Documentation

Team Bonsai

13.05.2022

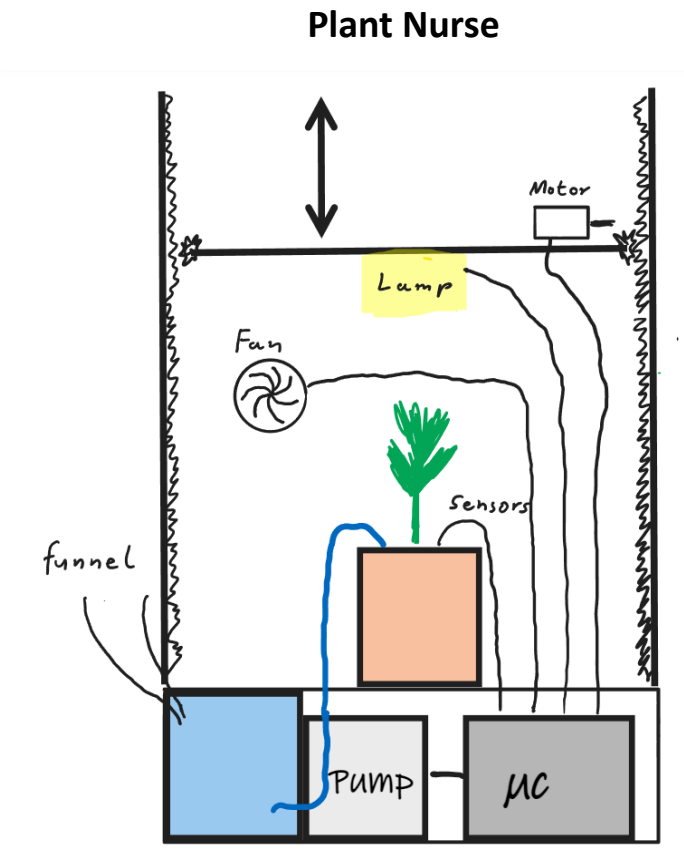
Team members

Nikolaos Karapoulatidis, Shehroz Bashir Malik, Christian Stratmann

Introduction

With our IoT device, growing plants in an optimal way shall be simplified and improved. The user should be able to control multiple “growing stations” via a single device wherever he is located at the time.
Each “growing station” is connected wirelessly to one external server where also external temperature sensors are connected to.

