Documentation
Team Exemplary
24.06.2022

1 Team members

- Asm Nurussafa
- Tasawar Siddiquy
- Arfat Kamal
- Nirojan Navaratnarajah

2 Introduction

For our project this semester, we would like to implement a small-home built smart plant-monitoring system for our household. The idea is to obtain real-time feedbacks for the condition of the plant, and act/control accordingly. For implementing this, we will be using IoT- WSN architecture. [1] A viable architecture in hierarchy can be as follows- the *sensing* layer, *control* layer, *backhaul* layer, *application* layer and *the decision* layer. The *sensing* layer comprises of the measurement sensors that interact with the environment to obtain data. These data are then computed and wirelessly transferred to the control nodes in the *control* layer. This control nodes would be responsible in transferring data to the back end of our system and control actions of our actuators. The *backhaul* basically bridges communication between these layers. For our project, we will using a synchronous Message Queuing Telemetry Transport (MQTT) communication protocol, which will be discussed more in detail in upcoming sections. The *application* layer defines the functionalities of our application and is responsible for seamless integration of data coming from these heterogeneous sensors. Finally, the *decision* layer performs analysis of these data from these devices and control the overall functional requirement domain of our system. These can be integrated easily wit IoT device, such as a smartphone in our case. The overall implementation is discussed further along the paper.

3 Concept description

I. **Main application of our prototype**:

As more and more IoT devices thrive in our daily lives, interests in automating most of the house-hold tasks have burgeoned. In addition, with the recent Covid-19 pandemic and imposed-lockdowns, most of the people have taken an interest in home gardening. Be it taking care of a favorite plant or to grow fruits in a small home-garden space, such activities have seen a spike in recent times. Hence, we only want to take this idea forward and make the process much easier by implementing a system that would provide real time information about us and allow us to make actions based on that, or better yet, perform tasks automatically. The main application of our project is to provide us temperature, humidity, and moisture readings of our plant to us. Based on these readings, we can choose to perform certain actions. As in our case, we will demonstrate a simple automated water pumping situation when the moisture falls below a certain level. With such system, people who are too busy to look after their plants, or people who are too old to keep track of plant's health can be enormously benefitted. Even when the user is away, they can still keep track of their plant's health. Our prototype is a simple, scalable version of what we really want to achieve, and this prototype can be used for further implementations.

