



instructables

Green House Automation



by Swastik Mohanty

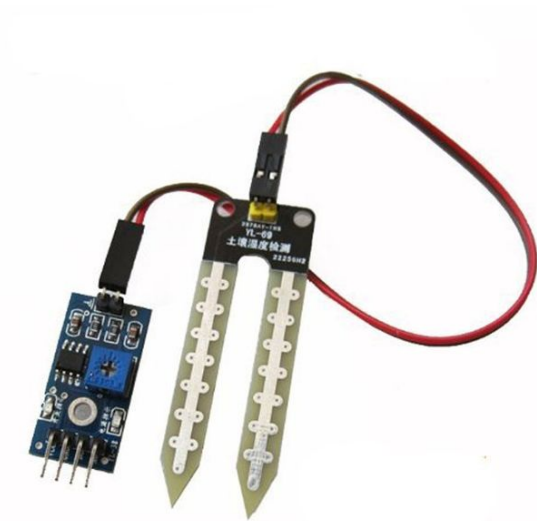
Green house automation is a project where three parameters of a green house, i.e. Soil Moisture, Temperature & Humidity, are monitored by the user remotely simply by using a web browser.

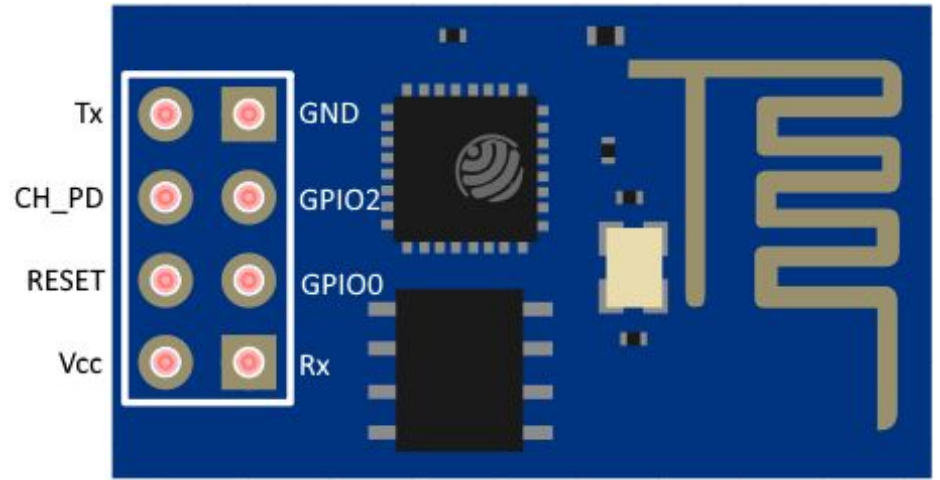
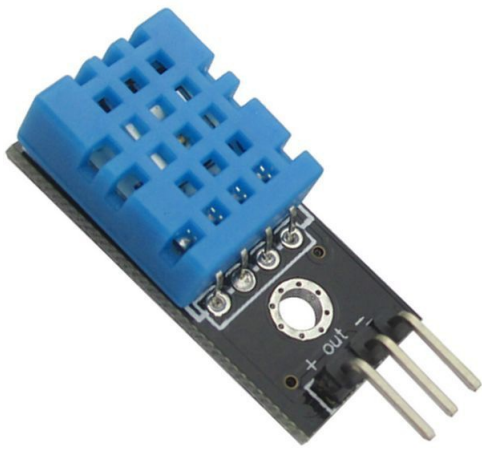


