SCHOOL OF SCIENCE AND ENGINEERING

Smart Farming System
Monitoring and Control of Some Agriculture Features Using an Arduino Based System

By: Salwa Mitouilli
Supervisor: Dr. Fouad Berrada

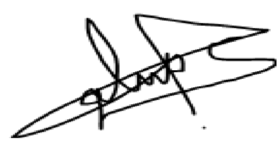
Report – spring 2020

Capstone Report

Student Statement:

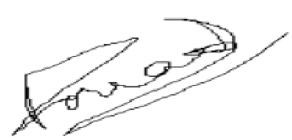
This is to officially acknowledge that the present project “Smart Farming” focuses on the development of a system that I (Salwa Mitouilli) designed and that Dr. Fouad Berrada supervised respects the ethics guidelines stated by AUI. To accomplish the work, I observed a lot of precision and caution to reach the initial goals. To my knowledge, the final prototype meets and includes all the desired initial criteria that sound useful and that have an added value in the real life of farmers, respecting the safety of workers, and it can be used friendly with a minimum of training.

Worth mentioning, I have been following the guidelines and regulations made available on the AUI portal by Dr. Salih Alj, the SSE Capstone projects coordinator, to meet and fulfill my CSC bachelor’s program requirements.



Salwa Mitouilli

Approved by the Supervisor(s)



Dr. Fouad Berrada

ACKNOWLEDGEMENTS

As an AUI student seeking a bachelor degree in Computer Science, I have successfully completed this Capstone project under the guidance and close supervision of Professor Fouad Berrada, to whom I would like to express my sincere gratitude for being always available, and for his generosity and patience. His encouragements and continuous support gave me the strength and the courage to move forward and to accomplish the required tasks. His experience in Biology and agriculture was a real asset to our success in the realization of the present project.

Also, I would like to thank my mother, brother and sister for their love, help and their full support. Special very big thanks to my father Mr. Ezzaki Mitouilli who dedicated his time, knowledge, advices and unconditional support to achieve this project.

Last but not least, I would like to express my deep gratitude for being a student in this wonderful university that offers all necessary equipment/labs, IT and library facilities that ensure a perfect environment for studies, in addition to the exchange programs with dozens of universities worldwide. Also, I would like also to stress on the importance of AUI scholarships to its staff family's members. Thank you AUI for supporting us.

CONTENTS

LIST OF FIGURES

ABSTRACT

Nowadays Internet of Things (IoT) technology is one of the fastest growing fields in different domains including agriculture. IoT improves the quality of our lives by bringing and fostering changes in many fields of activities to make them become handy, smart and endowed by sufficient artificial intelligence. Thanks to this technology, Smart farming systems know a cultural change toward modern agriculture which is more productive, consuming less water and even cheaper. The main goal of my project is to use IoT in the agriculture field in order to collect data instantly (soil Moister, temperature…), which will help one to monitor some environment conditions remotely, effectively and enhance tremendously the production and therefore the income of farmers. The present prototype is developed using Arduino technology, which comprise specific sensors, and a Wifi module that helps to collect instant data online. Worth mentioning the testing of this prototype generated, highly accurate data because while we were collecting them remotely any environmental changes were detected instantly and taking in consideration to make decisions.

Chapter I: Project Background

Introduction:

The present project deals with smart farming system, which would allow farmers access to live data such as temperature, humidity, and soil moister. The report is structured as follows:

The first chapter is a general knowledge about the IOT technology and the project's topic. The second chapter contains the steeple analysis, feasibility study, project's requirements, specifications, methodology used, project's design process, implementation, components and technologies used, developed algorithms, the future scope and conclusions.

IoT and agriculture:

IoT contains a solid mainstay of several technologies that enable networks of wireless sensors such as, embedded systems, big data, cloud computing, web services, and computer networking and protocols.

In agriculture's fields Internet of Things have several benefits, encompassing:

The various sensors designed for this specific field of farming that gives the opportunity to work remotely on many projects related to agriculture.

The different used sensors help instantly to collect and store data easily in cloud computing services. These live data can be accessed promptly and from any intelligent smart device.

As approved by experts, farmers can use the IoT systems to increase their productivity as well as the quality of their products [5].

In fact, it increases profits/incomes and reduces significantly their costs.

Having access promptly to exact accurate data helps in increasing the efficiency level in the use of water, pesticides, and fertilizers amounts managements.

Thus, IoT would help in protecting the environment and ecosystems as well.

Literature Review:

According to Andrew Moela [4], in the next decades, the farming industry is expected to become more important than any other time before.

It is expected by the United Nation that the world population of the world by 2050 would reach 9.7 billion, which would require a rise in the global agricultural goods production in agriculture to about 69%. To reach this vision goal, farming companies started to adopt the Internet of Things for accurate analysis and higher better production of agricultural goods.

The IoT is a technology that came out to help in pushing agricultural fields into a higher level. Nowadays, smart farming is already adopted by a number of modem farmers and its use is increasing and becoming more and more common among the new generation of educated young farmers. In the modern agriculture the use of sensors, drones, and high-tech agriculture technologies is becoming quickly the new norm [4].

The collection and the analysis of big data in agriculture will represent a very big deal in the future of modem farming, in preserving ecosystems and it would help the overall production growth of developing countries, such as Morocco. In fact, IoT technology offers more benefits in the real life. Researchers are doing more investigations into this technology toward a wider use and for a maximum of profits [4].

Chapter II: Steeple Analysis

The steeple analysis is a process that helps to make the right decision by considering seven factors (Societal, Technology, Environment, Ethics, Political, Legal, and Economic). These variables show the benefits and disadvantages of the products or of the companies. This step is very important to discover the threats of your products and also to highlight their strengths and benefits. Therefore, in this project I will use this method to analyze some macro-environmental factors:

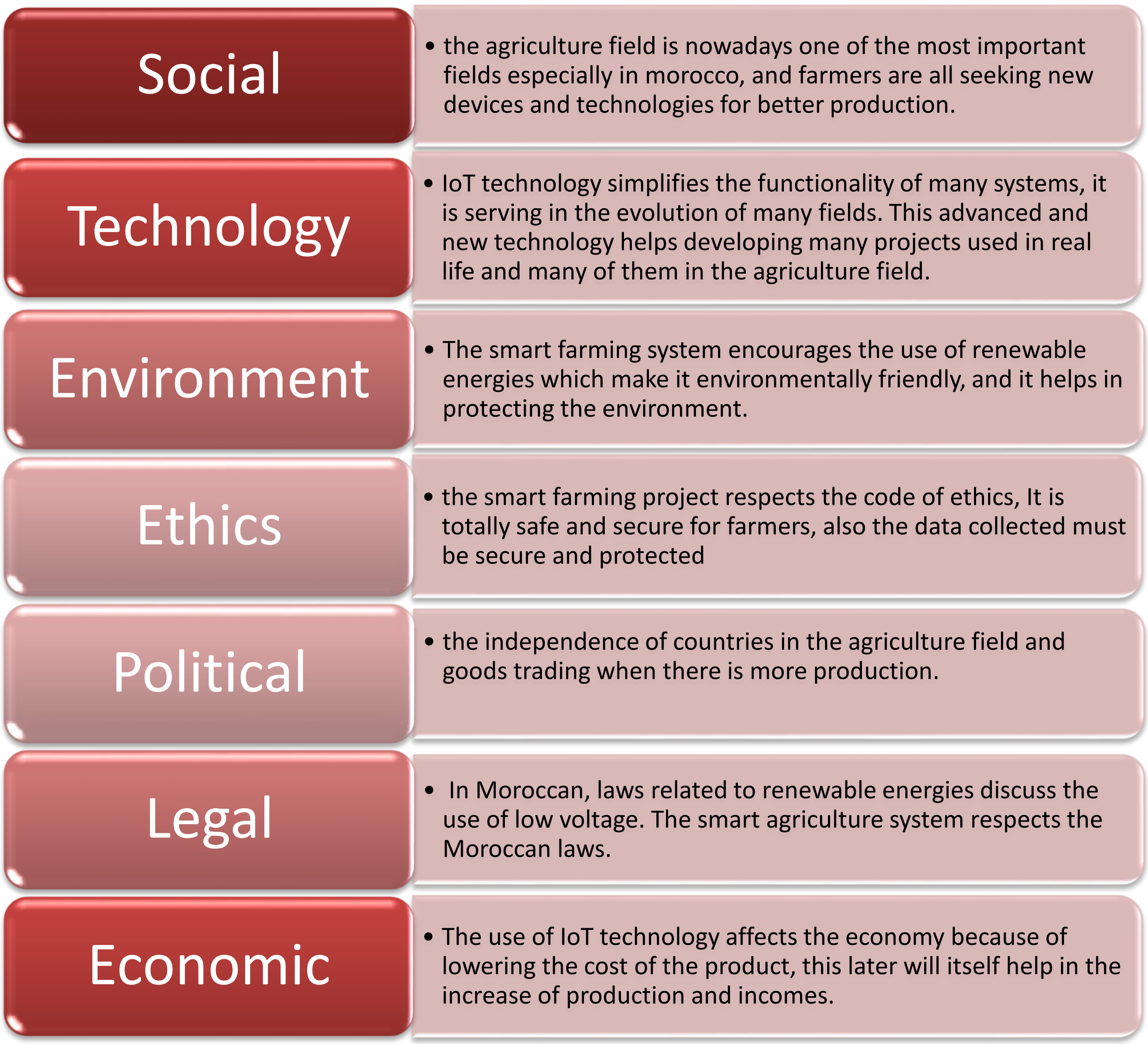


Figure1. Steeple analysis

Chapter III: Requirements specifications

Functional Requirements

The functional requirements indicate the functions and services of the present system. They describe the behaviour of the system in relation to the needs:

• Measure Temperature.

• Measure soil moister.

• Display the sensor readings on the LCD screen.

• Calculating the date and time.

• Irrigating the soil if needed.

• Turning on the fan if needed.

