



Subject:

Application to robotics & IOT

Document title:

**Automated Greenhouse System (Mockup)**

Prepared by:

Name

M&M

Course

4ºA

15/02/2023

Date

Reviewed:

Name

Prepared by:

Date

File name:

**M&M.pdf**

Date:

**15/02/2023**

Edition:

**1**

Nº pages:

**10**



## Documentary control

<b>Authors</b>	Miguel Vilanova Fenollar Miguel Monge Rodríguez
<b>Project Name</b>	Automated Greenhouse System (mockup)

## Version control

Version	Changes	Description	Date
0.1	Template: Cover, Documentary Control	First version of the document	15/02/2023
0.2	Introduction	Added the Introduction, objectives, motivation details	18/02/2023
0.3	Planification	Added the Planification details	19/02/2023
0.4	Design	Added the Design phase details	28/02/2023
0.5	SW Design	Explanation	08/03/2023
0.6	3D model	Design all pieces and printed	15/03/2023

\* N/A = Not applicable

## Index

Introduction .....	4
Planification.....	5
Determine the requirements .....	6
URD.....	6
Risk planning .....	7
Hardware.....	8
Select sensors and actuators.....	8
Design physical model.....	9
Software .....	10
Design NodeMCU interface.....	10
Develop control algorithm .....	18
Develop the software.....	13
References.....	25

## Figures

1: Gantt Project, file attached. ....	5
2: NodeMCU e-diagram .....	11
3: Node Red interface example.....	12
4: MQTT Explorer (MQTT broker portable).....	12

## Tables

i: URD.....	6
ii: Risk Planning.....	7

## Introduction

We intend to create and implement an automated greenhouse system that optimizes plant growth, maximizes resource efficiency, and promotes sustainability. By integrating sensors, actuators, and control algorithms, we aim to create an intelligent system that can adjust the environmental conditions in real-time to meet the changing needs of the plants.

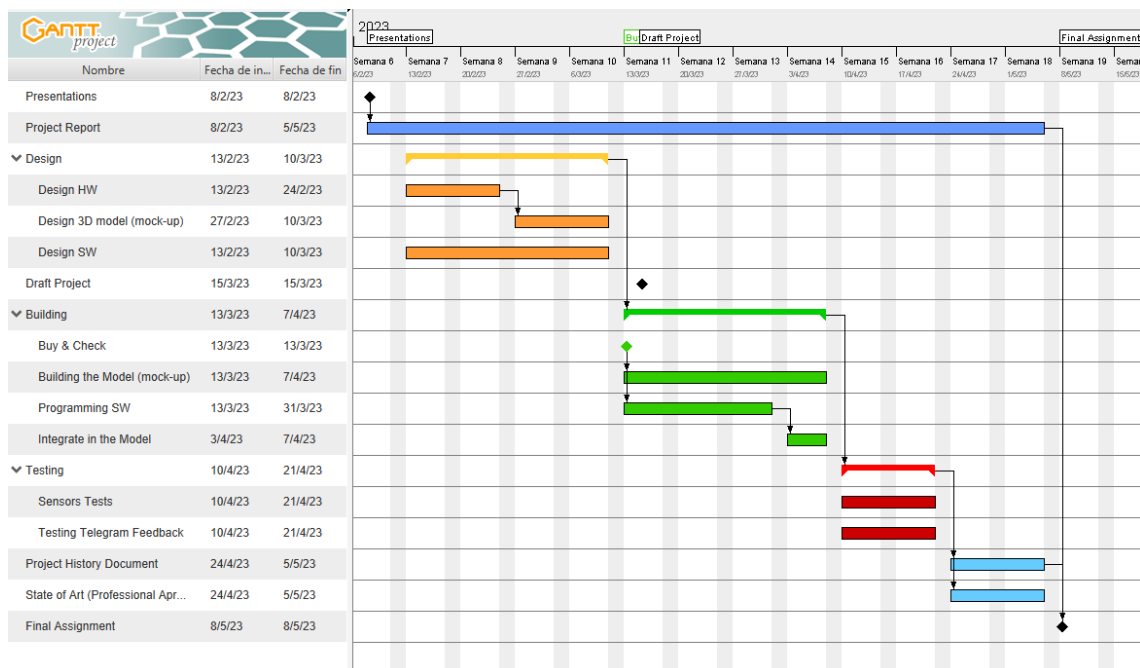
1. **Sensors collect data:** The sensors placed inside the greenhouse collect data on the environmental conditions such as temperature, humidity, light, soil moisture, and CO2 levels.
2. **NodeMCU collects and sends data:** The NodeMCU board, which is connected to the sensors, collects the data and sends it to the cloud-based platform, Node-RED, using Wi-Fi connectivity.
3. **Node-RED processes the data:** Node-RED is a visual programming tool that can process the data received from NodeMCU in real-time. It can analyze the data and trigger certain actions based on pre-defined conditions. For example, if the temperature in the greenhouse is too high, Node-RED can activate the cooling system.
4. **Data storage and analysis:** The processed data from Node-RED is then stored on a Raspberry Pi. The Raspberry Pi can be used to analyze and mine the data for valuable insights, such as identifying patterns in the growth of the plants, optimizing resource usage, and predicting future growth.