Step 1: Components Required

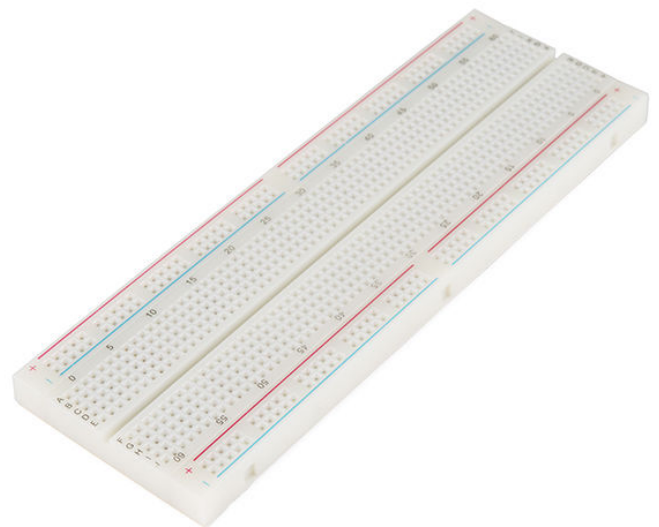
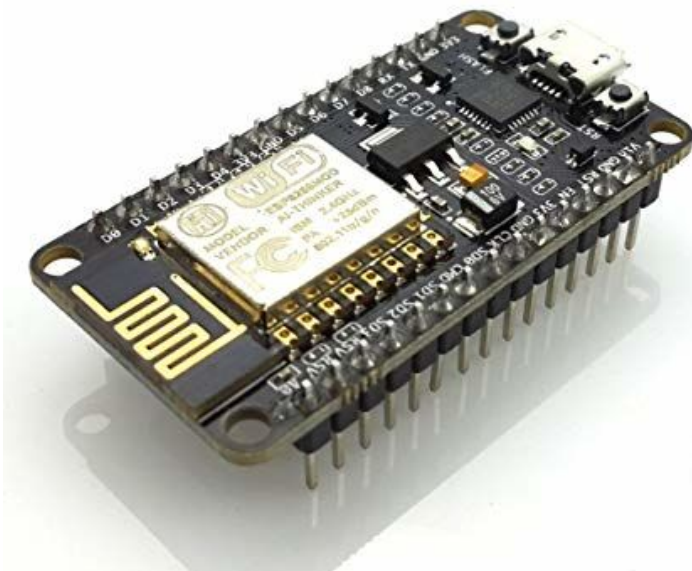
Some essential components required are listed below

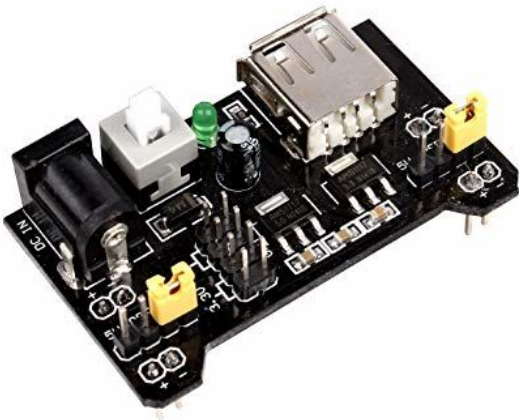
1. Raspberry PI Model B
2. NodeMCU Development Board
3. ESP8266 Wifi module
4. Moisture Sensor
5. DHT11 Temperature & Humidity Sensor
6. 5V Single Channel Relay
7. 5V Submersible Water Pump
8. Bread Board
9. Bread Board Power Supply Module





5V Relay Module 1





Step 2: Language & Protocol

- **C Language** is used for the micro controllers.
- **MQTT Messaging** : MQTT stands for MQ Telemetry Transport. It is a publish/subscribe, extremely simple and lightweight messaging protocol, designed for constrained devices and low-bandwidth, high-latency or unreliable networks. The design principles are to minimise network bandwidth and device resource requirements whilst also attempting to ensure reliability and some degree of assurance of delivery. These principles also turn out to make the protocol ideal of the emerging “machine-to-machine” (M2M) or “Internet of Things” world of connected devices, and for mobile applications where bandwidth and battery power are at a premium.
- **Python program** is used for automating the water flow and database connectivity.





Step 3: Eclipse Mosquitto MQTT Broker

Here I used the Mosquitto MQTT Broker for the easy message communication between the nodes.

Eclipse Mosquitto is an open source (EPL/EDL licensed) message broker that implements the MQTT protocol versions 5.0, 3.1.1 and 3.1. Mosquitto is lightweight and is suitable for use on all devices from low power single board computers to full servers.

The MQTT protocol provides a lightweight method of carrying out messaging using a publish/subscribe

model. This makes it suitable for Internet of Things messaging such as with low power sensors or mobile devices such as phones, embedded computers or micro controllers.

The Mosquitto project also provides a C library for implementing MQTT clients, and the very popular `mosquitto_pub` and `mosquitto_sub` command line MQTT clients.