Non-functional Requirements:

The non-functional requirements for the present system consider the following:

• **Availability:**

The proposed product can be available and operable successfully all the time.

• **Reliability**:

The system provides an accurate measurement of data, and it can have a longer lifespan.

• **Maintainability**:

The present system can be improved easily by integrating new components with enhanced features.

• **Simplicity**:

The proposed system is user friendly. The usage of this product doesn't require any prior learning.

Chapter IV: Feasibility Study

Feasibility analysis begins once the objectives are defined. It starts by proposing different prototypes, test them, bring modifications to improve them and ultimately keep the most promising one.

Our analysis should provide us with enough information to make reasonable estimations concerning the project cost as well as indications on how the new system will respond to user's specific requirements. We will be flexible in the sense that if we notice that we are not going to reach the specified initial goals to propose new ones that we deem appropriate.

In the feasibility analysis, we will focus on the most promising prototype that should be efficient and affordable taking in consideration the followings:

The Economic Feasibility:

In this project, I will consider the cost of the material to be used in the development of our new system as well as the fees related to the field work. Regarding the materials, I bought the different components of Arduino (Nodemcu ESP8266 board, power supply, temperature sensor, humidity sensor, soil moister sensor, LCD display, cables, resistances, and a watering pomp).

Metal/ Plexiglas, nails/ screws, plastic cover, peat, seeds/plants and drip irrigation for fertigation were also purchased. The total cost to buy the different materials and Arduino components is around 2500dhs.

Then, I developed a 40x60 cm prototype and arranged it in such way that I can test remotely the effect of some factors on plants growth, such as soil moister, temperature, etc.

The Technical feasibility:

For the technical feasibility part, Arduino components and chips are used. Technical evaluation should assess whether the needed Arduino components are available and if I will be able to combine and program them toward a functional final device. Worth mentioning, this was perfectly achieved as it will be shown in my final report.

Operational Feasibility:

This system will be quite easy and simple to operate. Therefore, no special training should be given to farmers. Technical performance should address issues related to the system information accuracy, and provided promptly and remotely through a wireless service.

Schedules and timelines:

The timetable below shows the estimated allocated duration of each task.

The period of execution and completion of the present project is supposed to be from February 3rd to March 21st.

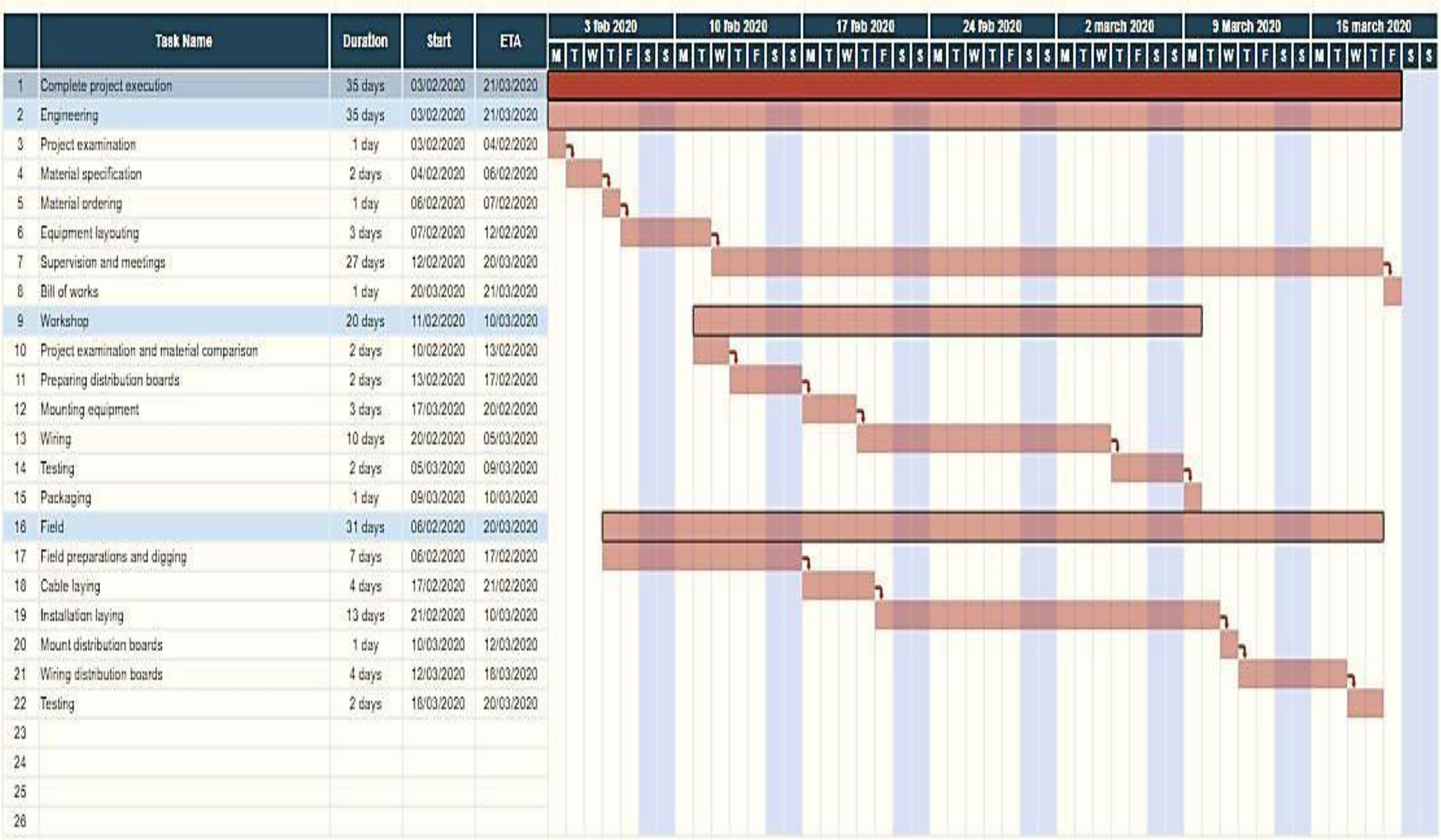


Figure 2. Timetable

Chapter V: Methodology & Design

The design of the project comes right after we are done with the project's analysis. Our system will be implemented based on many small functions. These functions should be capable to perform specific tasks. The whole system is the sum of all those functions. Therefore, our strategy is to use the function oriented design. DFD design process can be used to depict how functions change data and state of the entire system functions.

The implementation will go along with the testing to make sure that all functions are implemented properly. Still we are going to test in a field with pre-existing data and compare these results to ours, using our new system. We can then go further and test in different fields to see the level of accuracy in different types of soils. Given that we have several functions, several tests can be performed, so we may have to test each function at a time to make sure that all of them work correctly and give accurate outputs. Finally, we will test all of them together at the same time in a single system, knowing that some functions can be related to others so we may not separate them in the testing.

By implementing this system, agricultural lands, parks, gardens, golf courses and AUI's gardens can be irrigated in a more automated and efficient way. In addition, our system will be cheaper and safe, compared to other automated systems. While implementing this project, the user requirements and specifications will be taken in consideration, as well as respecting the design method and the testing strategy would be concisely developed.

The Diagram of the overall process:

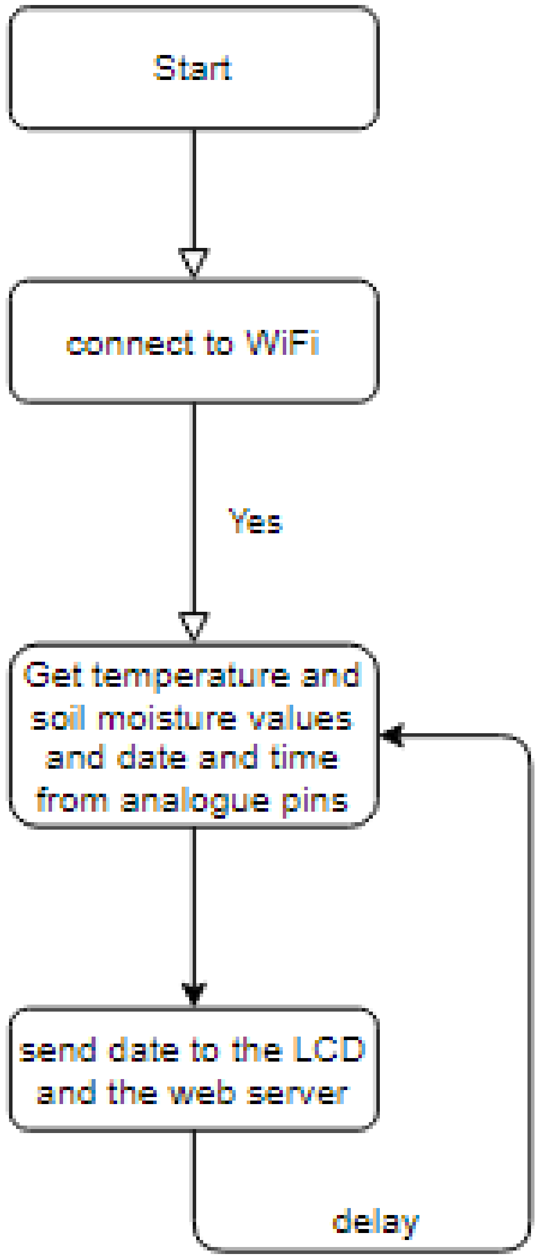


Figure 3. Design Diagram

Chapter VI: Description of the smart farming system

Definition of the Smart Based System:

The Smart Farming system is a project depending on IoT technology, using Arduino. To develop this project, I used many Arduino's hardware components, including NodeMcu module as the main grid or the microcontroller, and that will connect to Wi-Fi. Other different sensors such as, temperature sensor (LM35), soil moister sensor, LCD, relay, RTC module for date and time. In addition to some connectors and wires. All these components will be connected based on a specific circuit. For the software part, I used the Arduino application that uses C++ code. The code considers several libraries each one is related to specific component or sensor. I created also a web server where data collected would appear instantly and this by adding an HTML, CSS and JavaScript code.

