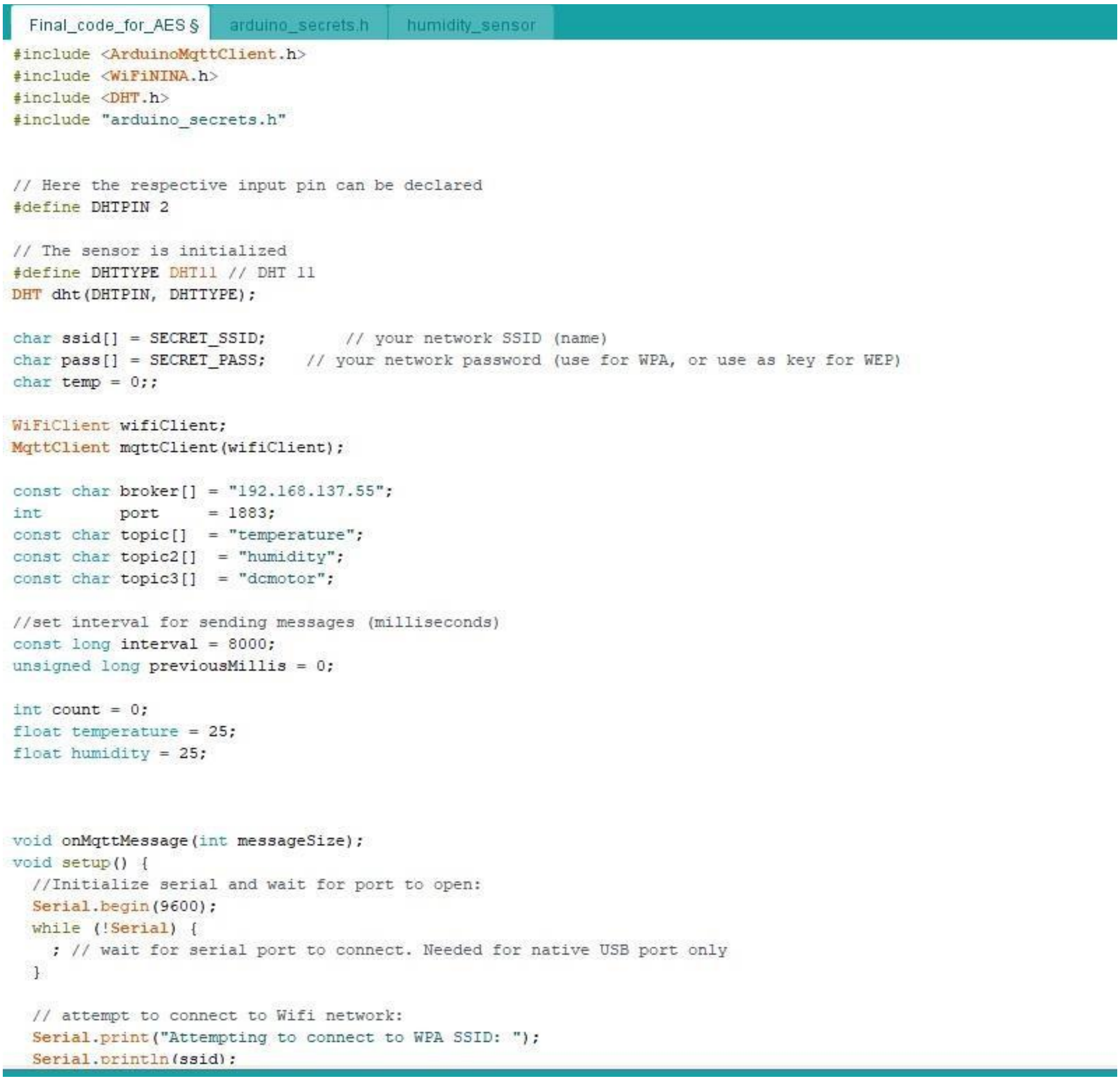


Fig. 3 Class Diagram

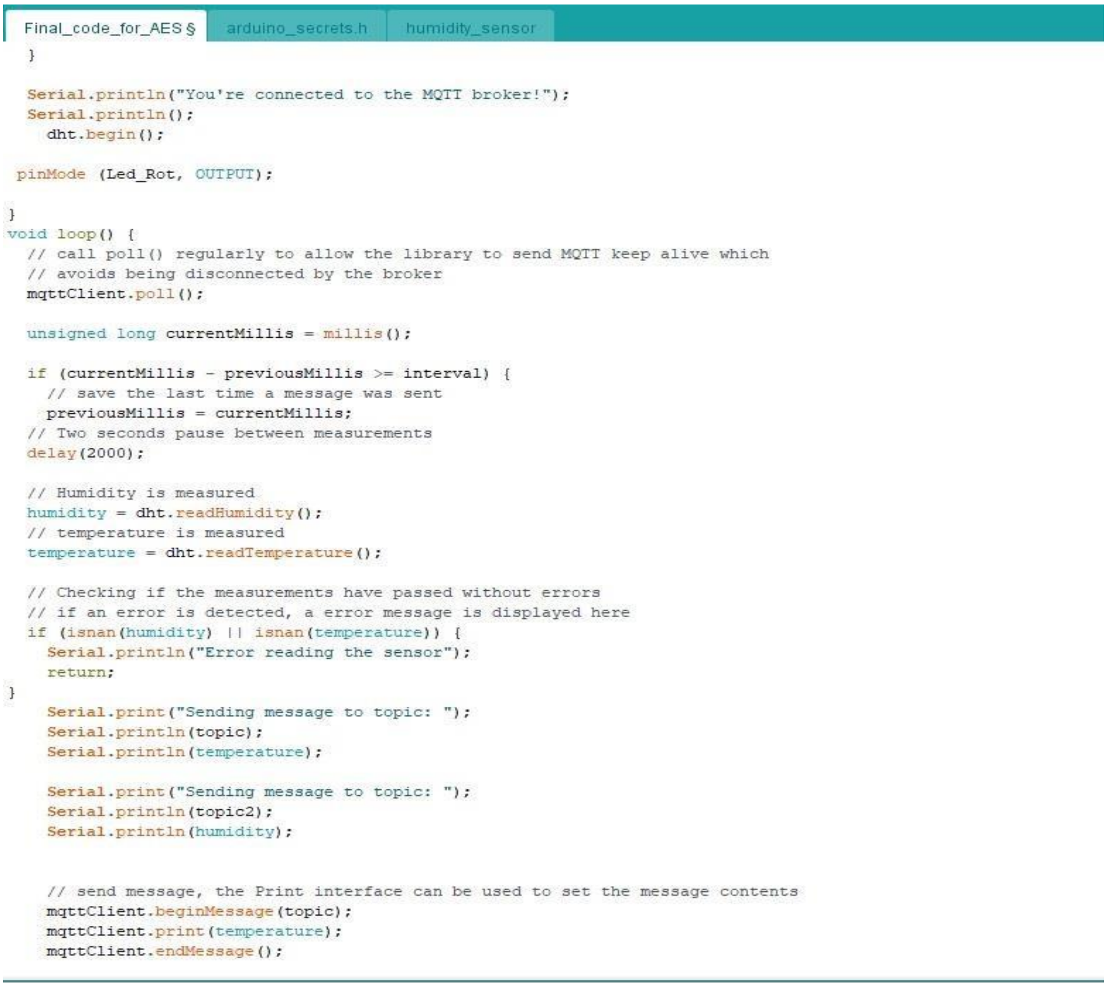
The actor in our smart pot will be the user. He will be able to monitor the temperature, humidity and light level readings. Based on these readings user will have the autgority to decide if he wants to change the position of the position by turning the DC motor on. To turn the DC motor on user will have to send a message using his android phone. Hence, the android phone will not only be working as mqtt subscriber but also as mqtt publisher.

Following is the code that we wrote for all the process that is being done in our smartpot system.