Concept description

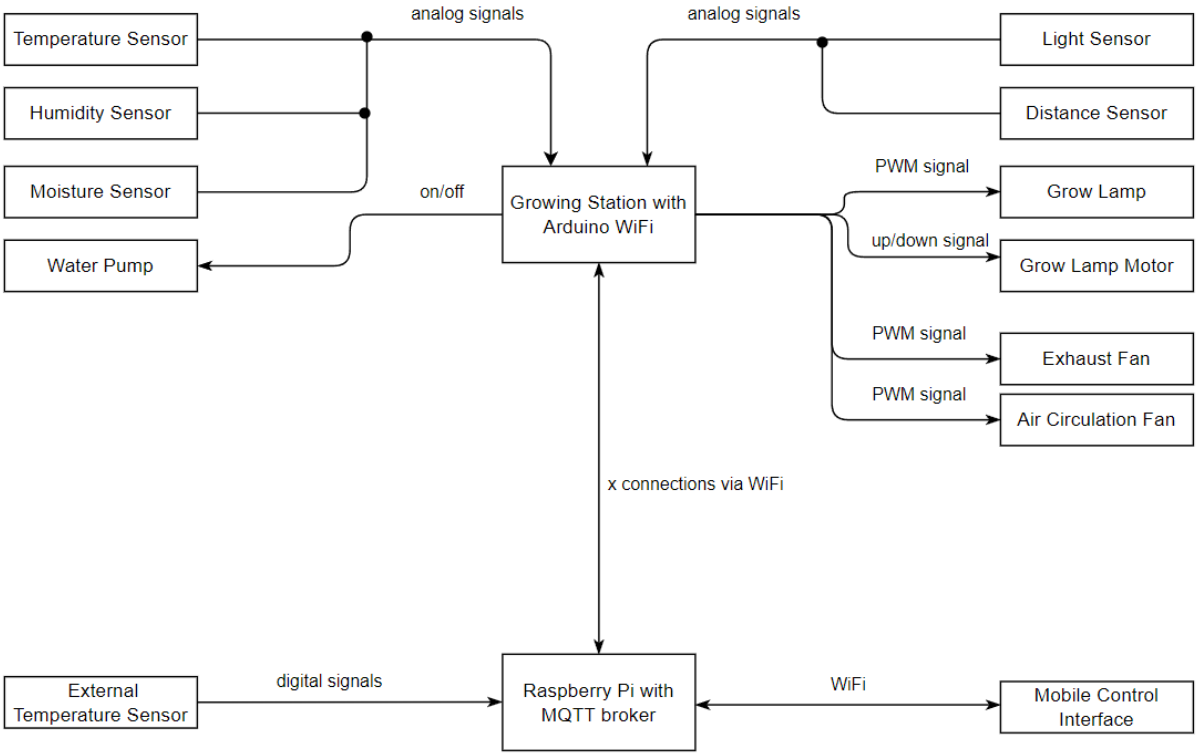


Picture 1: Concept drawing

Growing plants is mostly easy, but it can be a hard time to master.
Every plant has different needs for light, nutrition and water. As a medium interested user, keeping track of all those variables can be difficult. Specifically with multiple plants in multiple “growing stations”.

Therefore, every station is equipped with temperature, humidity, moisture, light and distance sensors.
Depending on the temperature and humidity sensor, exhaust and air circulation fans are controlled. The moisture sensor will trigger water pumps for watering the plants and the light sensor will control the growing lamps, depending on already receiving light from outside and the distance between lamp and plant.
At some points in the growing stage of a plant, the growing lamp has to cover a larger surface area or needs to go higher to prevent the lamp from burning the plant and requires the

motor, that can control the height of the lamp, to increase the distance between lamp and plant and eventually brightening the light to archive equal light intensity as before.



Picture 2: Block diagram of our project.

Project/Team management

We will be following the Agile Methodology. We will have weekly Agile Sprints. In the beginning of each sprint we will have sections dedicated to planning, design, coding and analysis. Initially, the team gathers together, physically or online, to have a brainstorming session. We discuss ideas to solve the problem, note down the requirements of the task ahead, adjust and incorporate our ideas to solve the aforementioned problem. An analysis is carried out to determine if our work is in line with the requirements. This is an iterative process and it is quite normal for us to restart our work if we have no productive outcomes. The tasks are split evenly amongst 3 members.

Shehroz:

- Raspberry Pi set-up and configuration
- MQTT on raspberry
- Documentation

Team Bonsai

Nikolaos:

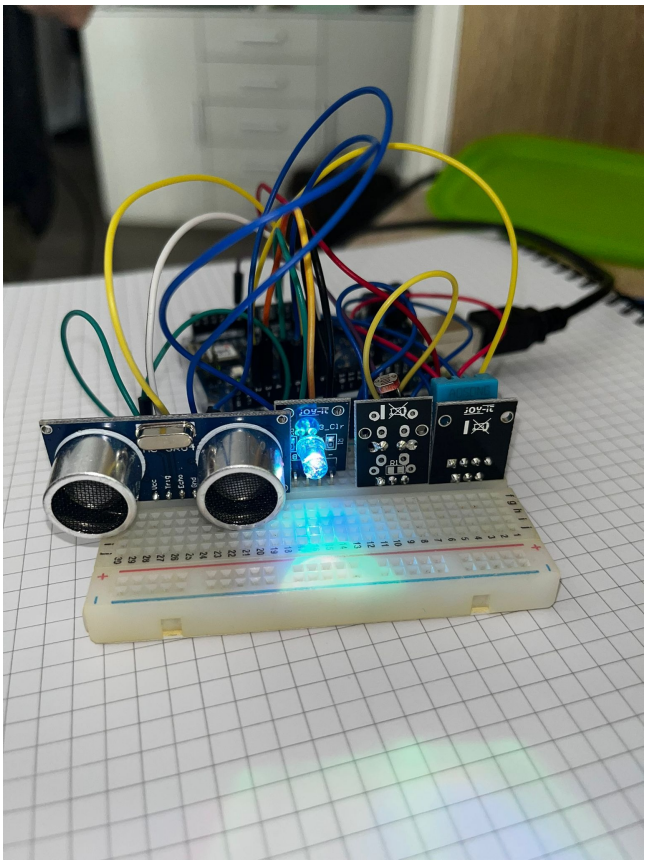
- Use-Case diagram
- Programming Arduino
- MQTT panel setup
- Documentation