II. Block diagram of our target application:

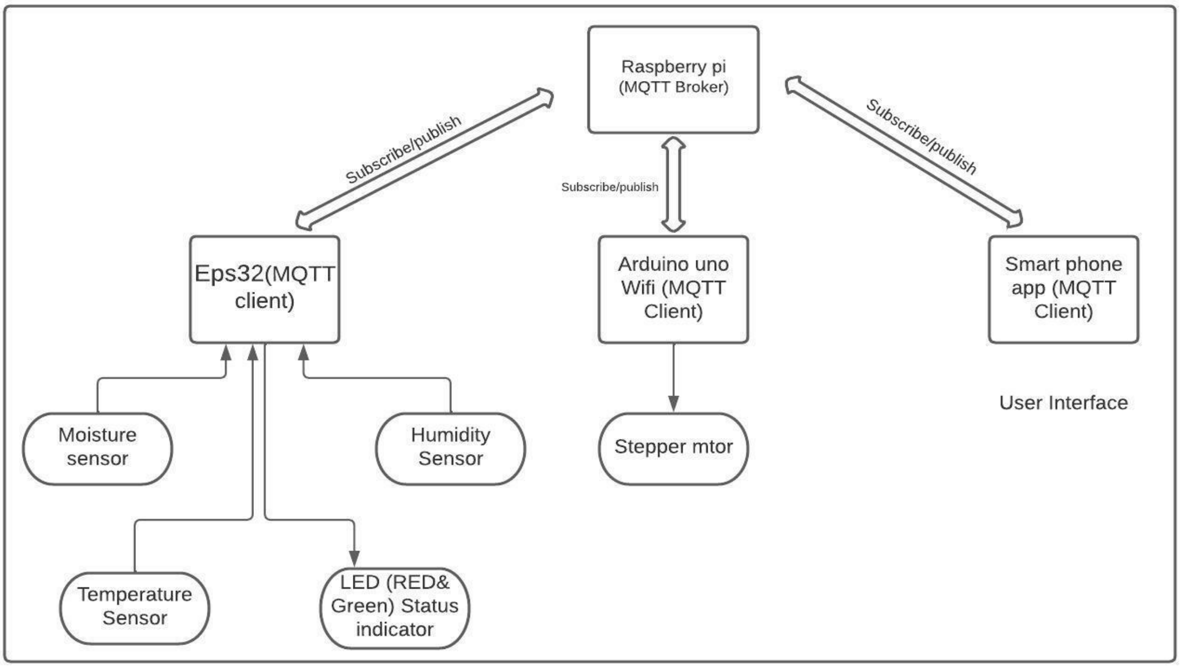

Fig. 1. Block diagram of our target application

Above, we can see the block diagram of our target application and how each of the sensors interact with each other in this scope.

For starters, we will be using the moisture sensor, temperature sensor and humidity sensor and use LED's (Light emitting diodes) to indicate the status of these sensors. These are then connected to the ESP32(MQTT client). Furthermore, we have a stepper motor actuator acting as a water pump, which is connected to an Arduino Uno Wifi (MQTT Client). These are connected to the Raspberry Pi (MQTT broker) which can be controlled using a Smart Phone App, **IoT MQTT Panel**, (MQTT client) to complete our scenario.

4 Project/Team management

To realize our project, we have decided to use a sort of the iterative development model. This will allow us to work on our project and test our scenario iteratively, meaning it lets us go back to previous steps and work on it and then, move on forward as we seem necessary. To organize the project in an orderly manner and to avoid miscommunication, we divided specific works for ourselves and held weekly meetings to comment on our progress and to decide on future tasks. When implementing, we got together in person as often as we could and worked on to realise our target scenario. A documentation of each group member`s progress is shown below. The dates stated indicate we had a meeting in each of these dates.

	#Week	07.05.2022	21.05.2022	04.06.2022	18.06.2022
	Task	Task 1	Task 2	Task 3	Task 4
		Project: Implement the Smart Plant Monitoring system.	Project: Implement the Smart Plant Monitoring system.	Project: Implement the Smart Plant Monitoring system.	Project: Implement the Smart Plant Monitoring system.
Asm Nurussafa	To-do	Research about the project topic, hardware, sensors and actuator selection.	- Research on connecting sensors to ESP32 and Arduino. - Research on connecting ESP32 and Arduino to Raspberry Pi over MQTT. - Overall project planning.	- Implementing automated motor control over moisture sensor reading with python.	- Refine and finalise the automated motor control over moisture sensor reading with python. - Work on realising the whole project. - Work on documentation of the project and presentation together. - UML Class Diagram, Sequence and Use-Case diagram.
	Status	Done	Done	Done	Done
Tasawar Siddiquy	Short summary	Research about the project topic, hardware and sensors and actuator selection.	- Research on connecting sensors to ESP32 and Arduino. - Research on connecting ESP32 and Arduino to Raspberry Pi over MQTT. - Overall project planning.	-Connecting Moisture Sensor, coding and creating MQTT topic to publish the values	- Work on documentation of the project and presentation together. - Work on realising the whole project.
	To-do	Done	Done	Done	Done
Arfat Kamal	Short summary	Research about the project topic, hardware and sensors and actuator selection.	- Research on connecting sensors to ESP32 and Arduino. - Research on connecting ESP32 and Arduino to Raspberry Pi over MQTT. - Overall project planning.	-Connecting Temparature & Humidity Sensor, coding and creating MQTT topic to publish the values	- Work on documentation of the project and presentation together. - Work on realising the whole project.
	To do	Done	Done	Done	Done
Nirojan Navaratnarajah	Short summary	Research about the project topic, hardware and sensors and actuator selection.	- Research on connecting sensors to ESP32 and Arduino. - Research on connecting ESP32 and Arduino to Raspberry Pi over MQTT. - Overall project planning.	-Selecting the rigth motor (stepper motor),coding in arduino IDE and creating a topic to subscribe to and activate it. -LEDs and buzzer coding in arduino IDE	- Creation of Block diagram for Paper and Presenation. - Work on realising the whole project - Creating the User Interface in the MQTT IOT panel app together
	To-do	Done	Done	Done	Done

5 Technologies

Below we discuss all the technological aspects we have used throughout our project.