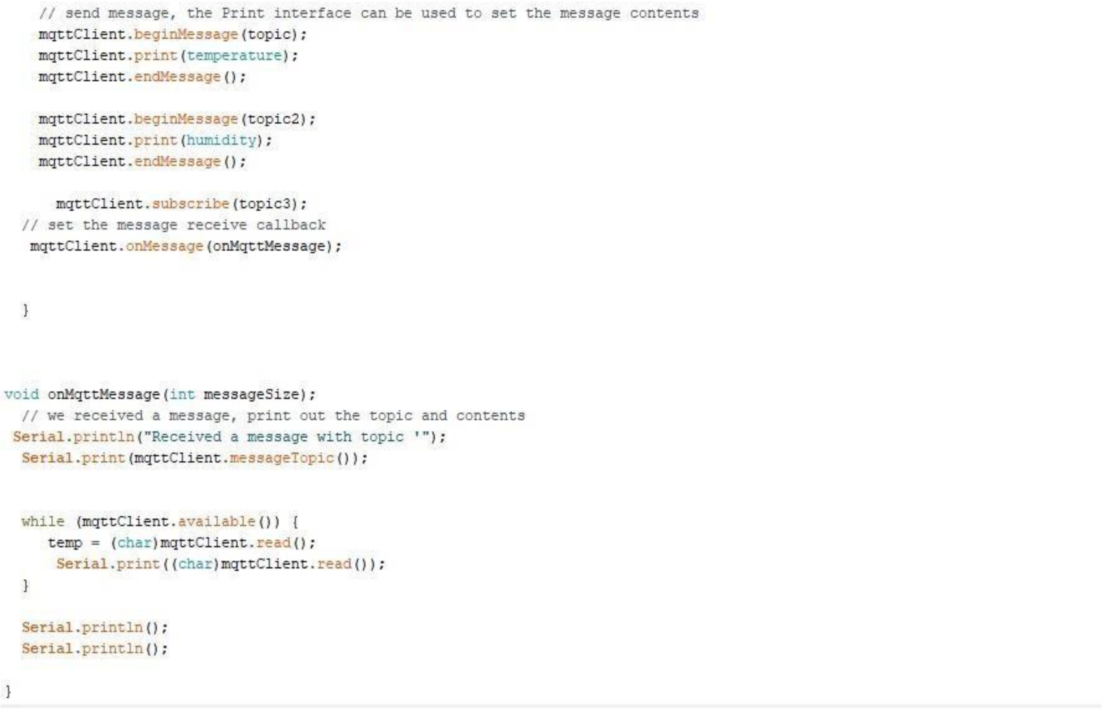
```
#include <ArduinoMqttClient.h>
#include <WiFiNINA.h>
#include <DHT.h>
#include "arduino_secrets.h"


// Here the respective input pin can be declared
#define DHTPIN 2

// The sensor is initialized
#define DHTTYPE DHT11 // DHT 11
DHT dht(DHTPIN, DHTTYPE);

char ssid[] = SECRET_SSID;        // your network SSID (name)
char pass[] = SECRET_PASS;    // your network password (use for WPA, or use as key for WEP)
char temp = 0;;

WiFiClient wifiClient;
MqttClient mqttClient(wifiClient);

const char broker[] = "192.168.137.55";
int        port     = 1883;
const char topic[]  = "temperature";
const char topic2[]  = "humidity";
const char topic3[]  = "dcmotor";

//set interval for sending messages (milliseconds)
const long interval = 8000;
unsigned long previousMillis = 0;

int count = 0;
float temperature = 25;
float humidity = 25;



void onMqttMessage(int messageSize);
void setup() {
  //Initialize serial and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }

  // attempt to connect to Wifi network:
  Serial.print("Attempting to connect to WPA SSID: ");
  Serial.println(ssid);
```

Fig. 4 Code



```
  }

  Serial.println("You're connected to the MQTT broker!");
  Serial.println();
    dht.begin();

 pinMode (Led_Rot, OUTPUT);

}
void loop() {
  // call poll() regularly to allow the library to send MQTT keep alive which
  // avoids being disconnected by the broker
  mqttClient.poll();

  unsigned long currentMillis = millis();

  if (currentMillis - previousMillis >= interval) {
    // save the last time a message was sent
    previousMillis = currentMillis;
  // Two seconds pause between measurements
  delay(2000);

  // Humidity is measured
  humidity = dht.readHumidity();
  // temperature is measured
  temperature = dht.readTemperature();

  // Checking if the measurements have passed without errors
  // if an error is detected, a error message is displayed here
  if (isnan(humidity) || isnan(temperature)) {
    Serial.println("Error reading the sensor");
    return;
}
    Serial.print("Sending message to topic: ");
    Serial.println(topic);
    Serial.println(temperature);

    Serial.print("Sending message to topic: ");
    Serial.println(topic2);
    Serial.println(humidity);


    // send message, the Print interface can be used to set the message contents
    mqttClient.beginMessage(topic);
    mqttClient.print(temperature);
    mqttClient.endMessage();
```