By using NodeMCU, Node-RED, and Raspberry Pi, you can create an automated greenhouse system that is both intelligent and efficient. The system can help you optimize the growth conditions for plants, reduce resource usage, and increase productivity.

Ultimately, our goal is to create a sustainable and efficient automated greenhouse system that can serve as a model for future agricultural practices. We believe that by combining technology, innovation, and ethics, we can create a better world for both plants and humans.

## Planification

- **Design:** This phase involves defining the objectives of the project and designing the hardware and software components. It starts on 13/2/23 and ends on 10/3/23.
- **Building and Testing:** This phase involves building the physical model of the automated greenhouse system, integrating the sensors and actuators, and programming the software. It starts on 13/3/23 and ends on 21/4/23.
- **Sensor Testing and Feedback:** This phase involves testing the sensors and collecting feedback through Telegram. It starts on 10/4/23 and ends on 21/4/23.
- **Project Documentation:** This phase involves documenting the project history and the state of the art of the automated greenhouse system. It starts on 24/4/23 and ends on 5/5/23.
- **Final Assignment:** This phase involves the final assignment submission. It takes place on 8/5/23.



1: Gantt Project, file attached.

## Determine the requirements

We can use humidity sensors to monitor the soil moisture levels of the plants. This can help to determine when to water the plants and avoid overwatering or underwatering.

Use the temperature sensor to monitor the temperature inside the greenhouse. This can be used to adjust the temperature based on the needs of the plants and optimize growth conditions.

Use the NodeMCU board to collect data from the sensors and send it to the cloud-based platform, Node-RED. We can process the data and trigger certain actions based on pre-defined conditions.

We can use a Raspberry Pi or a cloud-based database to store and analyze the data collected from the sensors. This can help to identify patterns and trends in plant growth and optimize resource usage.

Finally, we need humidity sensors to monitor the humidity levels inside the greenhouse. To adjust the humidity based on the needs of the plants and prevent plant diseases or pests.

### URD

ID	Requirement	Description	Priority
0	Temperature Sensor	Measures temperature inside the greenhouse	High
1	Humidity Sensor	Measures humidity inside the greenhouse	High
2	Soil Moisture Sensor	Measures soil moisture levels of the plants	High
3	CO2 Sensor	Measures CO2 levels inside the greenhouse	Low
4	Lighting Controls	Adjusts the intensity and duration of light for the plants	High
5	Irrigation System	Delivers water to the plants based on their needs	Medium
6	Ventilation System	Regulates air flow and temperature inside the greenhouse	Medium
7	NodeMCU + Wi-Fi Mod	Collects data from the sensors and sends it to the cloud-based platform	High
8	Raspberry Pi or Cloud	Stores and analyzes the data collected from the sensors	Medium
9	Node-RED Platform	Processes the data received from the sensors and triggers certain actions based on pre-defined conditions	High

i: URD

## Risk planning


It's important to anticipate potential difficulties and develop contingency plans to address them.

ID	Risk	Contingency Plan	Difficulty
0	Sensor malfunction or inaccurate readings	Regularly calibrate and test the sensor, have backup sensors available	Low
1	Sensor malfunction or inaccurate readings	Regularly calibrate and test the sensor, have backup sensors available	Low
2	Sensor malfunction or inaccurate readings	Regularly calibrate and test the sensor, have backup sensors available	Low
3	System failure or power outage	Have a backup power source available, manually adjust the lighting if necessary	Low
4	System failure or overwatering/underwatering	Backup irrigation components available, manually water the plants if necessary	High
5	System failure or inadequate air flow	Backup ventilation components available, monitor the temperature air flow manually	High
6	Wi-Fi or connectivity issues	Monitor connectivity, have backup communication methods available	Low
7	Data loss or corruption	Regularly back up the data, have backup storage solutions available	High
8	System failure or programming errors	Regularly test and debug the system, have backup programming solutions available	Low
9	Sensor malfunction or inaccurate readings	Regularly calibrate and test the sensor, have backup sensors available	Medium

ii: Risk Planning

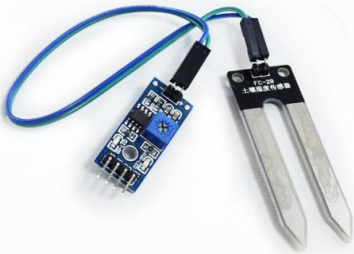

## Design

### Hardware

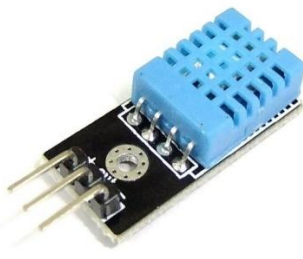

Motherboard	Description	Price	Photo
NodeMCU ESP8266 X1	We use this board to be able to control the sensors and actuators and communicate with the node-red through the mqtt protocol	4,00€/u	

#### Select sensors and actuators

Based on the plant requirements, we will select the appropriate sensors and actuators. For example, we might use temperature and humidity sensors, irrigation actuators, lighting controls.