Christian:

- Programming Arduino
- Class diagram
- Documentation
- Block diagram

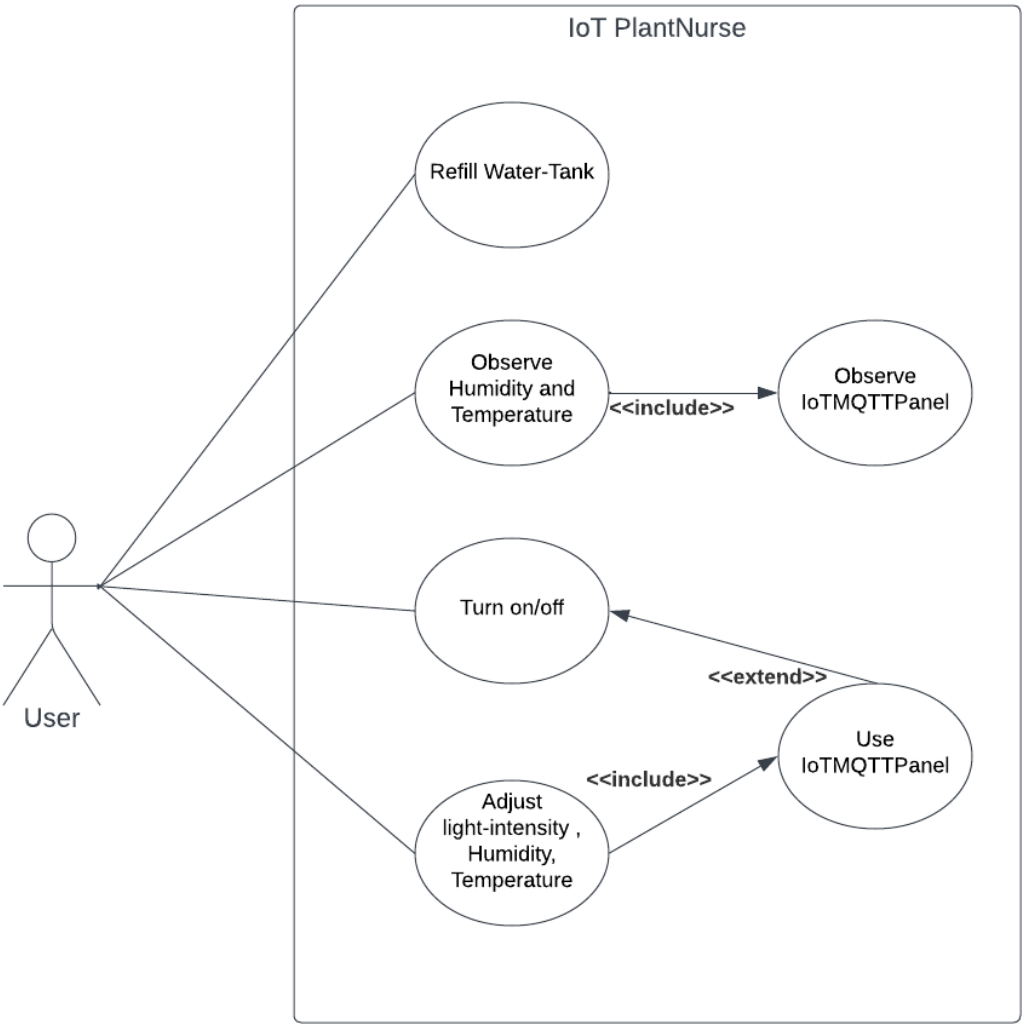
Technologies

We use together with the Arduino WiFi and a Raspberry Pi the X-40 Sensor kit.
For communication we use WiFi, as it is easy to implement, broadly used and it is the technology that is supported by the Arduino.
Furthermore we will use µC for programming the Arduino and C or maybe Python on the Raspberry Pi side.