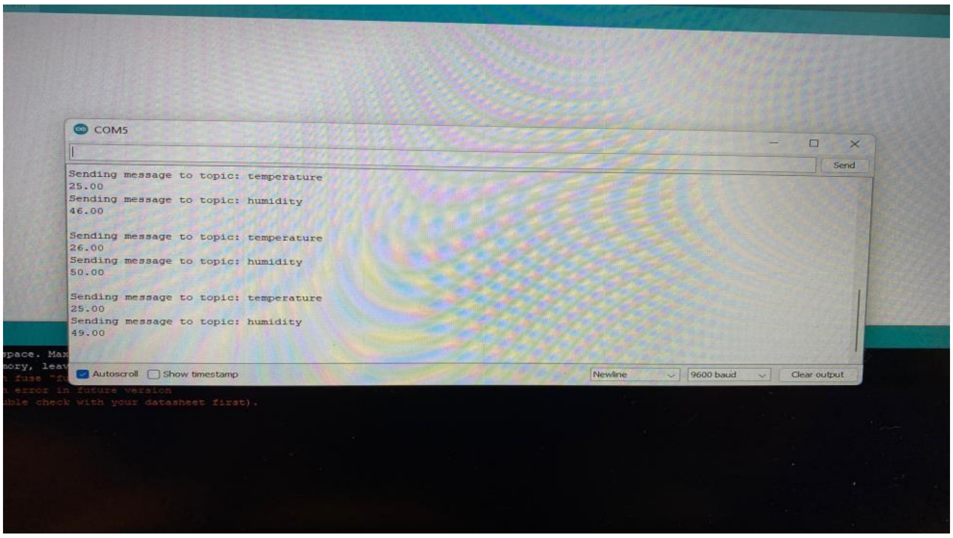
Fig. 5 Code



```
    // send message, the Print interface can be used to set the message contents
    mqttClient.beginMessage(topic);
    mqttClient.print(temperature);
    mqttClient.endMessage();

    mqttClient.beginMessage(topic2);
    mqttClient.print(humidity);
    mqttClient.endMessage();

      mqttClient.subscribe(topic3);
  // set the message receive callback
   mqttClient.onMessage(onMqttMessage);


  }



void onMqttMessage(int messageSize);
  // we received a message, print out the topic and contents
 Serial.println("Received a message with topic '");
  Serial.print(mqttClient.messageTopic());


  while (mqttClient.available()) {
     temp = (char)mqttClient.read();
      Serial.print((char)mqttClient.read());
  }

  Serial.println();
  Serial.println();

}
```

Fig. 6 Code

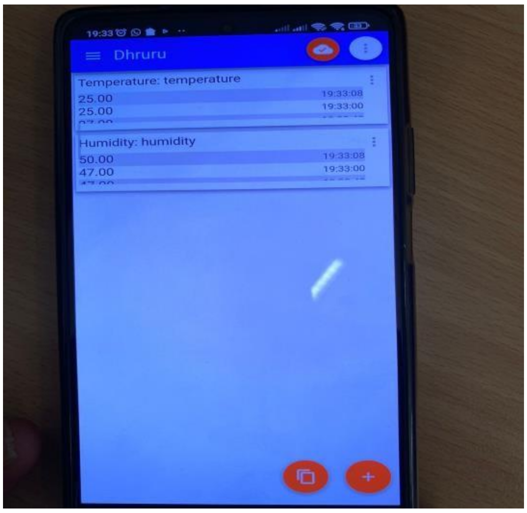
After uploading this code to to Arduino uno wifi we were able to establish it as mqtt client and the results were displayed on raspberry pi and the android phone. The attached pictures show our results.



```
Sending message to topic: temperature
25.00
Sending message to topic: humidity
46.00

Sending message to topic: temperature
26.00
Sending message to topic: humidity
50.00

Sending message to topic: temperature
25.00
Sending message to topic: humidity
49.00
```