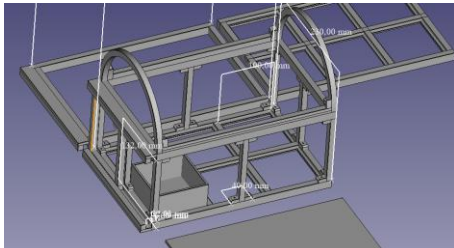
Sensor / Actuators	Description	Price	Photo
soil moisture X6	It measures soil moisture and is more accurate than others that we have used before	0,52€/u	
Servo motors X1	We use one servo motor to open de door and renew the air.	2,30€/u	



Temperature X1	we use this sensor to measure the temperature inside the greenhouse.	0,70€/u	
UV led strip X1	we use ultraviolet led to increase growth in low light hours.	3,68€/u	

#### Design physical model

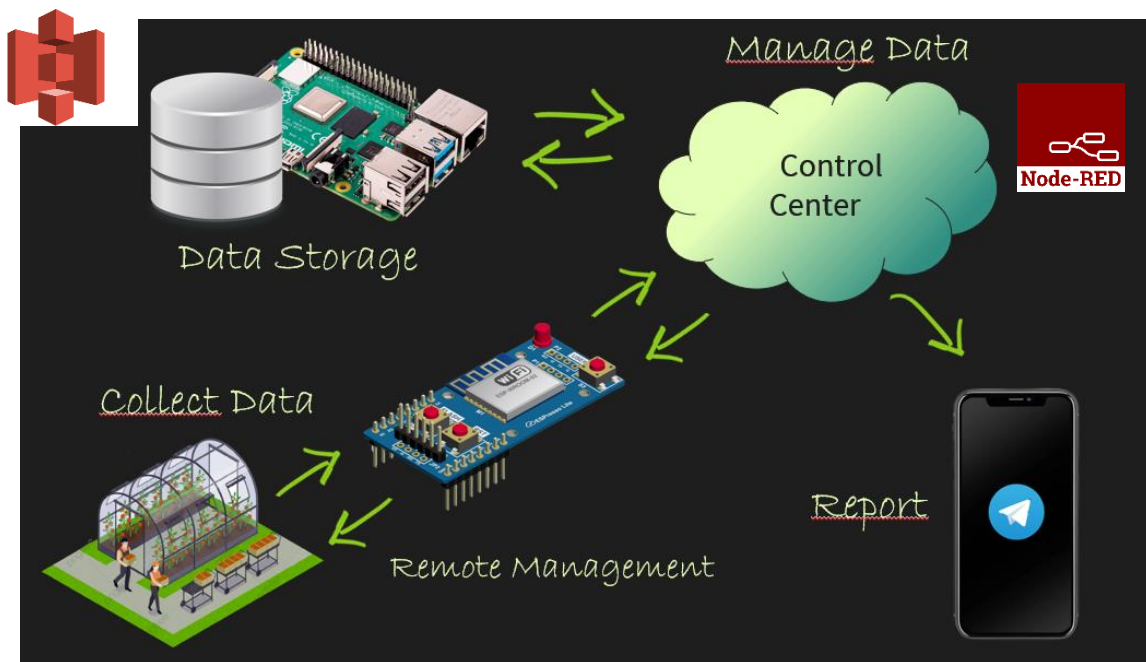
We will create a 3D model of the automated greenhouse system, including the layout of the sensors and actuators, and the positioning of the plants. This will help us visualize the system and ensure that it meets the objectives and plant requirements.

Piece	Time	Description	Price	Photo
3D filament X6	The totally time to print all the pieces was around 118 hours	we use 3d filament to print all the parts of the model.	27,50 €/u	

## Software

SW processes:

1. NodeMCU acts as the interface between the sensors and actuators.
2. Sensors collect data on the conditions inside the greenhouse and send it to the NodeMCU.
3. The NodeMCU then sends the data to Node-RED, which processes the information and can trigger certain actions based on predetermined conditions.
4. The processed data from Node-RED is then stored on a Raspberry Pi or cloud storage, where it can be analyzed and mined for valuable insights.



*By following these steps, we can ensure that the design phase of the automated greenhouse system is thorough and well-planned, and that it meets the objectives and plant requirements. This will set us up for success during the building and testing phase.*

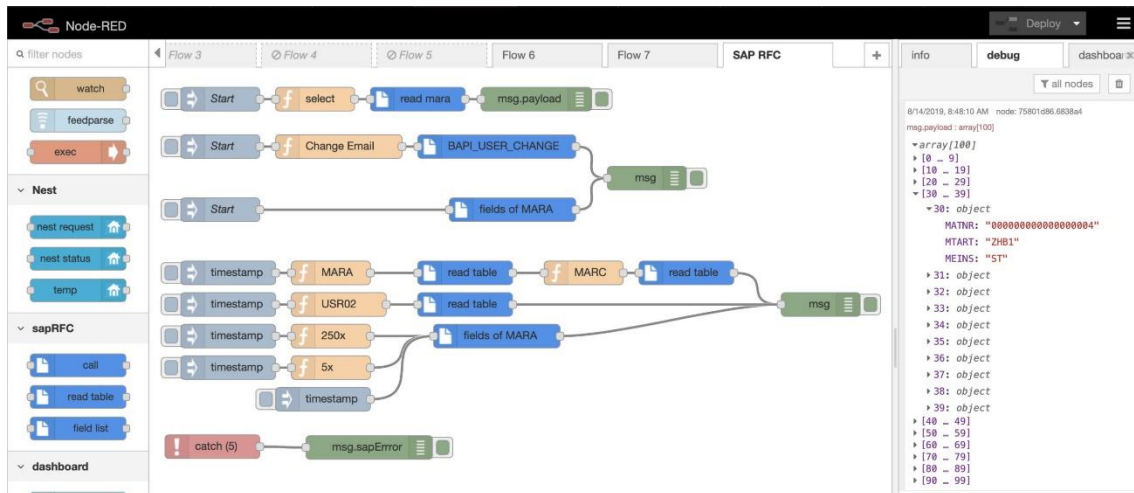
We will design the NodeMCU board interface that will collect data from the sensors and control the actuators. This might involve selecting the appropriate Wi-Fi module and programming the board to communicate with the cloud-based platform, Node-RED.