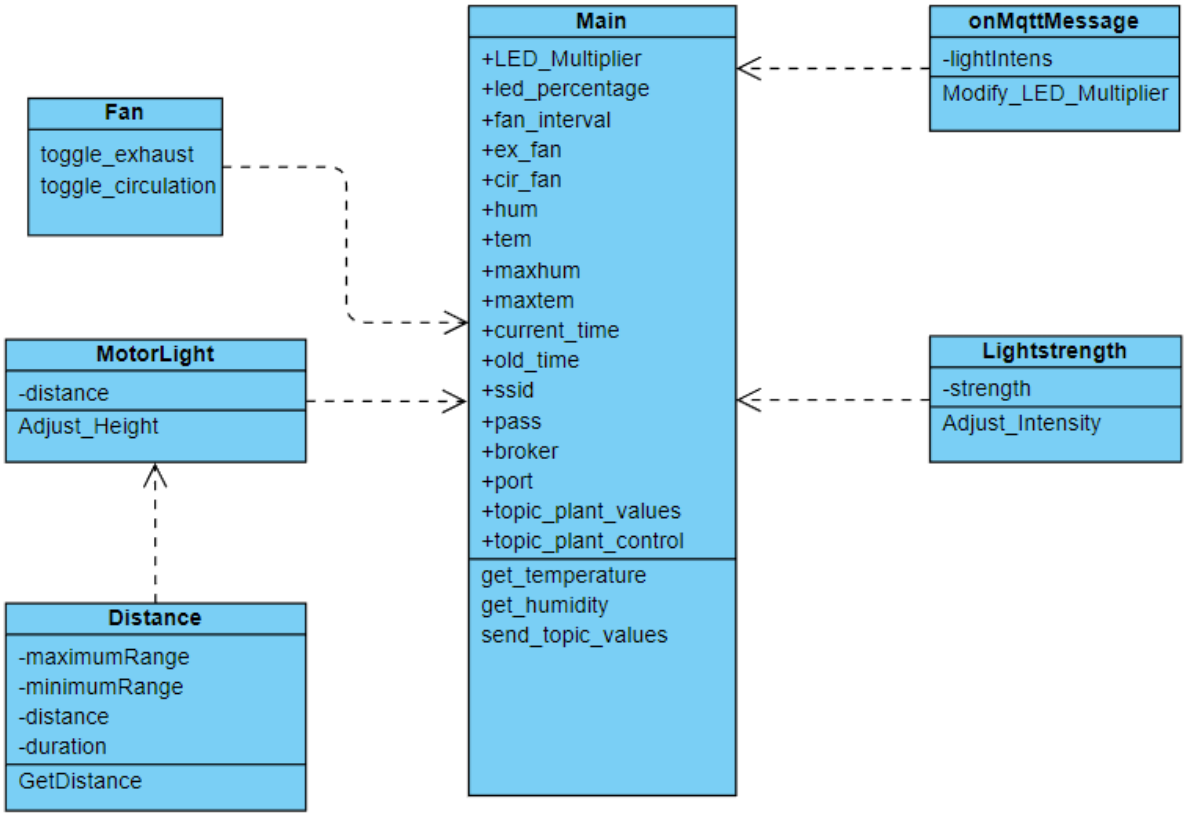
Implementation

Use Case



The Plant-Nurse notifies the user on the mobile control interface when the water tank is about to get empty. The user can refill the water-tank with a normal watering can. The current values of the temperature and humidity are shown on the user's smartphone by using the IoTMQTTPanel app . Depending on the plant type on which the Plant-Nurse should take care of, the required light intensity can change. For this reason the user has the possibility to adjust the light-intensity which will in turn change the PWM-signal controlling the LED-lamp. Depending on the plant's height a small motor elevates the lamp to protect the plant from getting burnt. Furthermore the user can enter the optimal humidity for the plant which is then ensured by the amount of water transported by the pump. Last but not least the user can change the maximum temperature and humidity to control the exhaust fan and air circulation fan. For example if the user enters 30 C° for the maximum temperature, the air circulation fan will be turned on when the temperature sensor detects an environmental temperature more than 30 C°.