1. Sensor and actuator technologies:

i. Temperature and Humidity sensor (DHT11):

Monitoring the temperature and humidity of the environment around our plant is of utmost importance. The humidity sensor is a device that senses, measures, and reports the relative humidity (RH) of air or determines the amount of water vapor present in gas mixture (air) or pure gas. When relative humidity levels are too high or there is a lack of air circulation, a plant cannot make water evaporate (part of the transpiration process) or draw nutrients from the soil. When this occurs for a prolonged period, a plant eventually rots. Therefore, it is essential to monitor humidity and take measures to maintain the humidity level at the best level. To realize this, we have used a DHT11 sensor. The DHT11 is a basic, ultra-low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air and spits out a digital signal on the data pin (no analog input pins needed). It's fairly simple to use but requires careful timing to grab data. We can get new data from it once every 2 seconds, so when using the library from Adafruit, sensor readings can be up to 2 seconds old. The specification for this component can be found under – [2].

ii. Moisture sensor (ME110):

Soil moisture sensors measure or estimate the amount of water in the soil. Water is required to serve as a carrier in the distribution of mineral nutrients and plant food. Plant cells grow by increasing in volume and for the cells to increase in volume they must take up water. So, the moisture level in the soil must be monitored regularly and reported. To realise this, we use the ME110 moisture sensor for Arduino. This soil moisture sensor is a simple breakout for measuring the moisture in soil and similar materials. The two large, exposed pads function as probes for the sensor, together acting as a variable resistor. The more water that is in the soil means the better the conductivity between the pads will be and will result in a lower resistance, and a higher SIG out. Further specifications for this component can be found under – [3]

iii. Stepper motor (28BYJ-48):

We used a stepper motor, with the intention for it to work as a water pump. The idea is, when the moisture levels drops below a certain level, the stepper motor will be activated to pump water. To realise this we used a 28BYJ-48 stepper motor. Specifications of such a motor can be found here- [4]

2. Communication protocols: (MQTT)

In this project the MQTT (**Message Queue Telemetry Transport**) protocol will be used to transfer sensor data from the ESP32 client to the Raspberry Pi (broker) and the Arduino Uno Wifi on the other hand keeps on subscribing to a topic which controls the actuation of the stepper motor . MQTT is a bi-directional communication protocol where each client can both produce and consume data by publishing messages and subscribing to topics. The big advantage of this two-way communication is that the IoT devices can send sensor data and at the same time receive configuration information and control commands.

The reason for selecting this protocol compared to HTTP, MQTT is faster, has less overhead and less power consumption. The other difference to HTTP is that in MQTT, a client does not have to pull the information it needs, but if there is new data to be sent, the server (broker) pushes the information to the client.

More information about the specifications of ESP32, Raspberry Pi 3 and Arduino Uno Wifi can be found here- [5, 6, 7].

3. Programming languages:

i. The sensors are connected to the EPS32 board with wires and the C/C++ language is used for coding the ESP32 board in the Arduino IDE in order to get the data from the sensors. Another Aruidno uno Wifi Rev2 is also acting as a client subscribes to a certain topic in which it activates a stepper motor connected to it with wires.
ii. The data gathered from the sensors will then be sent from the ESP32 to the Raspberry pi via the MQTT protocol,and on the other hand ,an Ardunio client listens on a topic which activates the motor.Normally the initial concept was to press a button on the mobile application and activate the motor,later on we decided to add an additional feature where a code will be running on the raspberry pi,which also subscribes to the topics, gathers the data and then decides itself if the motor shall be turned on,this process is basically an initial step of automating the steps which require manual interference. To implement this function the programming language that is used in the Raspberry pi is ‘Python’.