POWER  10-Bit Analog



It was developed by IBM and is now an open-source project. It is primarily used to integrate different IoT devices and services and can be used to process and visualize data streams in real time.

Node-RED uses a visual programming language called "flows" to allow users to wire together code blocks, called "nodes", to perform a task. It is often used in conjunction with hardware such as Raspberry Pi and other microcontrollers, as well as cloud-based services like AWS and Azure.



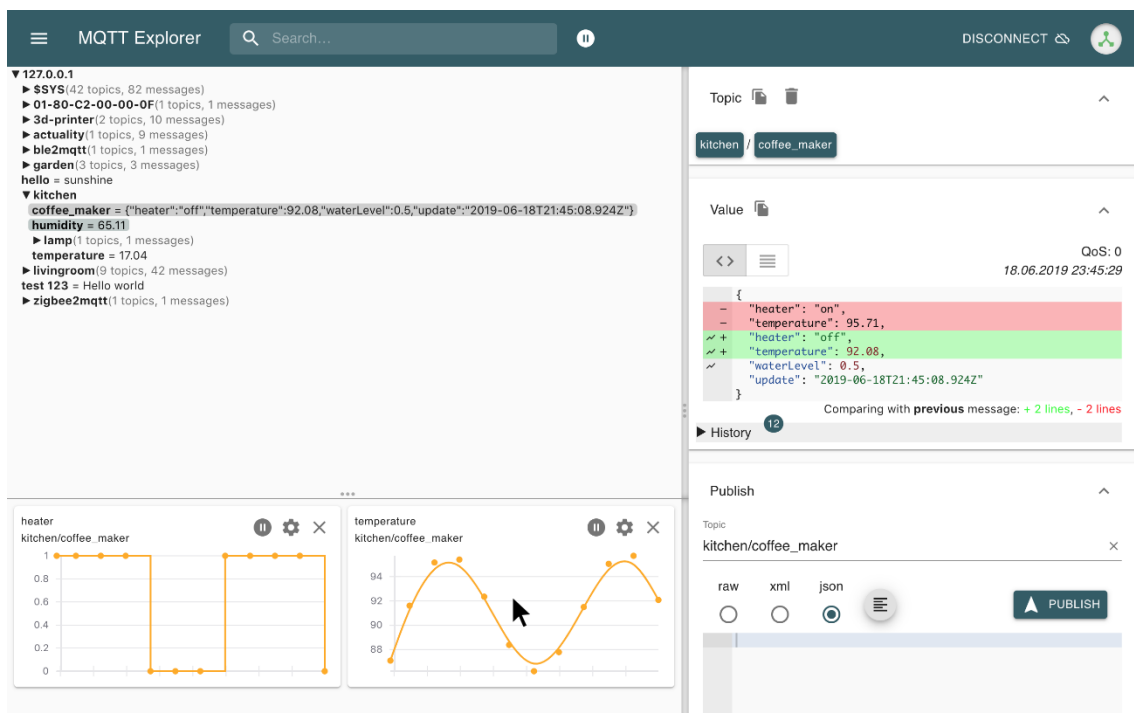
3: Node Red interface example

MQTT (Message Queuing Telemetry Transport) is a lightweight publish-subscribe messaging protocol that is used to send messages between devices. It was designed to be used with low-power devices and constrained networks, such as those found on the Internet of Things (IoT).

In MQTT, there are two main components:

- The message broker is responsible for receiving all messages and distributing them to the clients that are subscribed to the topic.
- The clients can be both publishers and subscribers, meaning they can both send and receive messages.

MQTT is a widely used protocol in the IoT and is supported by many devices and platforms, including Node-RED, which makes it easy to integrate with other systems.