The Hardware Components:

1. **Components and Modules**

In this part, I will define all the hardware components used in this project. The components are mainly composed of the sensors and the grids linked with cables.

1.1. NodeMcu ESP8266:

The ESP8266 NodeMcu is a microcontroller that allows connecting to Wi-Fi.

The ESP8266 can be programmed in several ways:

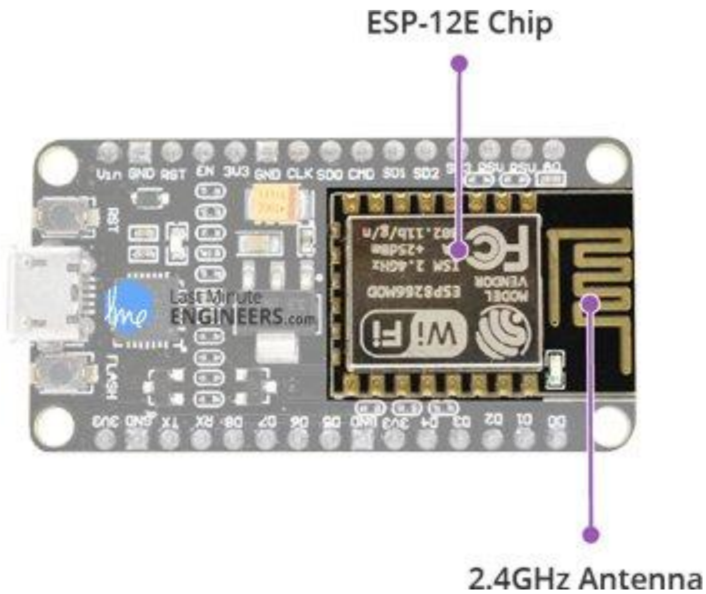
With Lua scripts, interpreted or compiled, with NodeMCU firmware

In C ++, with the Arduino IDE

In JavaScript, with the Espruino firmware

In MicroPython, with MicroPython firmware

In C, with the Expressive SDK or with the esp-open-sdk3 SDK

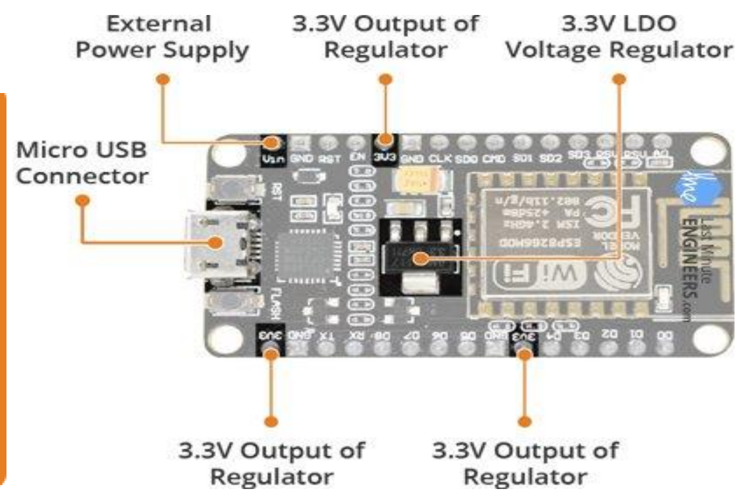


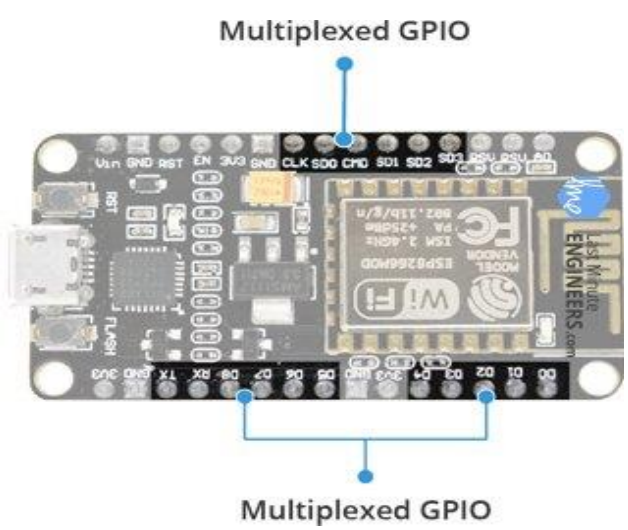
ESP-12E Chip

- Tensilica Xtensa® 32-bit LX106
- 80 to 160 MHz Clock Freq.
- 128kB internal RAM
- 4MB external flash
- 802.11b/g/n Wi-Fi transceiver

Power Requirement

- Operating Voltage: 2.5V to 3.6V
- On-board 3.3V 600mA regulator
- 80mA Operating Current
- 20 µA during Sleep Mode





Multiplexed I/Os

- 1 ADC channels
- 2 UART interfaces
- 4 PWM outputs
- SPI, I2C & I2S interface

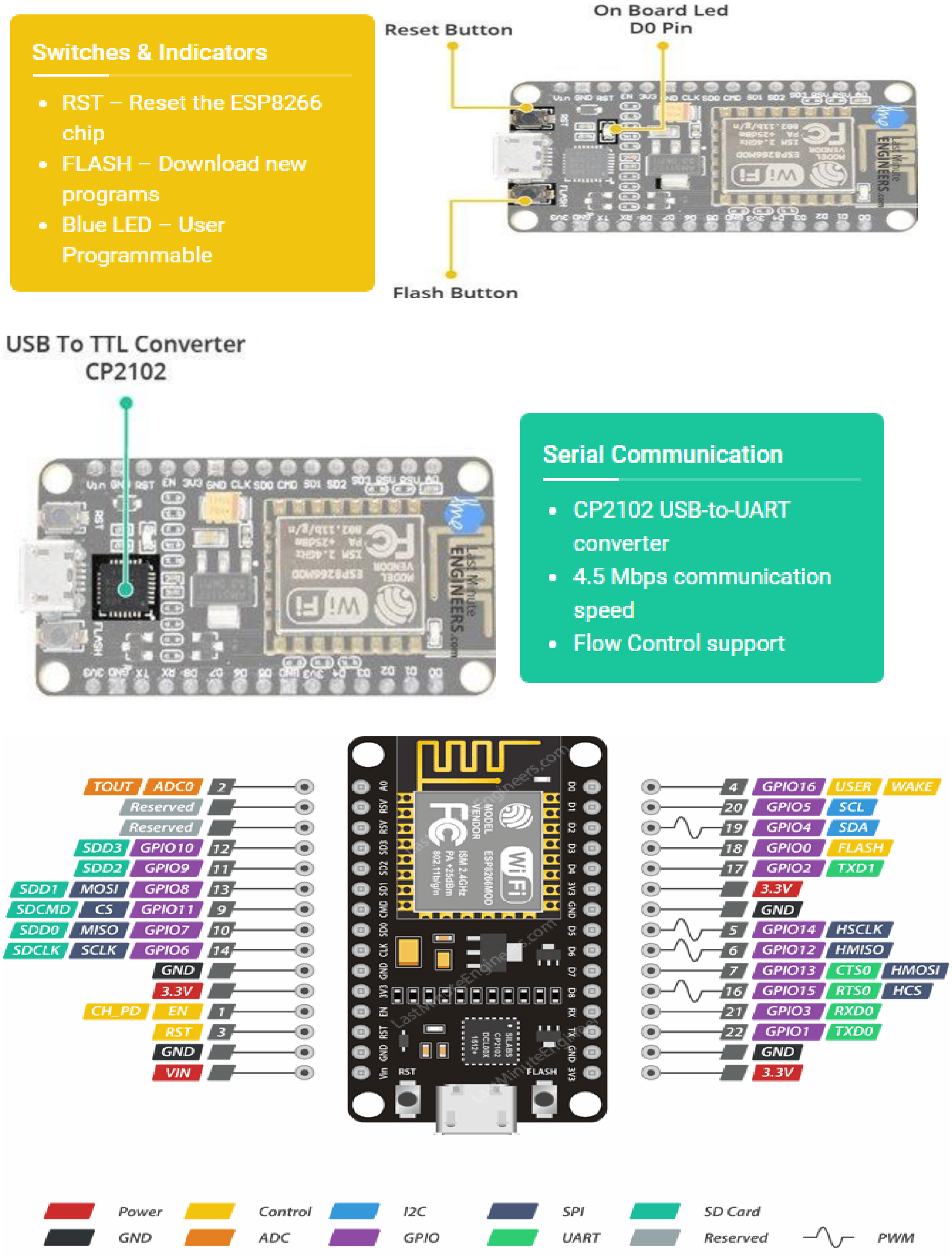


Figure 4. Nodemcu [3]

1.2. LM35:

The LM35 sensor is a precision integrated-circuit temperature device, contains an output voltage compatible with the Centigrade temperature. It is:

- Calibrated Directly in Celsius
- Linear
- 0.5°C Ensured Accuracy (at 25°C)
- −55°C to 150°C Range
- Suitable for Remote Applications
- Low-Cost
- Operates From 4 V to 30 V
- Low Self-Heating

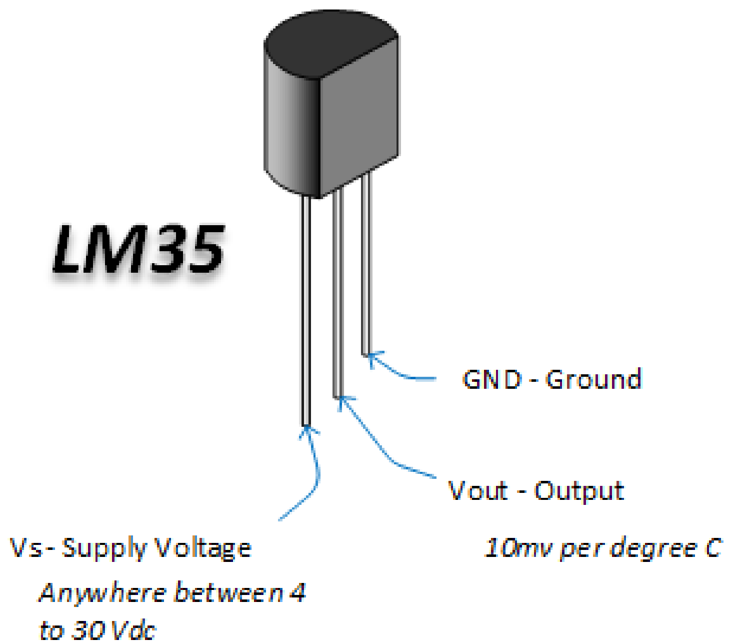


Figure 5. LM35 [6]

1.3. Soil Moister Sensor :

The soil moisture sensor composed of two probes used to measure the volume of water in the soil. These two probes permit the current to cross the soil, and then measure the moisture value from the resistance value.