6 Implementation

1. Static structure of our project’s environment:

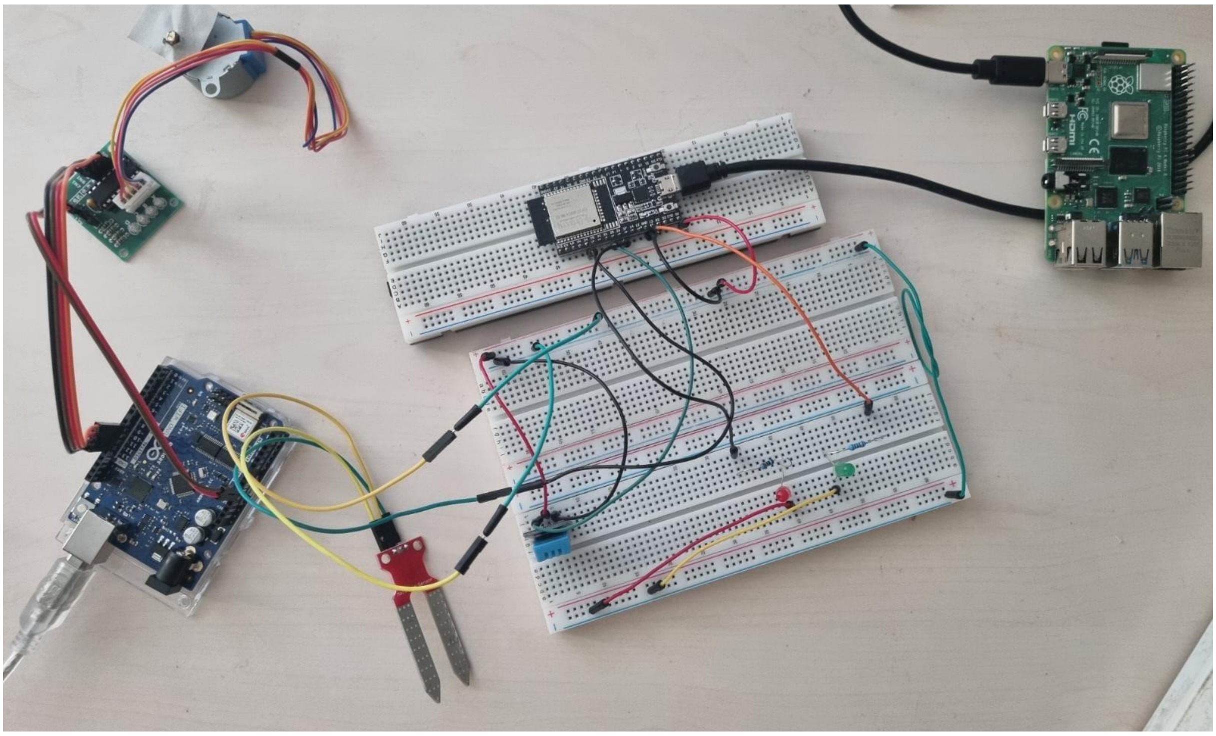

Fig. 2. Static Structure of our environment.

Fig. 2 shows the static structure of our setup environment and each of the components are connected to each other.

2. Class diagram:

In fig. 3 the class diagram for our setup environment. This is done using a hierarchical architecture model. At the lowest level are the sensors, namely- temperature sensor, humidity sensor and the moisture sensor. LED's are also connected in order to show the status of our connection. These are then connected to an ESP32 module (Client). Furthermore, we have a stepper motor module which is connected to an Arduino Uno Wifi (Client). The ESP32 and Arduino are then connected to the Raspberry Pi (broker), which is interconnected with a smartphone application.