4: MQTT Explorer (MQTT broker portable)

### Develop the software

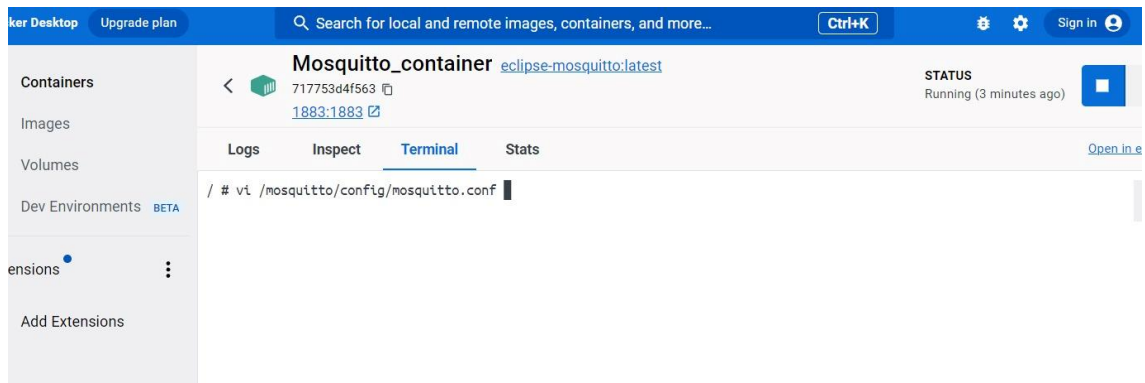
Finally, we will develop the software components, including the Node-RED platform and the data storage and analysis tools. This will allow us to process and analyze the data collected from the sensors and provide insights into the growth and efficiency of the plants.

### *Setup MQTT server (Docker)*

We need to download the latest [eclipse-mosquitto](#) image:

CMD

```
docker pull eclipse-mosquitto
```



We need to edit the [configuration](#) file to allow anonymous connections and listen on port 1883, also we can create a config file and run the docker with that configuration.

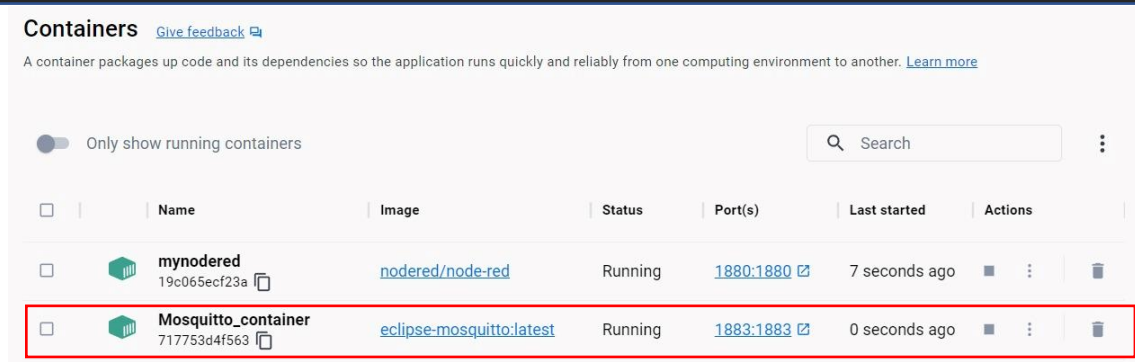
```
#
# listener port-number [ip address/host name/unix socket path]
listener 1883
```

```
listener 1883
```

```
allow_anonymous true
```

CMD

```
docker run -it --name mosquitto -p 1883:1883 -v mosquitto-config:/mosquitto/config eclipse-mosquitto
```



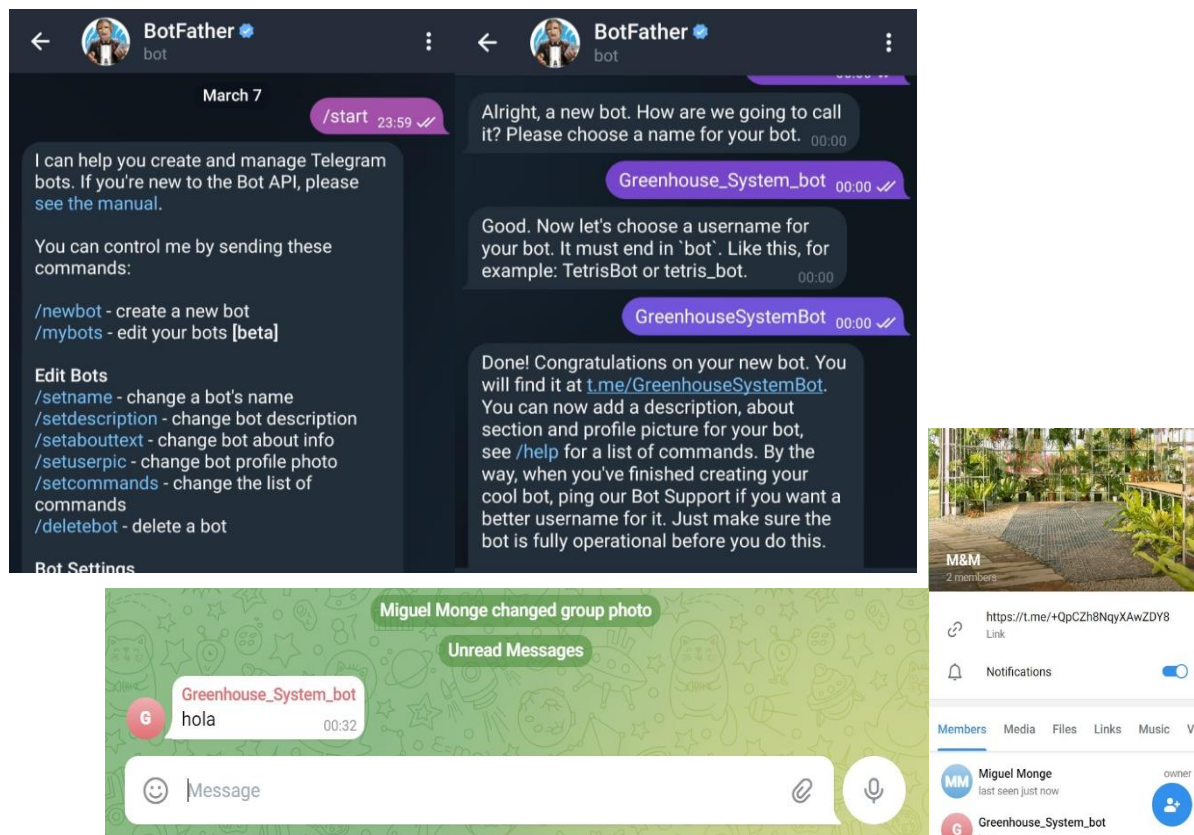
With that done we can connect to the server; we are going to use the portable version of mosquito explorer as said.