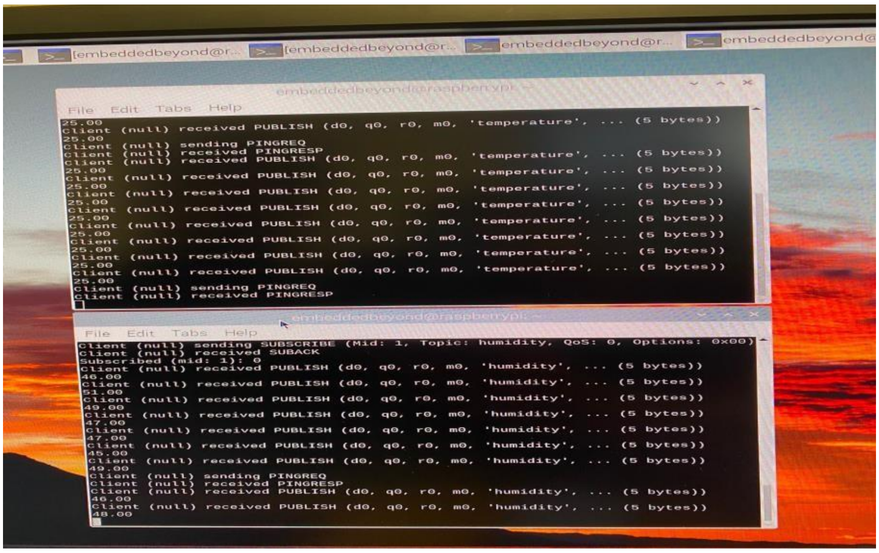
```
space. Max
mory, leav
 fuse "fu
 error in future version
uble check with your datasheet first).
```

Fig. 7 Arduino



```
Temperature: temperature
25.00                    19:33:08
25.00                    19:33:00

Humidity: humidity
50.00                    19:33:08
47.00                    19:33:00
```

Fig. 8 Mobile



```
25.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'temperature', ... (5 bytes))
25.00
Client (null) sending PINGREQ
Client (null) received PINGRESP
Client (null) received PUBLISH (d0, q0, r0, m0, 'temperature', ... (5 bytes))
25.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'temperature', ... (5 bytes))
25.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'temperature', ... (5 bytes))
25.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'temperature', ... (5 bytes))
25.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'temperature', ... (5 bytes))
25.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'temperature', ... (5 bytes))
25.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'temperature', ... (5 bytes))
25.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'temperature', ... (5 bytes))
25.00
Client (null) sending PINGREQ
Client (null) received PINGRESP
```

```
Client (null) sending SUBSCRIBE (Mid: 1, Topic: humidity, QoS: 0, Options: 0x00)
Client (null) received SUBACK
Subscribed (mid: 1): 0
Client (null) received PUBLISH (d0, q0, r0, m0, 'humidity', ... (5 bytes))
46.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'humidity', ... (5 bytes))
51.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'humidity', ... (5 bytes))
49.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'humidity', ... (5 bytes))
47.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'humidity', ... (5 bytes))
47.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'humidity', ... (5 bytes))
45.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'humidity', ... (5 bytes))
49.00
Client (null) sending PINGREQ
Client (null) received PINGRESP
Client (null) received PUBLISH (d0, q0, r0, m0, 'humidity', ... (5 bytes))
46.00
Client (null) received PUBLISH (d0, q0, r0, m0, 'humidity', ... (5 bytes))
48.00
```

Fig. 9 Raspberry Pi

## 6. Use Case

*The arduino reads the sensor and controls the Dc motor.Serial communication is establisxh between arduino and raspberry pi.Our raspberry pi acts as a mqtt which establish a connection to the smartphone via IOT smart panel App.The smartphone act as both publisher and subscriber using connection with raspberry pi.The following use case diagram describe the entire process.*