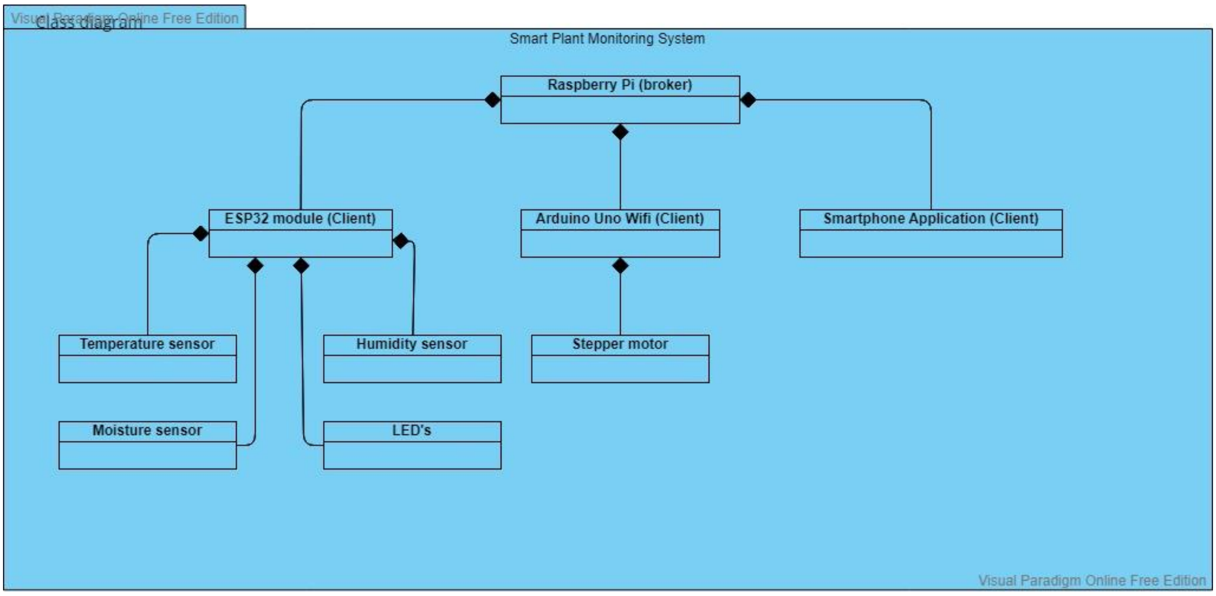

Fig. 3. Class diagram for our environment.

3. Use-cases for our environment:

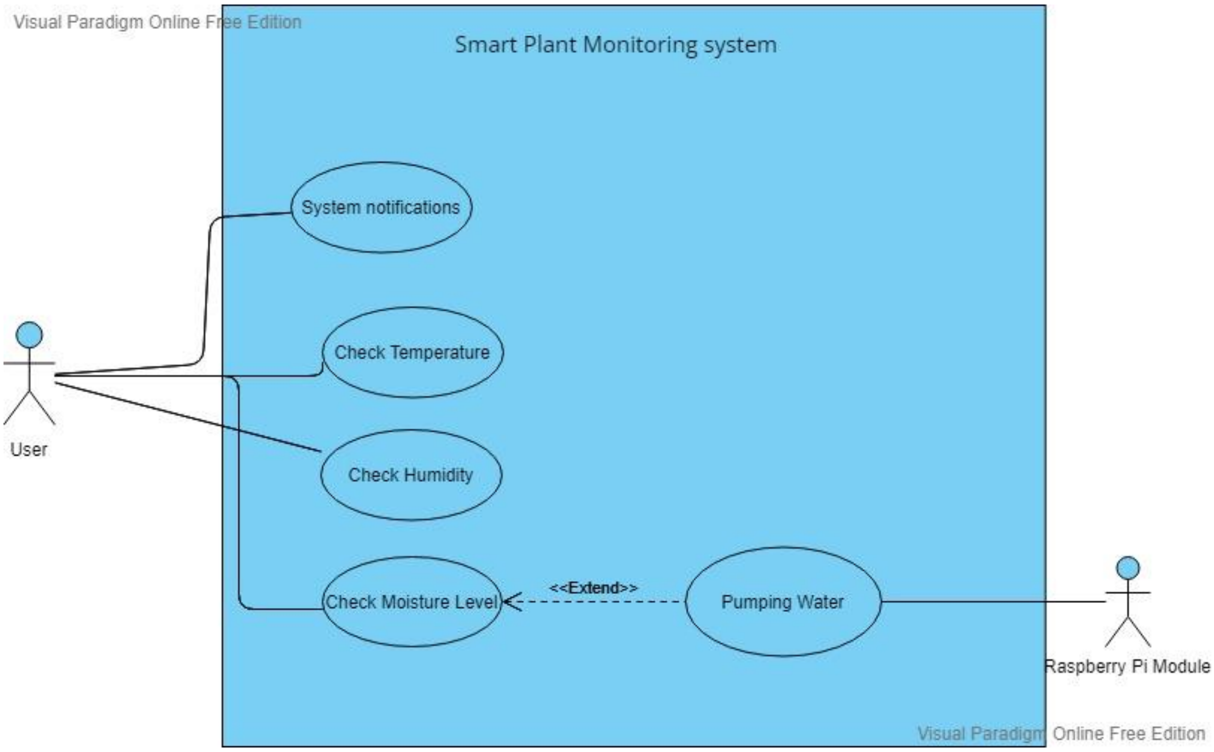

Fig. 4. Use Case for our environment.