### Setup Telegram Bots

Crafting and configuring bots on Telegram is remarkably straightforward:

1. Search for BotFather in the search bar.
2. Start a chat with BotFather by clicking on the "Start" button.
3. Send the command `/newbot` to BotFather and follow the prompts to create a new bot, enter the bot's name: "GreenhouseSystemBot".
4. Once you have created your bot, BotFather will provide you with a token.
5. We then added the bot to a Group Chat named "M&M".



We can add more bots like this to serve additional purposes or functions.

## Setup AWS S3

We need to create a bucket, a bucket is a cloud-based storage container provided by Amazon Web Services (AWS) for storing and retrieving data.

Amazon S3 > Buckets > Crear bucket

### Crear bucket Info

Los buckets son contenedores de datos almacenados en S3. [Más información](#)

#### Configuración general

Nombre del bucket

El nombre del bucket debe ser único en todo el mundo y no debe contener espacios ni letras mayúsculas. [Consulte las reglas para la denominación de los buckets](#)

Región de AWS

Copiar la configuración del bucket existente: *opcional*  
Solo se copia la configuración del bucket en los siguientes ajustes.

You can enable versioning, encryption, object level logging, and other options to the bucket but we don't need it for this project.

	Nombre	Región de AWS	Acceso	Fecha de creación
<input type="radio"/>	mygreenhousebucket	UE (Londres) eu-west-2	Bucket y objetos que no son públicos	7 Mar 2023 11:32:13 PM CET

### Setup Node Red (Docker)

Since we need to create and delete files in a container, we create a data volume within the container. A data volume is a specially designated directory within one or more containers that bypasses the container's union file system and provides a way for persistent data to be stored and shared among containers.

CMD

```
docker run -it -p 1880:1880 -v ~/nodereddata:/data --name mynodered nodered/node-red
```

#### Containers [Give feedback](#)

A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. [Learn more](#)

☐ Only show running containers

Search

<input type="checkbox"/>	Name	Image	Status	Port(s)	Last started	Actions
<input type="checkbox"/>	<b>mynodered</b> 19c065ecf23a	<a href="#">nodered/node-red</a>	Running	<a href="#">1880:1880</a>	7 seconds ago	
<input type="checkbox"/>	<b>Mosquito_container</b> 717753d4f563	<a href="#">eclipse-mosquitto:latest</a>	Running	<a href="#">1883:1883</a>	0 seconds ago	

Now you can easily create files in the data volume inside Node Red container, we will need this to upload logs to AWS S3 bucket (and then maybe process it with EC2).

Click on manage palette and search for:

node-red-contrib-telegrambot

&

node-red-node-aws:

View

Nodes

Install

filter nodes

node-red

3.0.2

> 49 nodes

In use

node-red-contrib-telegrambot

15.0.1

> 7 nodes

remove

disable all

telegram bot

telegram receiver

telegram command

telegram event

telegram sender

telegram reply

telegram control

disable

View

Nodes

Install

filter nodes

node-red

3.0.2

> 49 nodes

In use

node-red-node-aws

0.2.3

> 4 nodes

remove

disable all

amazon s3 in

amazon s3

amazon s3 out

aws-config

disable

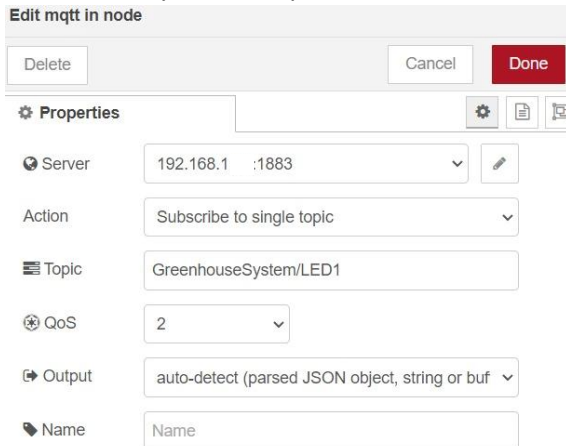
You can also use node-red-contrib-telegrambot-home, is easier to use and adds more functionality.



## SW test

Now it is time to check that is all correct.