Class Diagram



As we only used methods and no classes, we only display the modules of our program.
First, the main function is holding all the needed global variables, which may be modified by one of the modules at a later point. It is also reading the temperature and humidity sensor, as it only requires two lines of code. At the end of the main function, the status of the system is sent to the “topic_plant_values” topic via MQTT.
The “onMqttMessage” method is waiting for an incoming MQTT message and modifies the light intensity. This is done by a multiplier that is influencing the PWM value “strength” of the “Lightstrength” method.
The “Fan” method is toggling the fans on and off with the help of temperature and humidity readings. When the maximum temperature, or the maximum humidity is reached, the exhaust fan is turned on to get rid of the too moist, or too hot air.
The air circulation fan is turned on, after some interval.
The “MotorLight” method is controlling the distance between the lamp and plant. For doing that, it needs to call the “Distance” method that calculates the distance with the help of an ultrasonic sensor. It returns the distance value to the “MotorLight” method afterwards. With the distance in hand, it aims to keep the light source between 30 and 50 centimeters away from the plant, by turning the motor right, or left.

Team Bonsai

Code

In this section we are talking about how we implemented the modules from the class diagram into code.

Setup

In the setup we are connecting to the serial, Wi-Fi and MQTT. For the Wi-Fi and MQTT connection, we are waiting until the connection is established.
Afterwards, the output and input pins are declared, which were required by the ultrasonic sensor, RGB LED, light sensor and DHT11 sensor.
At the end of the setup, the different MQTT topics are subscribed and a method is initialized that is called, when a MQTT message arrives.

Loop

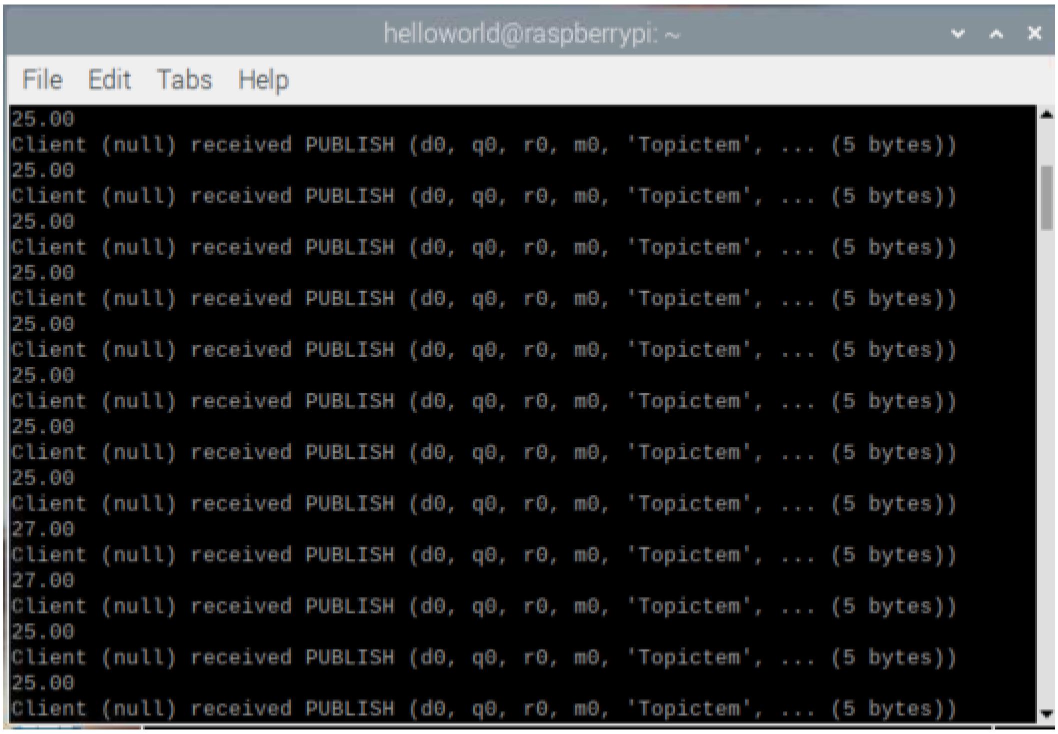
Every second, the DHT11 sensor values are read and published on the serial monitor and responsive MQTT topics.
Also, the distance between the LED and plant is adjusted with the help of the ultrasonic sensor. As the ultrasonic sensor only provides us with a signal when the soundwave returns, we had to do some time measurements and calculations.
The third thing that is done here is adjusting the light intensity of the LED by the sensor reading of the light sensor. Additionally, the strength is influenced by a multiplier variable ranging from 0 to 1. This variable can be changed via the MQTT “lightIntensity” topic.
At this point we faced a problem that resulted in weird numbers which didn’t make sense. We expected incoming integers ranging from 0 to 100, but when used in a calculation, we got values going over 600.
After some time we figured out that the program was converting the chars incorrectly. Instead of returning the corresponding value, it gave us back the index from the ASCII table. This was solved by simply subtracting a zero as a char from the actual char with the value of interest.
Lastly the fans are controlled. The exhaust fan is turned on whenever the humidity or temperature is above the max values and the circulation fan is turned on and off via an interval.