More electricity will be conducted from the soil if there is a big amount of water. Therefore, the resistance would decrease and the moister level would increase. If the soil is dry, the conducted electricity is poor, which means there are more resistance and therefore less moist level.

The soil moisture sensor has the following specifications:

Input Voltage	**3.3 – 5V**
Output Voltage	0 – 4.2V
Input Current	35mA
Output Signal	Both Analog and Digital

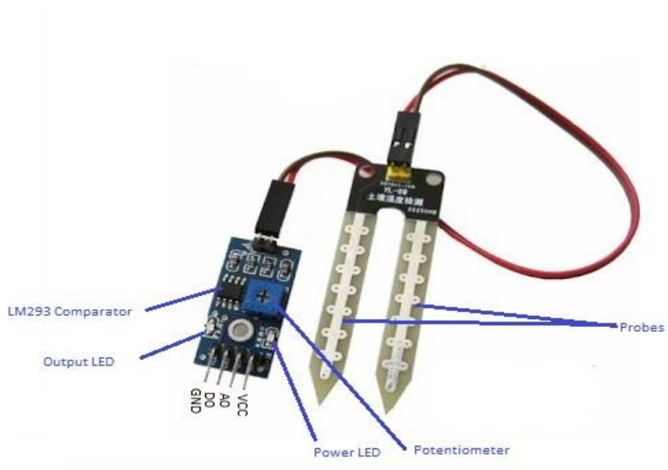
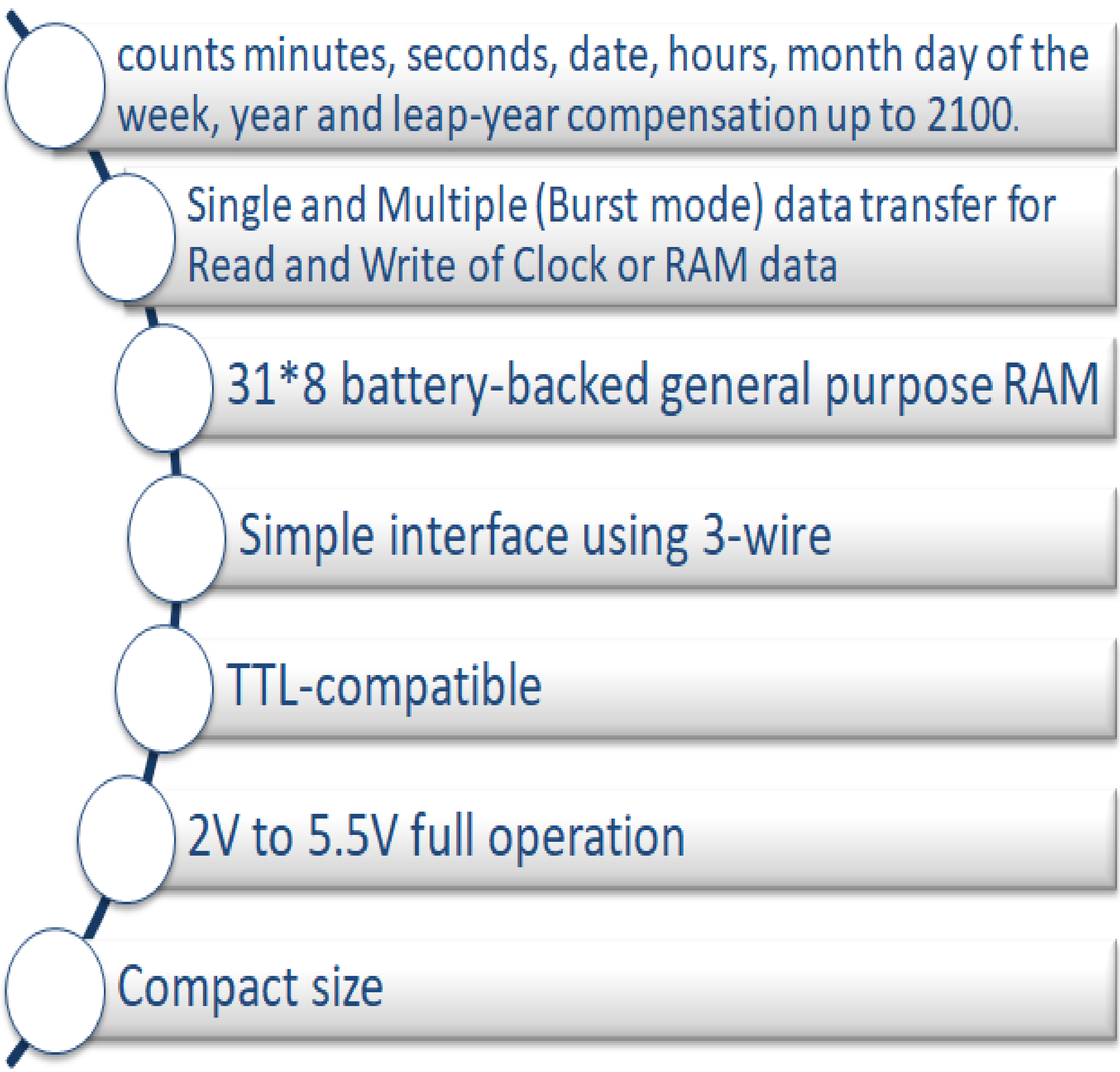


Figure 6. Soil moister sensor [1]

RTC Module DS1302 :

RTC Module is a real-time clock that allows getting the instant date (day, month, and year) and timing (hours, minutes, and seconds).

Features of the RTC Module:



Technical Specifications

• Supply voltage (typically):	3.3V
• Voltage range on any pin (with respect to ground):	-0.5V to 7.0°C
• Operating Temperature (Commercial):	0°C to +70°C
• Operating Temperature (Industrial):	-40°C to +85°C
• Storage Temperature range:	-55°C to +125°C
• Soldering Temperature:	260°C

Figures 7. RTC Module [1]

1.4. relay :

The relay is a component having three edges that are VCC (voltage), GND and TTL attached to the Nodemcu. I used two relays, the first one gets the soil moister data from the Nodemcu, it is “ON” when the soil is dry (soil moister voltage is high), in order to operate the watering pomp, and it is “OFF” when the water level is enough in the soil (soil moister voltage is lower). The second relay turns on the fun when the temperature is very high.

Characteristics of the relay:

Dimensions:	length 4.0 cm, width 2.7 cm and height 1.8 cm Weight 14g
Max voltage:	240V AC and 30V DC
Max current:	10A.
Coil voltage:	5V
Power:	0.36W
Coil resistance:	55 Ohm

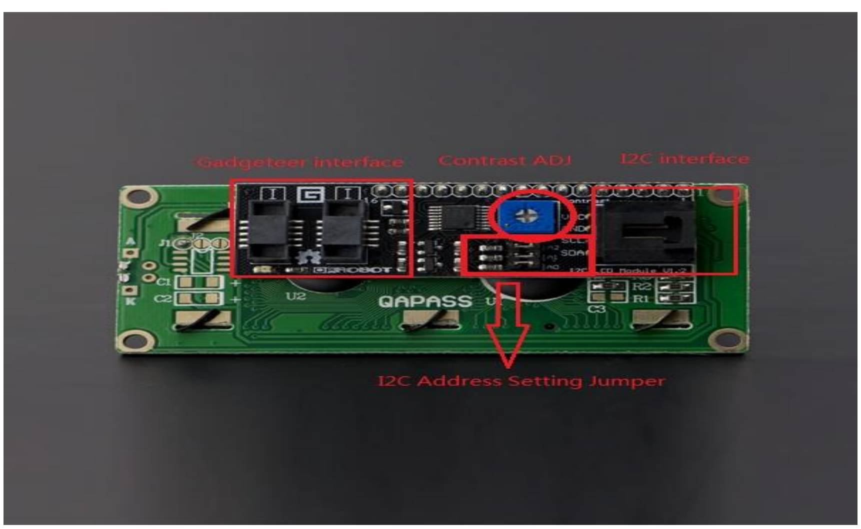
See Figure 8 in Appendix

1.5. LCD Module:

The LCD display screen with the I2C communication interface. It is able to display 2 lines and 16x2 characters. Arduino LCD display projects usually run out easily of pin resources, and its wiring connection is also quite difficult. The I2C LCD display contains only 4 pins: VCC, GND, SDA, and SCL. It will save at least 4 digital/analog pins on Arduino.

LCD Module specifications:

- Compatible with Arduino/Genuino UNO, Leonardo, Mega, 101 (Intel Curie), Micro, Nano, Mini
- I2C Address:0x20-0x27(0x20 default)
- Back lit (Blue with white char color)
- Supply voltage: 5VInterface:I2C/TWI x1,Gadgeteer interface x2
- Adjustable contrast
- Size: 80x36x20mmz(3.1x1.4x0.7in)



Figures 9. LCD [2]

1.6. Connectors:

I used different wiring types and connectors to build this prototype and link all my components together. These connectors used are the following:

- Female/Male Color Ribbon Flat Cable Jumper Dupont 40-wire 20cm 2.54mm (Figure 10)
- 150 mm wire with male connectors on the ends (Figure 11)
- 20cm Female to Female Circuit Board Cable Colorful Wire (Figure 12)
- Mini USB cable of Nodemcu (Figure 13)

See the pictures in the Appendix.