1. Connect to MQTT server.
  - a. Use your local Ip, localhost:1883 does not work since is a container.



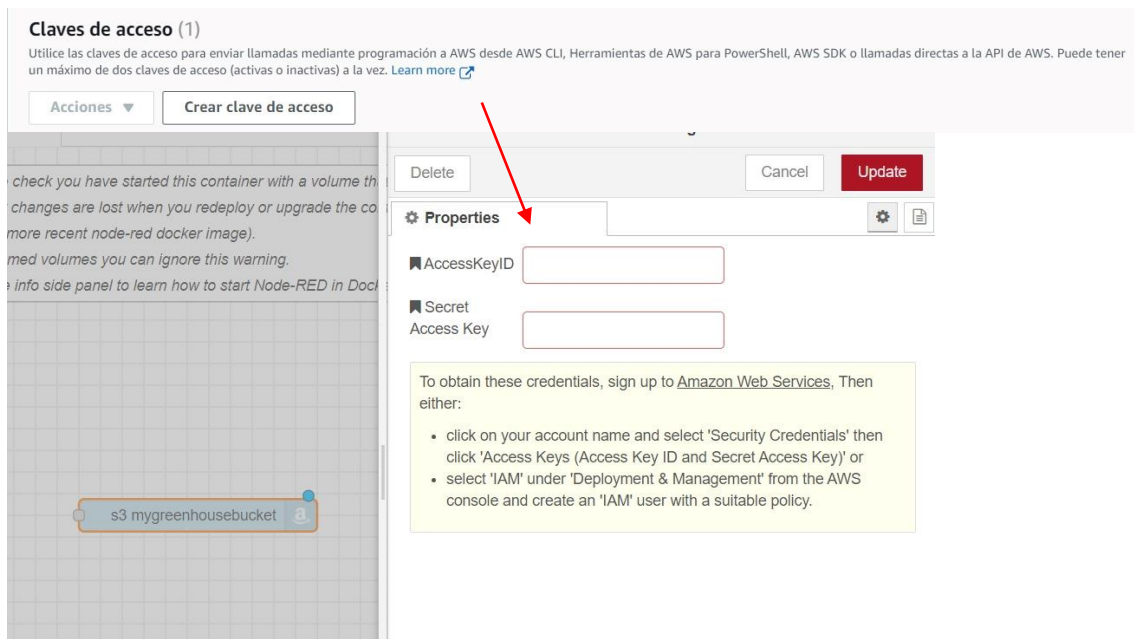
2. Make sure the telegram bot token and chat IDs are correct.
  - a. Parse msg.payload as follows to tell the bot which chat to type in:

```

1  var message = "LED 1 = " + global.get("myData");
2  // Configuramos el payload
3  msg.payload = {
4    chatId: -912627664,
5    type: 'message',
6    content: message
7  };

```

3. Check that S3 bucket is connected:
  - a. You will need to create access keys and put them into the node properties.



**Claves de acceso (1)**

Utilice las claves de acceso para enviar llamadas mediante programación a AWS desde AWS CLI, Herramientas de AWS para PowerShell, AWS SDK o llamadas directas a la API de AWS. Puede tener un máximo de dos claves de acceso (activas o inactivas) a la vez. [Learn more](#)

Acciones ▼ Crear clave de acceso

check you have started this container with a volume th  
changes are lost when you redeploy or upgrade the co  
more recent node-red docker image).

med volumes you can ignore this warning.