To describe how our use-cases work and interact within the modules, we have included a sequence diagram in Fig. 5, which should be self-explanatory.

4. Results:

In Fig. 6, we can observe the results produced by our environment. A link to a live demonstration of this is attached in the reference/GitHub.

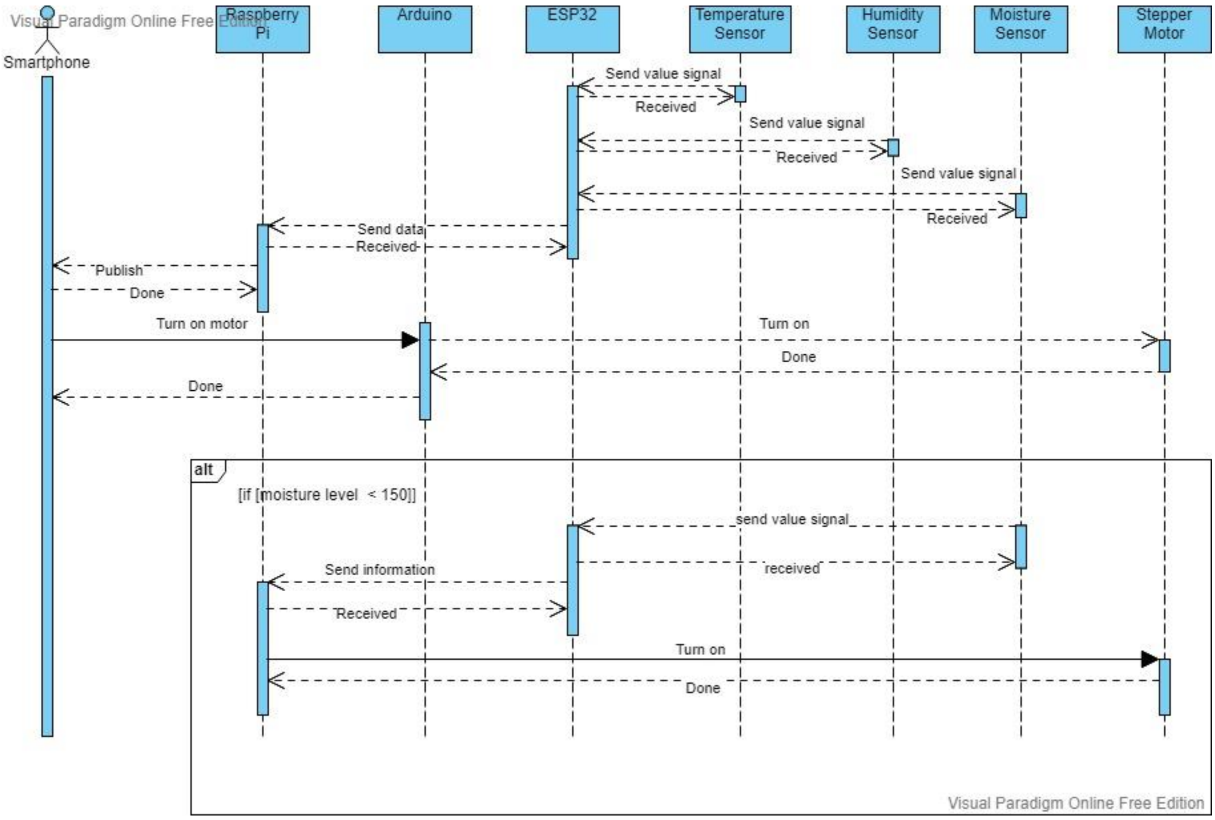

Fig. 5. Sequence diagram for our use-cases.