1.7. Power supply :

For the power or the energy source, I choose to use a solar renewable clean energy as a source of power for my prototype. A solar panel is used as an additional source of electricity (solar energy EP-0606A), with the following characteristics:

model	6V 6W
Solar cell	Polycrystalline silicon
Dimensions	29x19.5x1.8 cm
Output wire	3M
Application range	Charge 3V-5V batteries
Water-proof	IPx6
Optimum operating voltage	6V
Optimum operating current	1000mA
Open-circuit voltage	7.2V
Short-circuit current	1150Ma
STC	Irradiance 1000W/m^2? Module temperature 25 C, Solar spectrum AM=1.5

See the solar panel figure in the Appendix (Figure 14)

Description of Circuit:

In our project's circuit the programmed NodeMcu grid is connected with all different components: soil moisture sensor, temperature sensor, LCD screen, RTC module, and Relay modules. NodeMcu is also used as a Wi-Fi module. This model works based on collecting data from all the sensors and sending them to the NodeMcu. The NodeMcu gives the updates of data in web server through cloud computing. The real time data comes through Wi-Fi to the web server which we can access by typing the address IP given to the Wi-Fi module (NodeMcu) in any web browser. The Nodemcu also send data to the LCD and to the two relays. One relay is connected to the pump and the other to the fan.

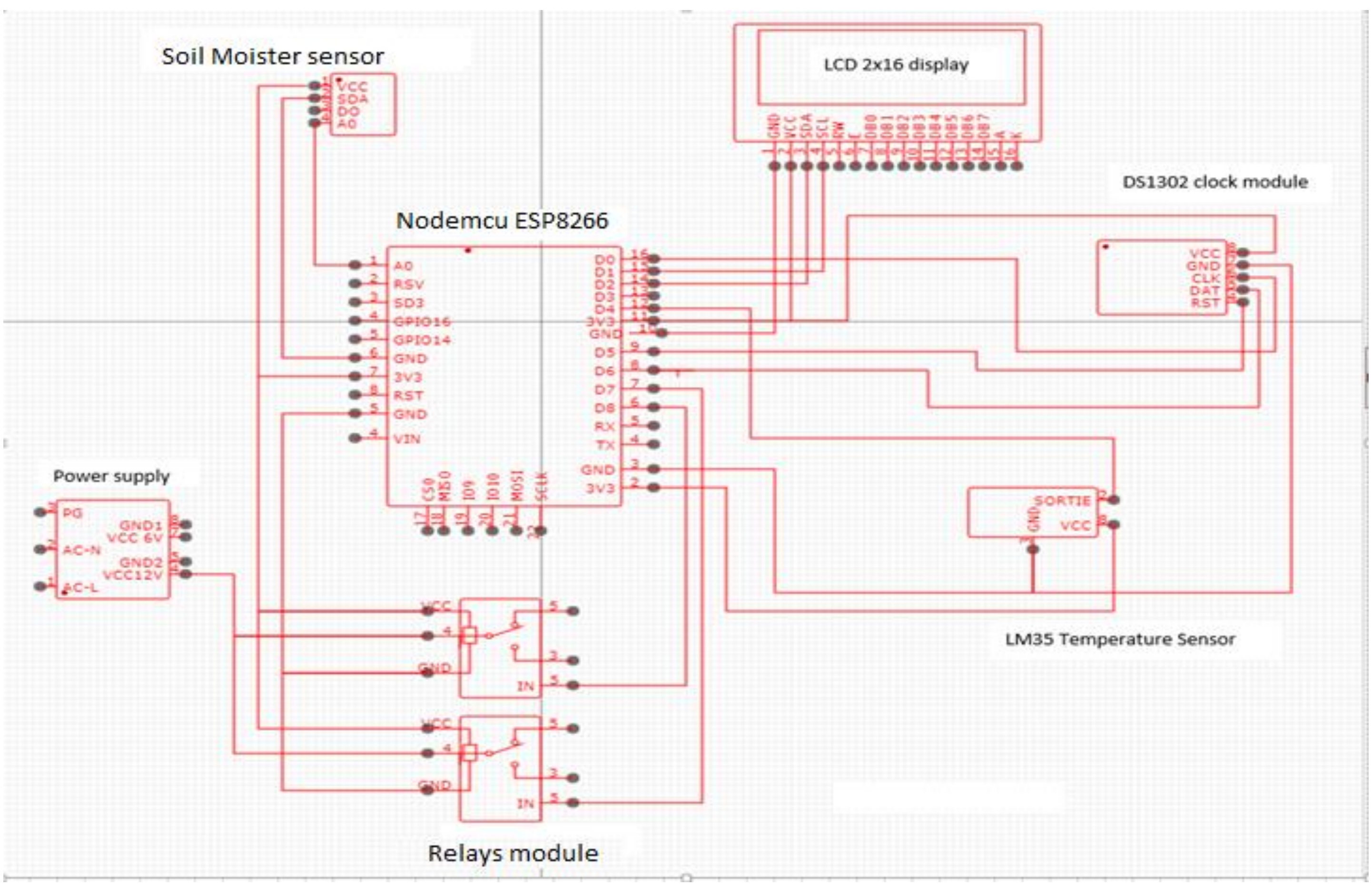


Figure 15. circuit

The following are the wiring information between each component and the NodeMcu module:

- **Temperature sensor:**
  GND –GND
  Vs – 3.3 V
  Vout – D4

- **Soil Moister Sensor:**
  GND – GND
  VCC – 3.3V
  A0 – A0

- **Relay1 (for the pump):**
  GND – GND
  VCC – 3.3V
  In – D8

- **Relay2 (For the fan):**
  GND – GND
  VCC – 3.3V
  In – D7

- **RTC Module:**
  VCC – VIN
  GND – GND
  CLK – D0
  DAT – D6
  RST – D5
- **LCD screen:**
  GND – GND
  V – VIN
  SDA – D2
  SCL – D1

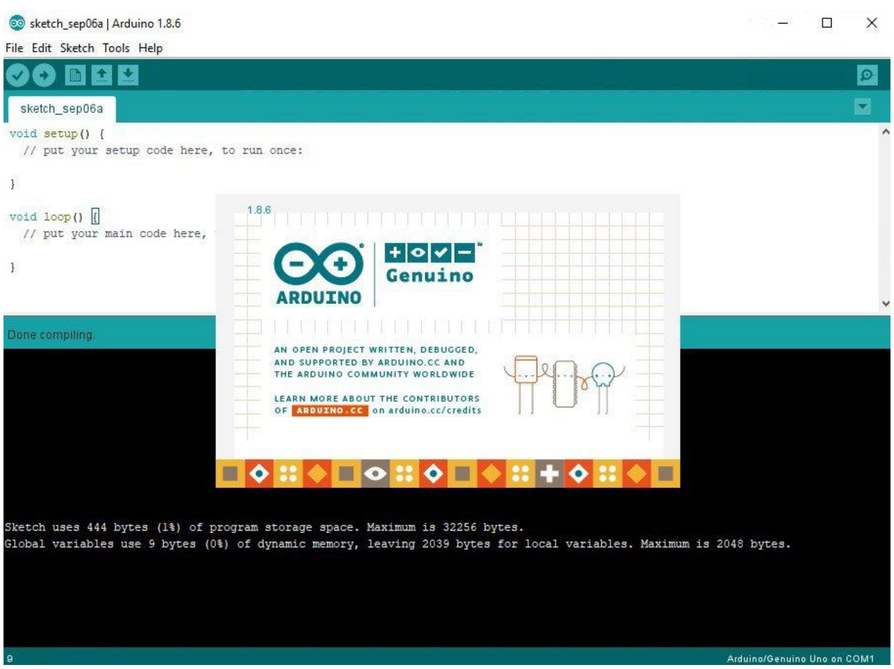
The Prototype:

I called prototype all the hardware components within the small field I built. This prototype contains a small greenhouse that I built using metal and Plexiglas and where I've grown my plants. I equipped this greenhouse with the Arduino components all together in a small box, another box of power supply unit, a solar panel, a fan, a watering system and a small lamp. The Arduino unit can work either using the solar energy or electricity.