Info side panel to learn how to start Node-RED in Doc

s3 mygreenhousebucket

Properties

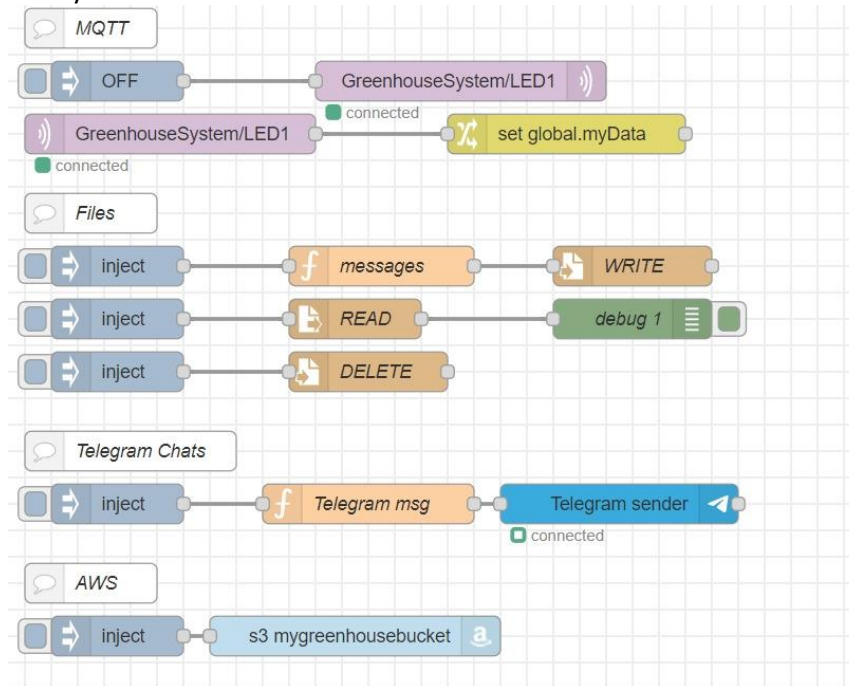
AccessKeyID

Secret Access Key

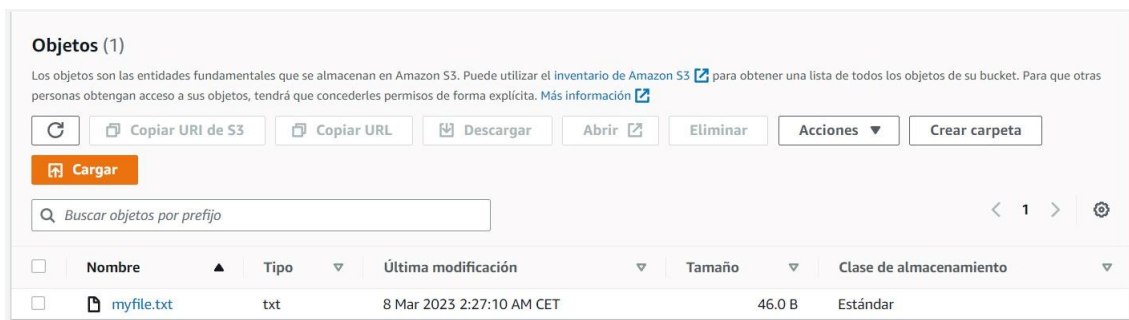
To obtain these credentials, sign up to [Amazon Web Services](#). Then either:

- click on your account name and select 'Security Credentials' then click 'Access Keys (Access Key ID and Secret Access Key)' or
- select 'IAM' under 'Deployment & Management' from the AWS console and create an 'IAM' user with a suitable policy.

Now you can create a test flow as follows:

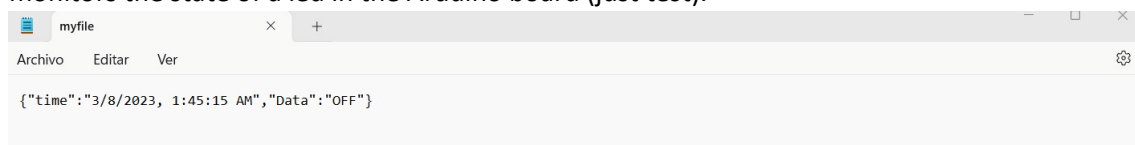


As we can see the test file is created in local storage, the data volume we created previously, and now we send that file to the AWS bucket.

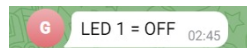


Inside this file we can see that the data collected is “OFF” with a timestamp.

This is because we created a topic called LED1 inside GreenhouseSystem general topic that monitors the state of a led in the Arduino board (just test).



Then we store the string “OFF” as a global or flow variable and we use a javascript function to send it to telegram:



Another function appends the msg to a file.txt and store in data volume.

**Edit amazon s3 out node**

Delete Cancel Done

**Properties**

**AWS** AWS

**Bucket** mygreenhousebucket

**Filename** myfile.txt

**Local Filename** /data/myfile.txt

**Region** eu-west-2

**Name** Name

Then this file is sent to S3.

Develop control algorithm

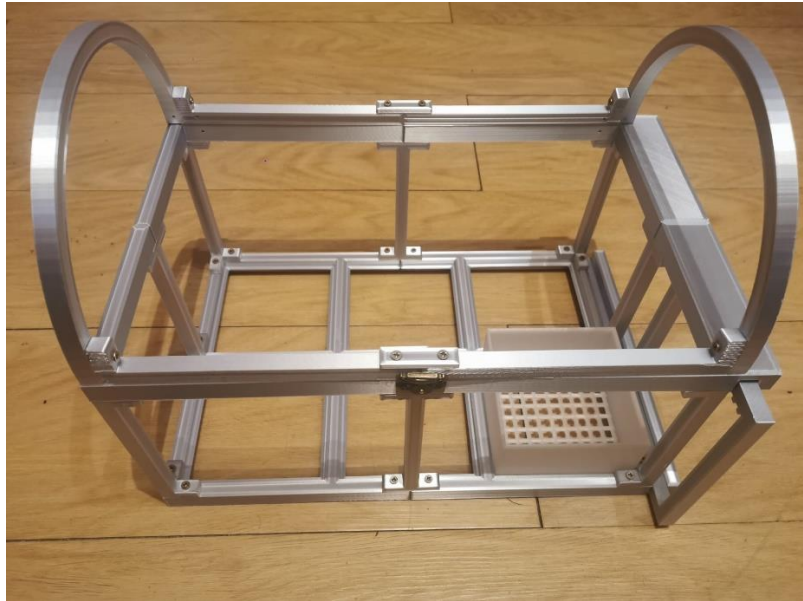
We will develop the control algorithms that will adjust the environmental conditions based on the data collected from the sensors. This might include using fuzzy logic or other machine learning techniques to optimize resource usage.

## Building and Testing

### Buy & Check

### Building the 3D mockup

This is how the complete impression of all the pieces has been and assembled. The next step is to include the sensors and actuators and finalize the model.



---

### Programming

## Testing

## State of Art (Professional Approach)

## Conclusions



## References

No hay ninguna fuente en el documento actual.