MQTT

MQTT was used to provide an interface between the Raspberry Pi and the Arduino setup. The first step was to install the MQTT Broker software on the Raspberry Pi. There were a few challenges with MQTT. We observed the system worked fluidly without authorization.
Multiple topics were created according to our needs.

Temperature

Topictem was used to record the temperature values:

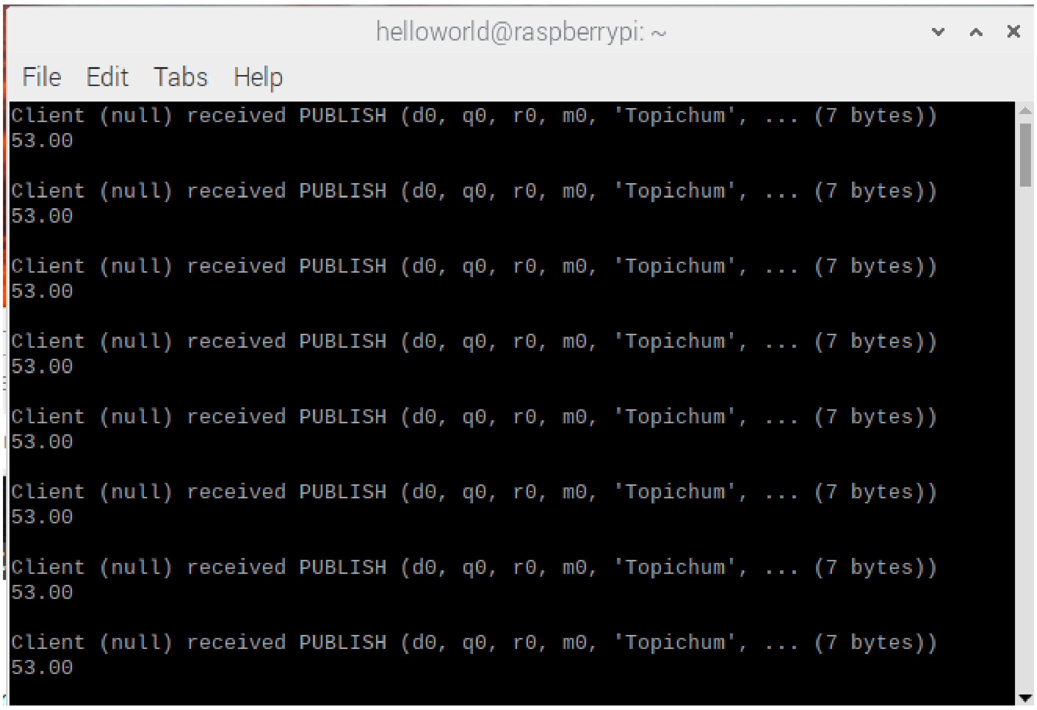


This value was verified with an external temperature sensor

There were slight fluctuations between the recorded value and the actual value. We believe this to be due to the difference in sensitivities of the Arduino sensor and the external sensor. Keep in mind, these are not lab-grade equipment so slight deviations are expected.