- The electric box : See the figure 16 in the Appendix
- The greenhouse : See the figure 17 in the Appendix

The Software Component:

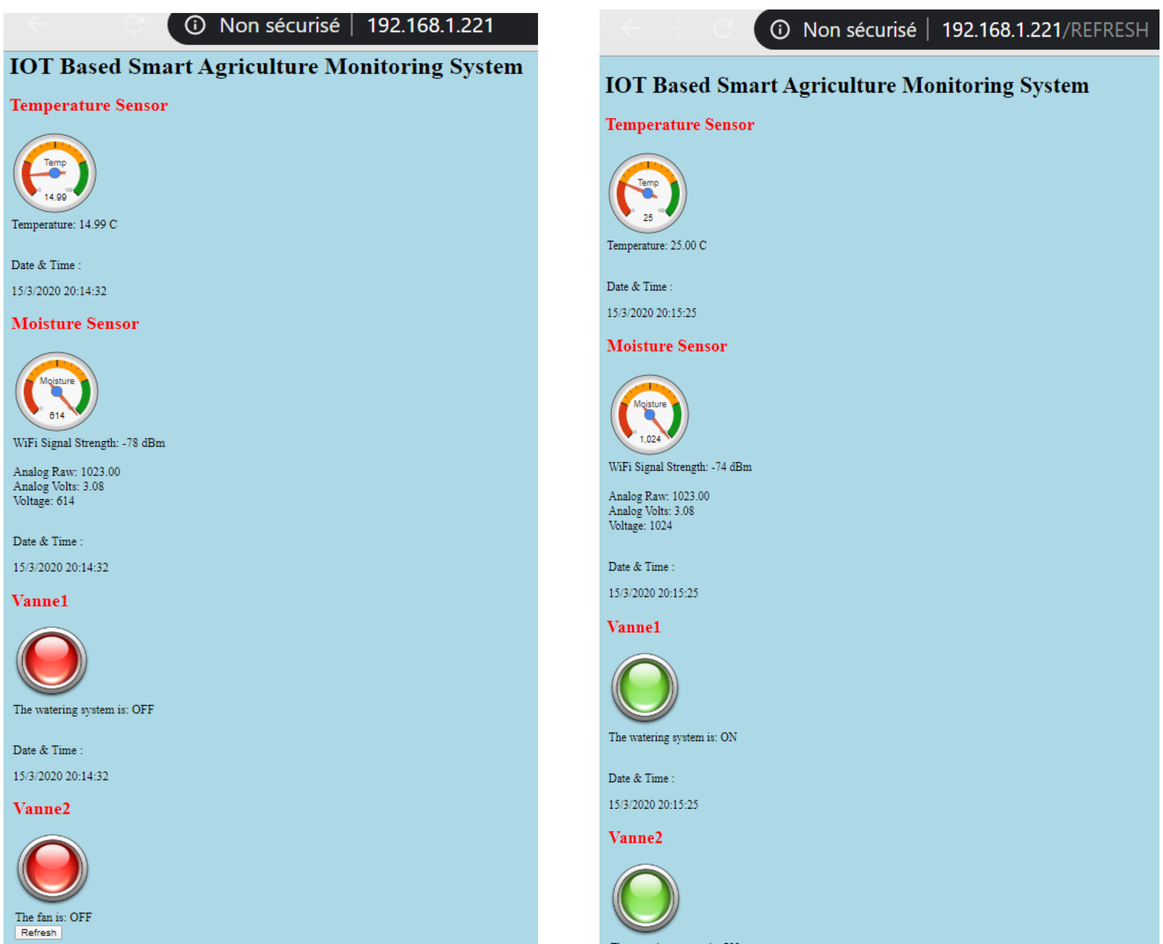
Smart Farming system was developed using the Arduino Software (IDE) which enable to write the code and upload it to a microcontroller board (Arduino Uno or NodeMcu…). It can be installed and run in Windows, Linux or Mac OS. This environment is written in languages C and C++.



Figures 18. Arduino IDE

Chapter VII: Web Server:

The web server is a web page that I can access using the IP address of the Nodemcu, it can be open in any browser but only using the same internet that the Nodemcu uses. This web server was created using html, css and javascript. But it’s code is used within the Arduino code and printed using C++ language (the code is presented and explained below in the algorithm section). The data collected using the hardware component is displayed instantly in this web server and projected. It will be refreshed every 10sec to show the new data.



Figures 19. Web server

Chapter VIII: THE DATABASE MODULE:

ENTITY RELATIONAL DIAGRAM (ERD):

An entity-relationship diagram (ERD) demonstrates the logical structure of databases; it defines the entity sets stored in a database, their attributes, and shows the relationships between them. An entity is a concept of data or an object, and an entity set is a collection of similar entities. Attributes define the properties of these entities.

This is the ERD for the smart farming system:

Entities: Nodemcu, Soil Moister Sensor, LM35, LCD, Relay1, Relay2, Database.

The cardinality: Many to many

Most importantly, the Database table have 3 attributes that represents the data stored which are: The soil moister (result1), the temperature (result2), and the time & date (result3). For this later, we can select, modify or add the data.

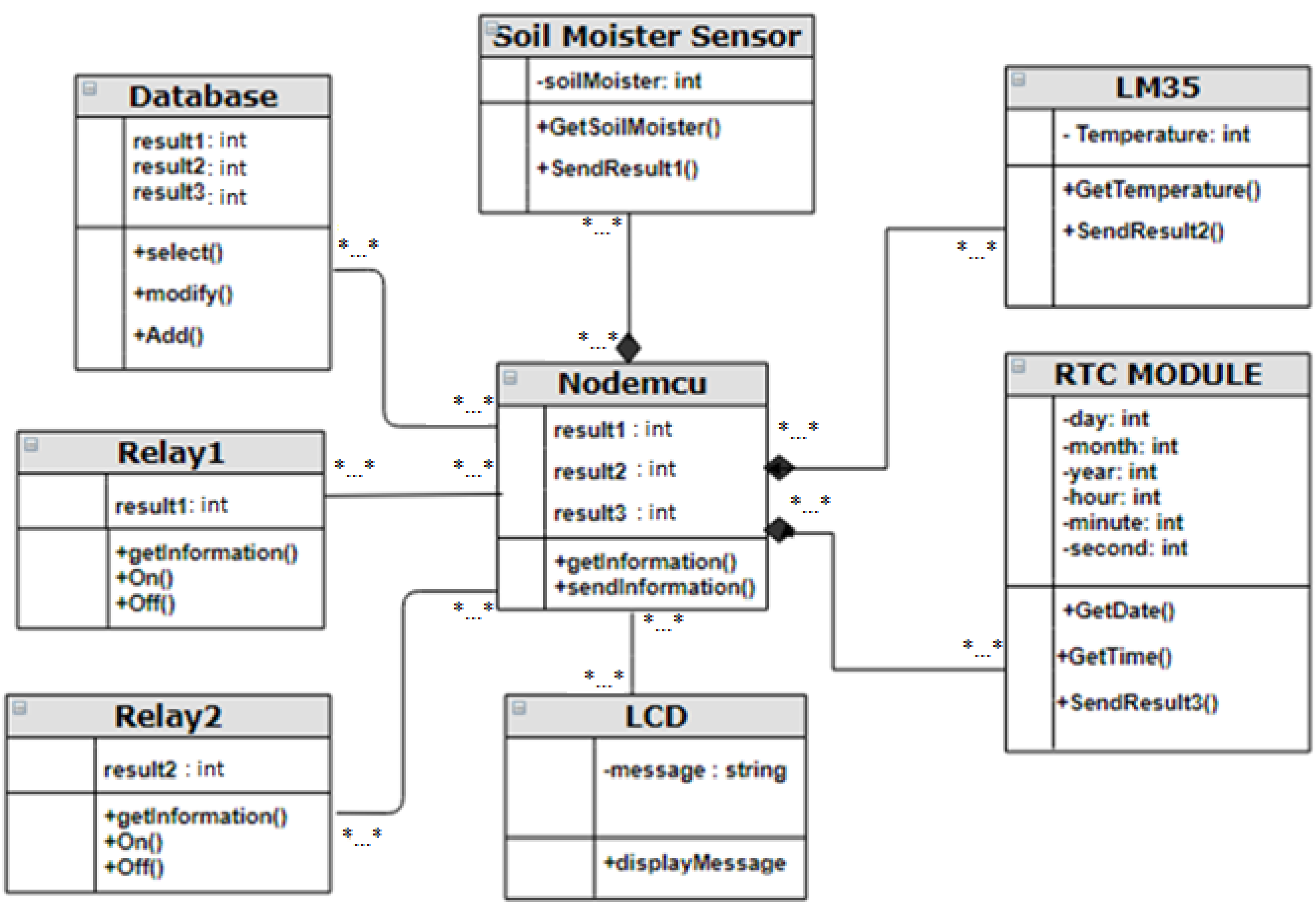


Figure 21. ERD Diagram

Normalization:

Normalization is a database design technique. It organizes tables in order to reduce useless data and dependency of data, by dividing larger tables into smaller ones and links them using relationships to ensure logical storage of data.

1NF (first normal form):

All rows have column with only one value, and the values are atomic:

Temperature	Soil Moister	Time & Date	DAY	Month	Year	Hour	Minute	Sec

2NF (second normal form):

No non-prime attribute depends on a subset of the primary key. Therefore, there is no partial dependency in data

Time & Date	DAY	Month	Year	Hour	Minute	Sec

Temperature	Soil Moister	Time & Date

3NF (third normal form):

If a transitive dependency exists, we remove the transitively dependent attribute from the relation by placing the attribute in a new relation along with a copy of the determinant. Therefore, there is no transitive dependency.

Temperature	Time & Date (primary key)

Soil Moister	Time & Date (primary key)

Time & Date (primary key)	DAY	Month	Year	Hour	Minute	Sec

Database Implementation:

Smart Farming System Database is implemented using Firebase, because it is a real time database.

I used Nodemcu Firebase. I created a new project in my firebase account. To store the data in firebase, I went through the following process:

- I downloaded new libraries in the Arduino IDE : (FirebaseArduino.h), (ArduinoJson.h).
- For user authentication to Firebase, I used the secret code (from firebase account) , and the firebase host (https://arduino-d7791.firebaseio.com/), I defined them in my code then call them in the setup for authentication.
- I created the variables I need in the firebase project, then in the code I can set the values of those (similar) variables to be instantly stored.