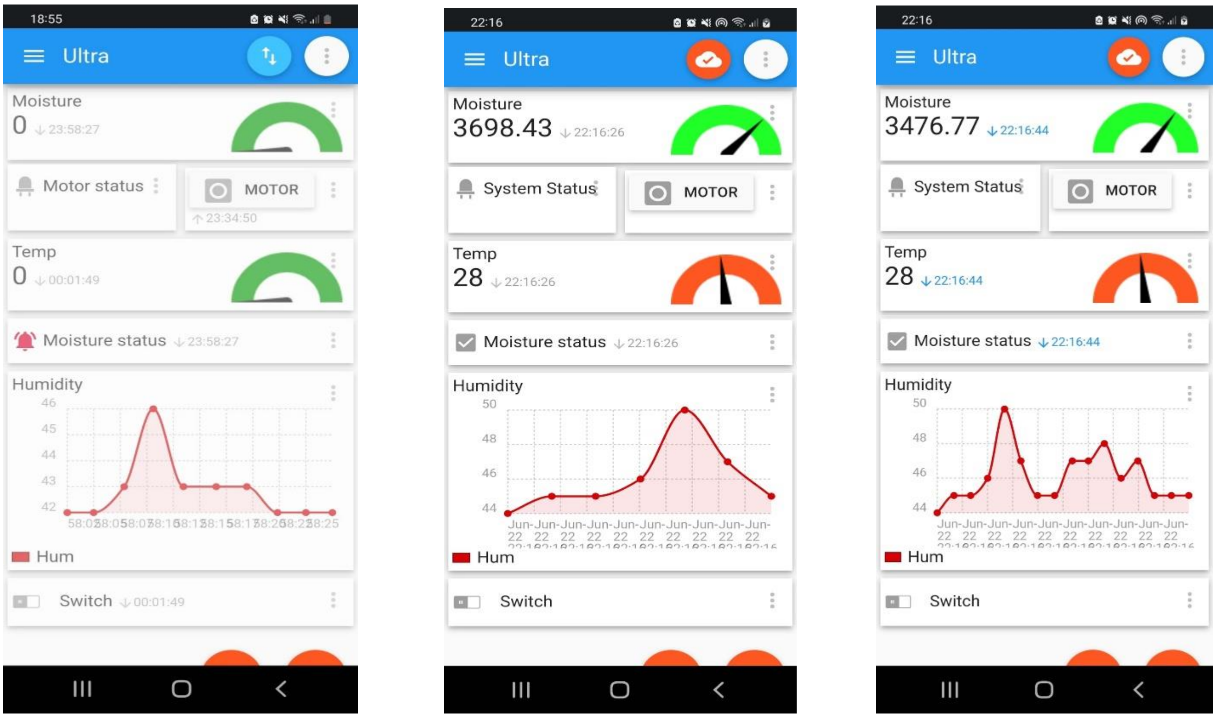

Fig. 6. Results.

7 Use Case

Give instructions on how to use your application. Potentially using an/more example(s), figures, screenshots etc.

Team Exemplary

8 Sources/References

Link to our github repository- https://github.com/Asm-Nurussafa/Advanced-Embedded-System--Team-Exemplary

Live demonstration- https://www.youtube.com/watch?v=jHqQJxqiC4c&ab_channel=UnitedIndiaExporters

[1] Bayih, A. Z., Morales, J., Assabie, Y., & de By, R. A. (2022). Utilization of Internet of Things and Wireless Sensor Networks for Sustainable Smallholder Agriculture. Sensors, 22(9), 3273.

[2] Anonim, T. humidity module DHT11 Product Manual.

[3] User Manual For Soil Moisture Sensor for Arduino(ME110). Retrieved from https://funduino.de/wp-content/uploads/2019/04/001616242-an-01-en-BODENFEUCHTE_SENS___MOD___IDUINO_ME110-1.pdf

[4] 28BYJ-48 datasheet. DigiKey. (n.d.). Retrieved June 22, 2022, from https://www.digikey.de/en/datasheets/mikroelektronika/mikroelektronika-step-motor-5v-28byj48-datasheet

[5] Datasheet, E. S. (2021). v. 3.7.

[6] Team, T. A. (n.d.). Uno WIFI REV2: Arduino documentation. Arduino Documentation | Arduino Documentation. Retrieved June 22, 2022, from https://docs.arduino.cc/hardware/uno-wifi-rev2

[7] Raspberry Pi 3 Model B+. Retrieved from https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-Model-Bplus-Product-Brief.pdf