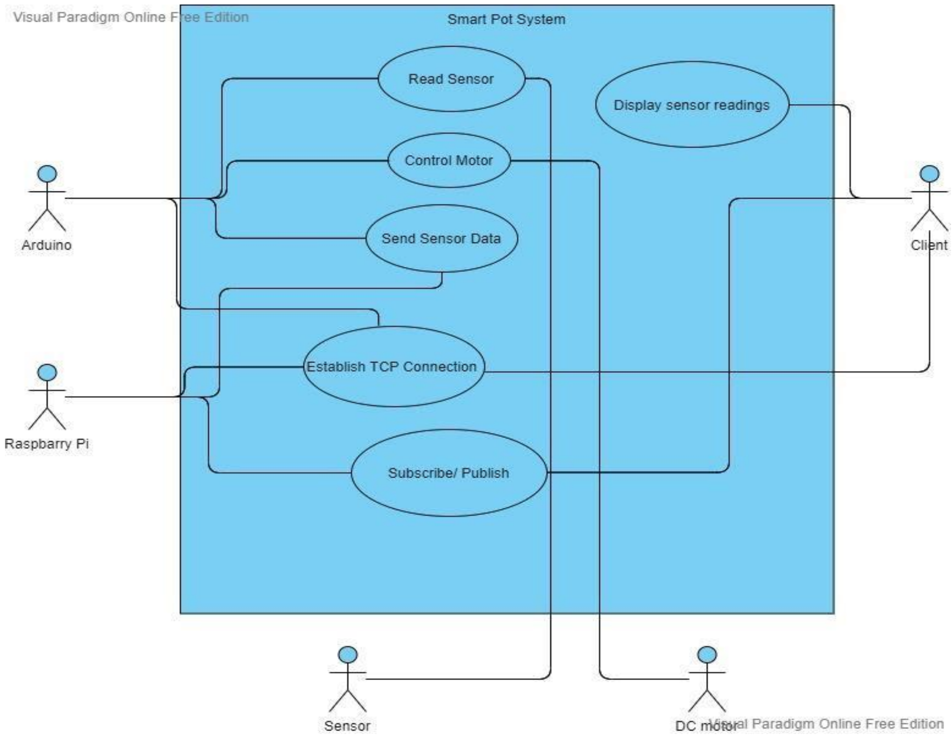
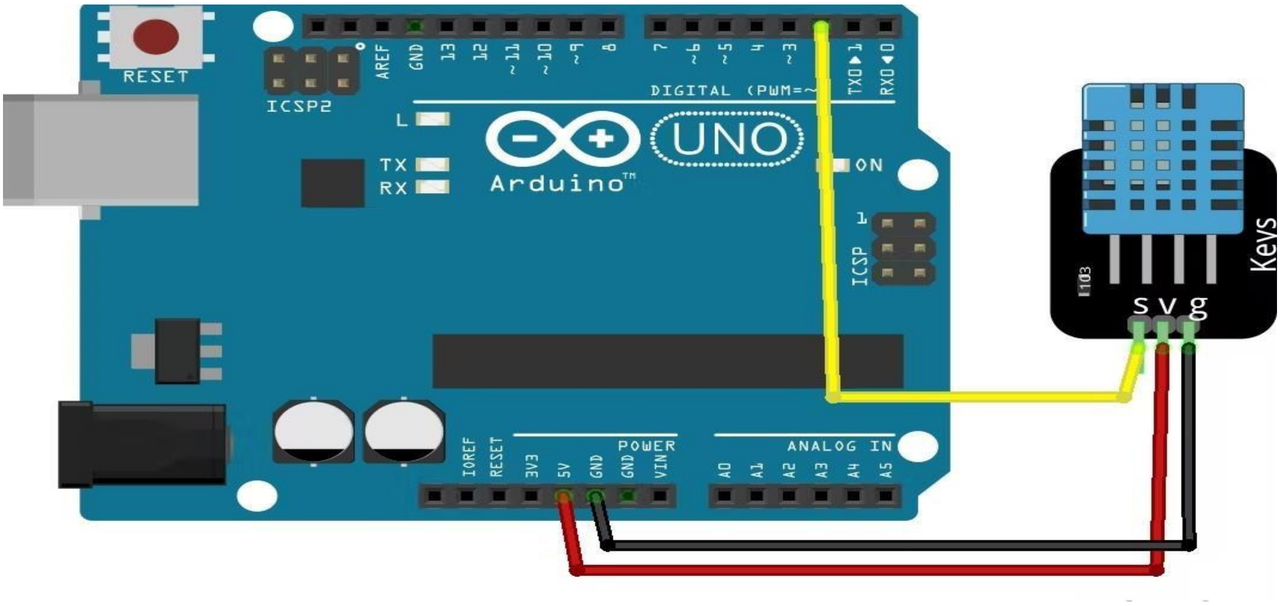


*Fig.7 Use Case*

*In order to use the application, we will need complete hardware. DHT sensor should be connected to the Arduino uno wifi as shown in the following figure.*



*The DHT sensor will give us the readings of temperature and humidity.*