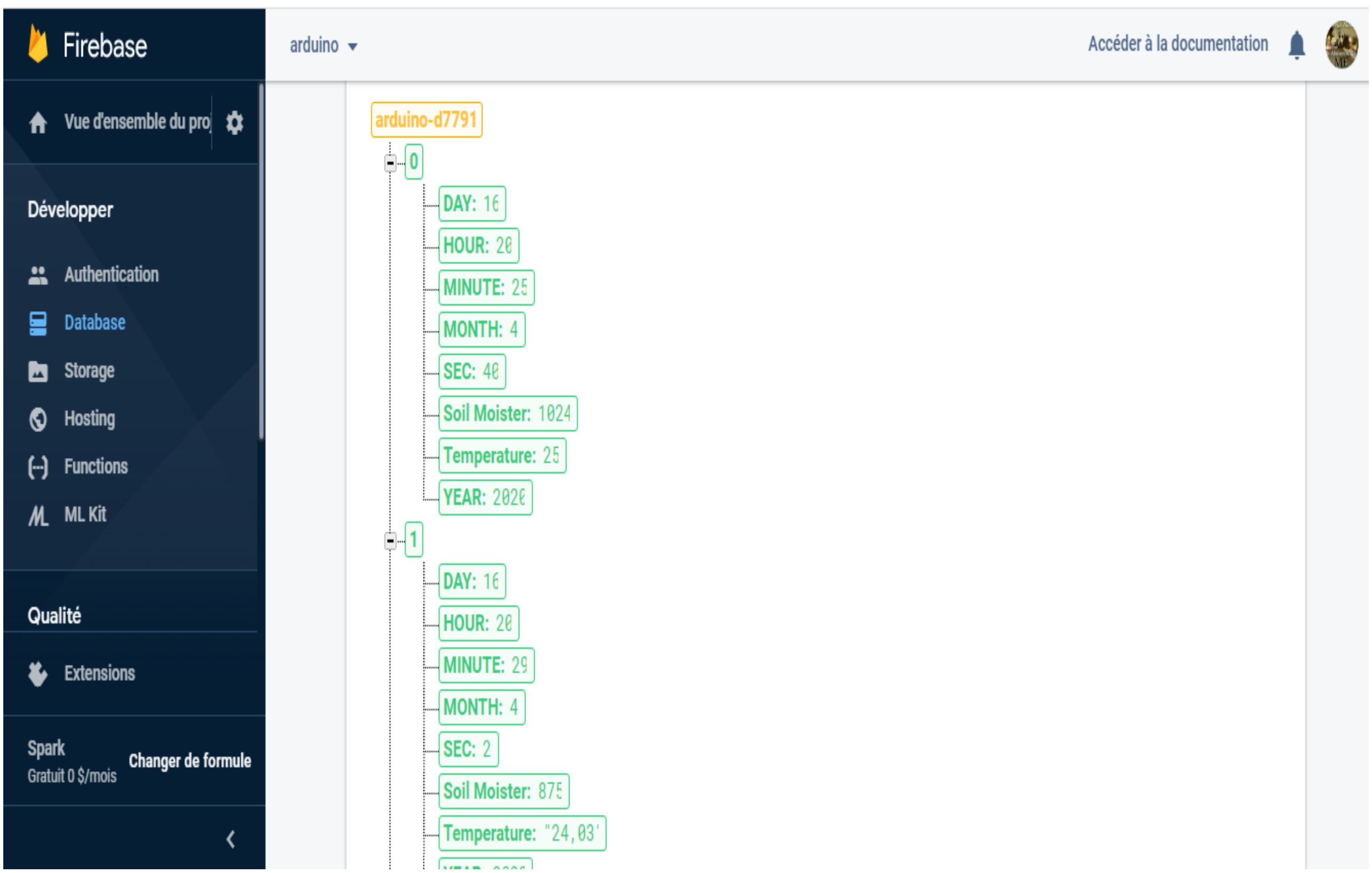


Figure 20. Firebase

Chapter IX: Algorithms and Flowcharts:

Algorithms:

Libraries used and the defined pins:

```
#include <SoftwareSerial.h>
#include <LM35.h>

#include <LiquidCrystal_I2C.h> //
LiquidCrystal_I2C lcd(0x27,16,2);
#include <FirebaseArduino.h>
#include <ArduinoJson.h>
#include <ESP8266HTTPClient.h>
```

Library for serial display

Library for temperature sensor

Library for LCD screen

Library for Firebase

Library for Firebase (Json)

Library for Nodemcu

```
//initializations & definitions
LiquidCrystal_I2C lcd(0x27,16,2);
#include <virtuabotixRTC.h>
#define D0 16
#define D1 5
#define D2 4
#define D3 0
//#define D4 2
#define D5 14
#define D6 12
#define D7 13
#define D8 15
```

Library for clock module

Initializations of Wi-Fi details, Firebase authentication and other pins:

```
//definitions and initialization
#define FIREBASE_HOST "https://arduino-d7791.firebaseio.com/"
#define FIREBASE_AUTH "[redacted]"
String myString;
SoftwareSerial NodeMCU(D1,D2);
virtuabotixRTC myRTC(D0,D6,D5);
SoftwareSerial NodeMCU(D1,D2);
const char* ssid = "Abderrahmane";  //Use your WiFi ID
const char* password = "[redacted]"; //Use your WiFi Password
int analogPin = A0; //Pin used for the measurement on the voltage divider
int vanne = 13; //Pin linked to the transistor to controll the first relay
int vanne2 = 15; //Pin linked to the transistor to controll the second relay
int raw = A0;  // variable to store the raw input value
int Vanne3 = 0;
bool van;
```

Few functions related to the relay & the start of the setup function:

```
WiFiServer server(80);

void ouvreVanne() {
  digitalWrite(vanne, HIGH);
}

void fermeVanne() {
  digitalWrite(vanne, LOW);
}

void setup() {
  Serial.begin(115200);
  delay(10);
```

The setup function: Wi-Fi Connection, serial begin, begin LCD, server begin, Firebase begin :

```
//setup

void setup() {
  Serial.begin(115200);
  delay(10);
  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  // Set the ip address of the webserver
  // WiFi.config(WebServerIP, Gatway, Subnet)
  // or comment out the line below and DHCP will be used to obtain an IP address
  // which will be displayed via the serial console
  WiFi.config(IPAddress(192, 168, 1, 221), IPAddress(192, 168, 1, 1), IPAddress(255, 255, 255, 0))
  // connect to WiFi router
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  // Start the server
  server.begin();
  Serial.println("Server started");
  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());


  fermeVanne();
  fermeVanne2();

  lcd.init(); //initialisation of LCD screen
  lcd.cursor_on();
  lcd.blink_on();
  lcd.backlight();
  lcd.setCursor(0,0);

  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);

  myRTC.setDS1302Time(20, 10, 20, 2, 16, 04, 2020); //set date and time

}
```

Begin the loop where all sensors will start sending data and refreshing it with a delay of 1000ms:

```
//the loop

void loop() {
  lcd.init();
  lcd.cursor_on();
  lcd.blink_on();
  lcd.backlight();
  WiFiStrength = WiFi.RSSI(); // get dBm from the ESP8266
  analogValue = analogRead(D1); // read the analog signal
  // convert the analog signal to voltage
  // the ESP2866 A0 reads between 0 and ~3 volts, producing a corresponding value
  // between 0 and 1024. The equation below will convert the value to a voltage value.
  analogVolts = (analogValue * 3.08) / 1024;
  float voltage = analogRead(A0);
  float temperature = (5.0 * voltage * 100.0)/(1024.0*20);
// display temperature and soil moister voltage in lcd screen and in serial
  Serial.print("Temp: ");
  lcd.setCursor(0,0);
  lcd.print("Temp: ");
  Serial.print(temperature);
  lcd.setCursor(5,0);
  lcd.print(temperature);
  Serial.print("c ");
  lcd.setCursor(10,0);
  lcd.print("c ");
   raw = analogRead(analogPin);
  Serial.print("Soil Moister: ");
  lcd.setCursor(0,1);
  lcd.print("Voltage: ");
  Serial.println(raw);
  lcd.setCursor(9,1);
```

```
 //if soil moister voltage is more than 1000 the first relay will be on (the pump will be working)
 if(raw > 1000) {
   ouvreVanne();
   delay(3000);
   fermeVanne();
   van = 1;
 } else {
   van = 0;
 }
 Serial.print("vanne: ");
 Serial.println(van);
 lcd.init();
 lcd.cursor_on();
 lcd.blink_on();
 lcd.backlight();

//if the temperature reaches 25 or above the second van will be on (the fan will be working)
  if(temperature >= 25) {
   ouvreVanne2();
   delay(3000);
   fermeVanne2();
   van2 = 1;
 } else {
   van2 = 0;
 }

lcd.setCursor(0,0);
lcd.print(myRTC.dayofmonth);
lcd.print("/");
lcd.print(myRTC.month);
lcd.print("/");
lcd.print(myRTC.year);
lcd.setCursor(0,1);
lcd.print(myRTC.hours);
lcd.print(":");
lcd.print(myRTC.minutes);
lcd.print(":");
lcd.print(myRTC.seconds);


// SEND DATA TO FIREBASE

Firebase.setString("Temperature",temperature);
Firebase.setString("soil moister",raw);

Firebase.setString("DAY",myRTC.dayofmonths);
Firebase.setString("MONTH",myRTC.month);
Firebase.setString("YEAR",myRTC.year);
Firebase.setString("HOUR",myRTC.hours);
Firebase.setString("MINUTE",myRTC.minutes);
Firebase.setString("SEC",myRTC.seconds);
delay(1000); // slows amount of data sent via serial
```

The html code for the web server:

```
//******************************************* WEB PAGE HTML CODE *************************************************************
client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println("");
  client.println("<!DOCTYPE HTML>");

client.println("<html>");
  client.println(" <head>");

  //****************** Soil moister chart ************************************

  client.println("<meta http-equiv=\"refresh\" content=\"60\">");
  client.println(" <script type=\"text/javascript\" src=\"https://www.gstatic.com/charts/loader.js\"></script>");
  client.println("  <script type=\"text/javascript\">");
  client.println("    google.charts.load('current', {'packages':['gauge']});");
  client.println("    google.charts.setOnLoadCallback(drawChart);");
  client.println("   function drawChart() {");

  client.println("      var data = google.visualization.arrayToDataTable([ ");
  client.println("        ['Label', 'Value'], ");
  client.print("        ['Moisture',  ");
  client.print(raw);
  client.println(" ], ");
  client.println("       ]); ");
```