Step 4: Flow of Data in the Whole Project

In the above image the nodes are

1. NodeMCU
2. Raspberry PI
3. ESP8266

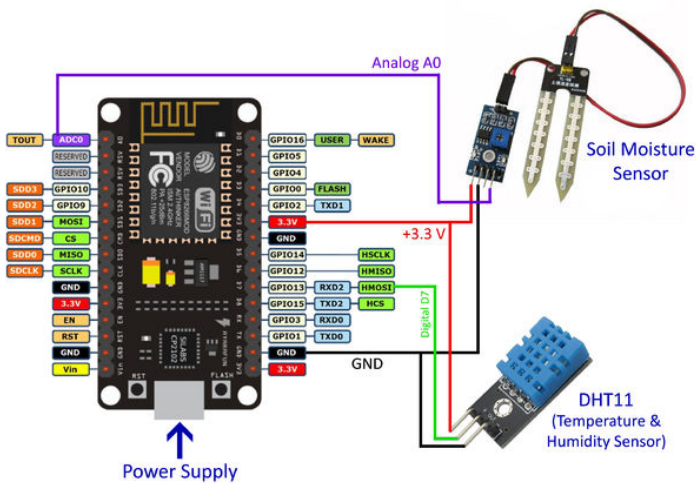
NodeMCU is the sensing part of the Green House & the ESP8266 is the actuating part which supplies the water when the ground needs water according to the sensors.

Raspberry PI contains the Mosquitto Broker and a Python client which subscribes the messages coming from the MQTT Broker and stores the data into a SQL server.

Step 5: Connection of Sensors With the NodeMCU

The DHT11 temperature and humidity sensor and the water moisture sensor is able to operate on 3.3 volts.

NodeMCU can't provide more than 3.3 volt. So the sensors can directly connected with the NodeMCU microcontroller board.



Step 6: Connection of the Submersible Water Pump With the ESP8266

A submersible water pump is used to supply the water whenever needed.

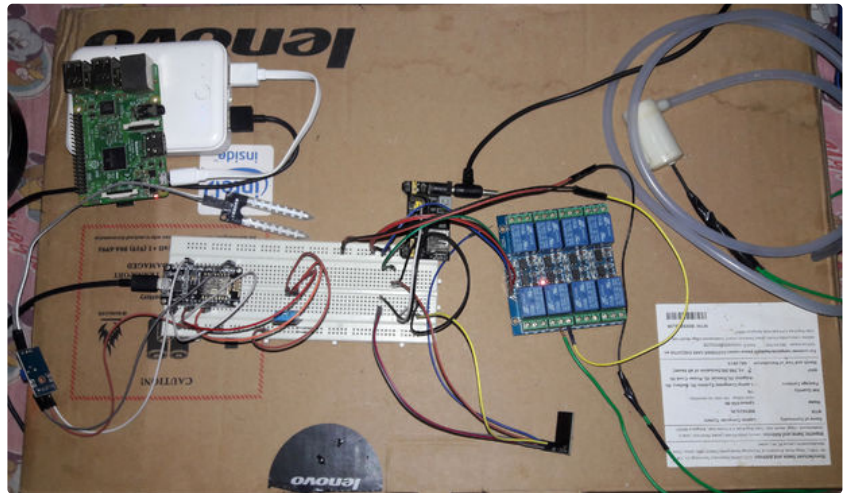
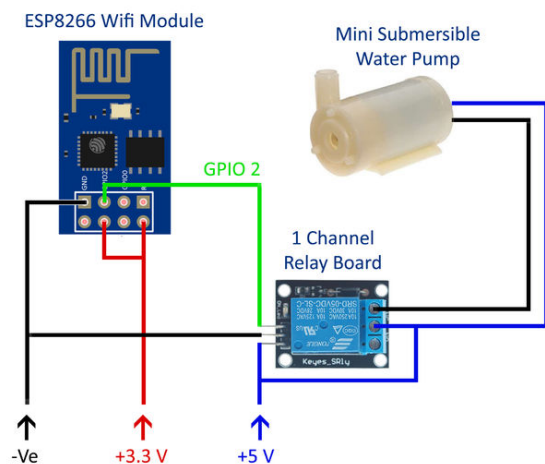
Water pump needs 5 volt power supply for it's operation.