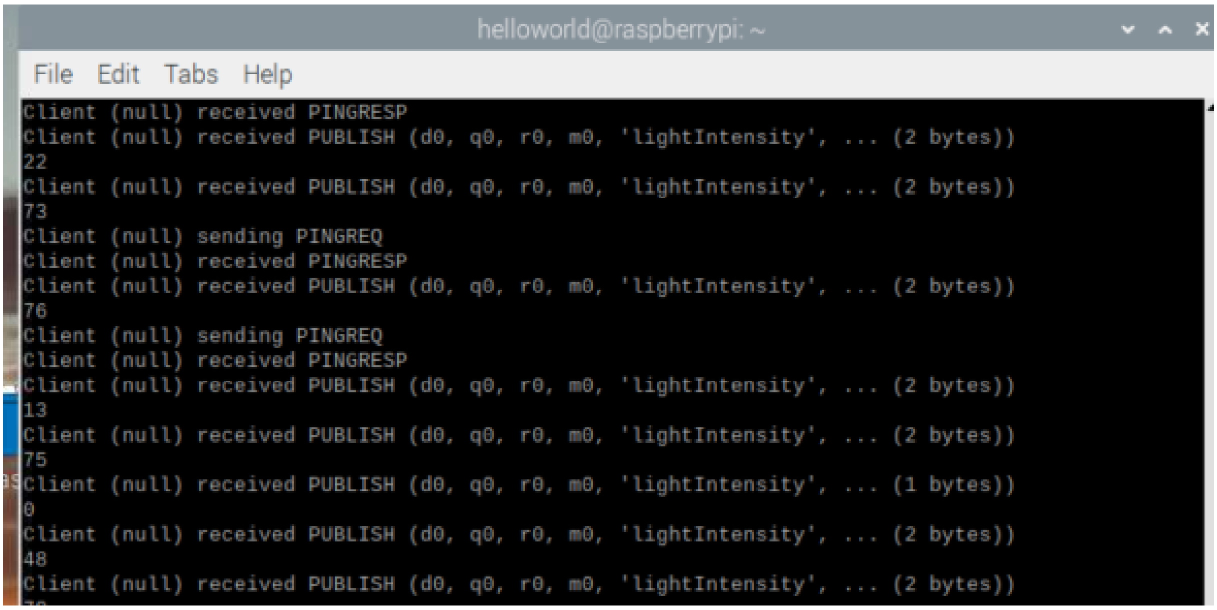
Humidity

The next value to be recorded is the Humidity Value. A topic was created for this and named Topichum. It reported values between 50 and 53 percent. The external sensor showed 44%. This is a rather large deviation as compared to the temperature.



Light Sensitivity

Light Sensitivity values were recorded as expected. The fluctuations are due to manual interference. We covered the sensor and the values changed accordingly. 13% is when covered and 76% in a normal ambient light setting.



Future Work

What we have implemented so far of our concept is the transmission of the humidity and temperature value to our MQTT panel, as well as the adjustment of the light intensity, max. Temperature and max. Humidity by using the MQTT panel app. We use serialPrint to simulate the activation time of the fans and the LED elevation motor.
To realize the “real-life” Plant-Nurse we would need to add a motor, powered by an external power source. This power source could be switched by a relay which is controlled by our arduino WiFi. The same principle can be used for the exhaust fan. Since we have already implemented the functionality of the motor and exhaust fan, the programming effort would be very low. We just would need to map it to 2 digital outputs.
What is not implemented yet in the code is the control of the water pump. However, it would work basically like the code for the exhaust fan or elevation motor (turning the water pump based on the desired moisture level).
Last but not least we would need a chassis which is holding every component in place and a rack with a suitable gear attached on the elevation motor.

Conclusion

In general, we have demonstrated an Arduino communicating with a Raspberry Pi using MQTT. We have also achieved our goal of creating a plant monitoring system that has the capability of adjusting variables that affect the environment for the plant. There is a video in our Github Repository elaborating our results.

Sources/References

Provide the sources on the technologies and algorithms you used in your project (Github).

https://docs.arduino.cc/tutorials/uno-wifi-rev2/uno-wifi-r2-mqtt-device-to-device
https://randomnerdtutorials.com/testing-mosquitto-broker-and-client-on-raspbbery-pi/
https://randomnerdtutorials.com/testing-mosquitto-broker-and-client-on-raspbbery-pi/
https://play.google.com/store/apps/details?id=snr.lab.iotmqttpanel.prod&hl=de&gl=US