A single channel relay is needed to connect the motor. When the GPIO pin of the ESP8266 is

activated the relay turned on and automatically supplies the water using the submersible water pump.

Here external power supply is provided to ESP8266 board, Relay & the submersible water pump.

My complete hardware connection is in above image.



Step 7: Installing Mosquitto Broker & Running Python Program in Raspberry Pi

Following are the steps for installing the Mosquitto broker in Raspberry PI

It should automatically start mosquitto.

Open the terminal and type the following commands

To Stop and start the service I needed to use

```
sudo apt-add-repository ppa:mosquitto-dev/mosquitto-ppa
```

```
sudo service stop mosquitto
```

```
sudo apt-get update
```

```
sudo service start mosquitto
```

```
sudo apt-get install mosquitto
```

Most sites I discovered where using the format.

```
sudo apt-get install mosquitto-clients
```

```
sudo /etc/init.d/mosquitto stop
```

Step 8: How MQTT Works?

MQTT is one of the most commonly used protocols in IoT projects. It stands for Message Queuing Telemetry Transport.

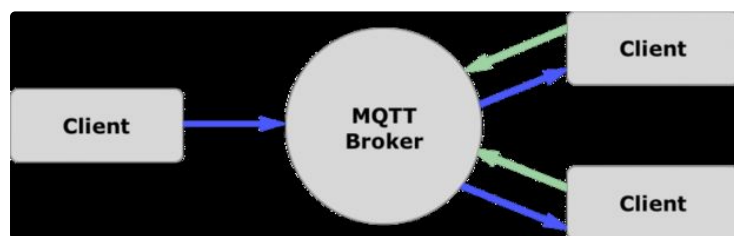
In addition, it is designed as a lightweight messaging protocol that uses publish/subscribe operations to exchange data between clients and the server. Furthermore, its small size, low power usage, minimized data packets and ease of implementation make the protocol ideal of the “machine-to-machine” or “Internet of Things” world.

clients and a server. Likewise, the server is the guy who is responsible for handling the client’s requests of receiving or sending data between each other. MQTT server is called a broker and the clients are simply the connected devices. So:

* When a device (a client) wants to send data to the broker, we call this operation a “**publish**”.



* When a device (a client) wants to receive data from the broker, we call this operation a “**subscribe**”.

Like any other internet protocol, MQTT is based on



Step 9: Programming NodeMCU and ESP8266

Following are the source code for NodeMCU and ESP8266 Microcontroller board

	https://www.instructabl...	Download
	https://www.instructabl...	Download

Step 10: Designing a Web Page and Connecting to the SQL Database

Web Page is designed using HTML, CSS and PHP language.

PHP is used to extract the sensor readings from the database and show it into the HTML page.


A python program is used as a heart of this project.

Works that are the python program doing is as follows.

1. It subscribes to a topic in which the sensor sends the sensor readings.
2. It publish water pump on/off command to the MQTT broker.
3. It stores the sensor reading into a SQL database.

Here in my case the python program and the SQL database is present in a Laptop. The web page running through a Local Host.

Following is the Source code of my python program.

 <https://www.instructabl...> Download

Step 11: Complete Working

Following are the steps in which the process goes on.

1. NodeMCU works as sensing part and reads the Temperature, Humidity and the soil moisture level.
2. It sends the readings to the MQTT broker with a topic "Topic 1"
3. In a laptop the python program is running and it subscribes to a topic "Topic 1" with the MQTT broker.
4. When the NodeMCU sends the readings then the Mosquitto MQTT Broker immediately sends the data to the python program.
5. Python program then calculates whether there is water needed in the Green House. Then it stores the readings into the SQL Database.
6. If water needed in the Green House, then the python program publishes water pump on/off message to the Mosquitto MQTT broker with a topic "Topic 2"
7. ESP8266 works as an actuator. It subscribes to the topic "Topic 2" in which topic the python program is publishing the message. When the python program publishes any message then the message is immediately transferred to the ESP8266. According to the on/off message, it turns on/off the submersible water pump.
8. Last phase to display the live readings in the web page. The web page fetches the data from the SQL database in which the python program stores the data directly and displays the readings in the page.