Head of web server:

```
  // setup the google chart options here
client.println("    var options = {");
  client.println("      width: 400, height: 120,");
  client.println("      redFrom: 0, redTo: 25,");
  client.println("      yellowFrom: 25, yellowTo: 75,");
  client.println("      greenFrom: 75, greenTo: 100,");
  client.println("       minorTicks: 5");
  client.println("    };");

  client.println("   var chart = new google.visualization.Gauge(document.getElementById('chart_div'));");

  client.println("  chart.draw(data, options);");

  client.println("  setInterval(function() {");
  client.print("  data.setValue(0, 1, ");
  client.print(raw);
  client.println("    );");
  client.println("    chart.draw(data, options);");
  client.println("    }, 13000);");
```

Styling of web server (CSS):

```
//********************************************** STYLING ********************************

 client.println( "<title> Project's Webpage </title>");
client.println(" <style>" );
client.println( " body { ");
  client.println( "background-color: lightblue;");
 client.println( "");
client.println( "h1{");
client.println("border: 1px solid black;");
client.println("text-align: center;");
client.println("text-decoration-color: green;");
client.println("}");
client.println("p{");
client.println("text-align: center;");
client.println("position: right;");
client.println("font-family: verdana;");
 client.println(" font-size: 20px;");
```

Body of web server code:

```
// **************************************** BODY ********************************************

client.println("<body>");
client.println("<h1> IOT Based Smart Agriculture Monitoring System </h1>");

client.println("<div class=<\"container\">");

//*********************************** TEMPERATURE *******************************************

client.println("<font color=\"red\"> <h2> Temperature Sensor </h2> </font>");
//client.println("<img src=\"thermometer.Webp\" width=\"200\" height=\"400\">");
 // below is the google chart html
 client.println("<div id=\"chart_T\" style=\"width: 300px; height: 120px;\"></div>");

client.println("<table><tr><td>");
  client.print("Temperature: ");
  client.println(temperature);
  client.println(" C ");
client.println("<p> </br> Date & Time : </p>");
client.print(myRTC.dayofmonth);
```

```
client.println("<table><tr><td>");
  client.print("Temperature: ");
  client.println(temperature);
  client.println(" C ");
client.println("<p> </br> Date & Time : </p>");
client.print(myRTC.dayofmonth);
client.print("/");
client.print(myRTC.month);
client.print("/");
client.print(myRTC.year);
client.print("  ");
client.print(myRTC.hours);
client.print(":");
client.print(myRTC.minutes);
client.print(":");
client.println(myRTC.seconds);
//client.println("<img src=\"HUMIDITE.jpg\" width=\"200\" height=\"200\">");


//********************************** SOIL MOISTER ********************************************

client.println("<font color=\"red\"> <h2> Moisture Sensor </h2> </font>");
 // below is the google chart html
  client.println("<div id=\"chart_div\" style=\"width: 300px; height: 120px;\"></div>");

 client.println("<table><tr><td>");
  client.print("WiFi Signal Strength: ");
  client.println(WiFiStrength);
  client.println("dBm<br>");
  client.print("<br>Analog Raw: ");
  client.println(analogValue);
  client.print("<br>Analog Volts: ");
  client.println(analogVolts);
   client.print("<br>Voltage: ");
 client.println(raw);
 client.println("<p> </br> Date & Time : </p>");
  client.print(myRTC.dayofmonth);
  client.print("/");
  client.print(myRTC.month);
  client.print("/");
```

```
//************************************* VANNE ************************************************
client.println("<font color=\"red\"> <h2> Vanne </h2> </font>");



client.println("<table><tr><td>");

  if (van == 1){
  client.println("<img src=\"https://loopinghpf.net/sevenwonders/resources/img/icons/green_light.png\" width=\"100\" height=\"100\">");
  client.print("<br>The watering system is: ");
  client.println("ON");
  } else {
    client.println("<img src=\"https://findicons.com/files/icons/1933/symbols/128/red_light.png\" width=\"100\" height=\"100\">");
    client.print("<br>The watering system is: ");
    client.println("OFF");
  }

client.println("<br><a href=\"/REFRESH\"\"><button>Refresh</button></a>");

client.println("</div>");
client.println("<script src=\"hello.js\"></script>");
client.println("</body>");
client.println("</html>");
delay(1);

  Serial.println("Client disonnected");
  Serial.println("");


}
```

The web server will be refreshed with a delay of 10ms.

FLOWCHART:

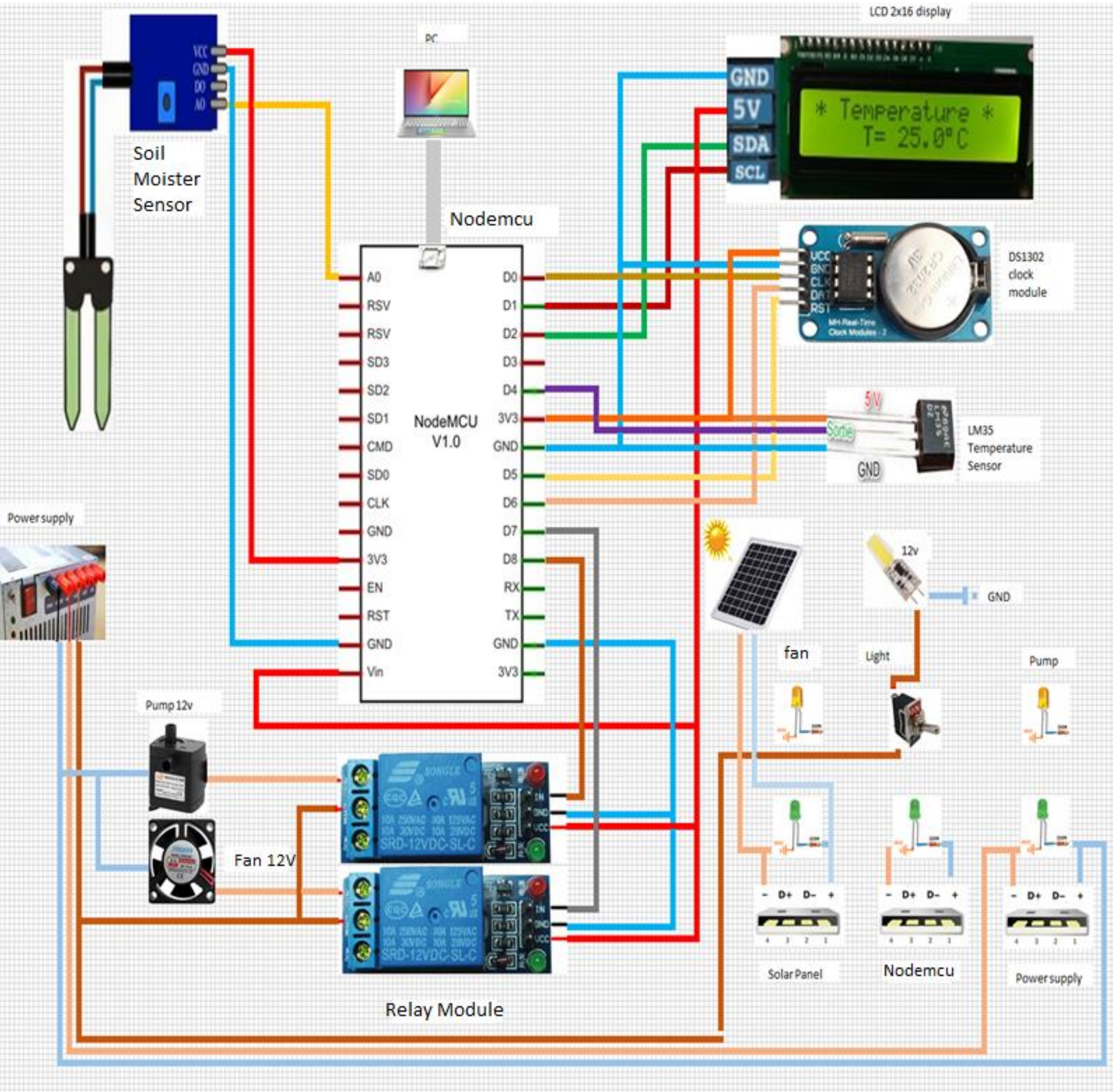


Figure 22. Flowchart

Chapter X: Technologies Enablers:

	Arduino IDE used to implement the algorithm and run it to program the Nodemcu
	Firebase used for data storage
	Online tool used to design diagrams
	Excel used to draw the Flowchart
	EasyEda used to draw the circuit

Chapter XI: CONCLUSIONS:

Conclusion:

This capstone project gave me the chance to learn new technologies and work with new tools, this was a real proof that AUI has taught us to be long-life learners and to master self-learning before teaching us other class materials. Of course, this project is a combination of what I learned from all my computer science classes, the programming languages, the database systems and the engineering process that is important in any engineering project, all together with what I learned from other disciplines and also by myself about IoT and the use of Arduino helped me to build an embedded system.

In general, the project was successful and worked properly and succeeded in delivering the prototype on due time. I am proud and happy for this achievement especially that this is my first real big theoretical and practical project. It enabled me to get concrete results and to realize that I can indeed build products that would be beneficial in real life and that I can customized upon demand as future projects.

Future work:

For future work there are many features to develop or add to the prototype:

- Adding a Mobile Application for Android to make it easier to access data remotely and where data can be stored easily.
- Develop an IOS version of the previous application
- Add other Arduino components or increasing sensors for more features, and to fetch more data that can be collected.
- Add an artificial intelligent system to predict the production of goods.
- Integrating a GPS module to enhance this Agriculture IoT Technology to full-fledged Agriculture Precision ready product.

References:

[1] CircuitsToday. *Arduino and Soil Moisture Sensor -Interfacing Tutorial.* Retreived from http://www.circuitstoday.com/arduino-soil-moisture-sensor

[2] Dfrobot. *I2C 16x2 Arduino LCD Display Module.* Retreived from https://www.dfrobot.com/product-135.html

[3] Lastminuteengineers. *Insight Into ESP8266 NodeMCU Features & Using It With Arduino IDE.* Retreived from https://lastminuteengineers.com/esp8266-nodemcu-arduino-tutorial/

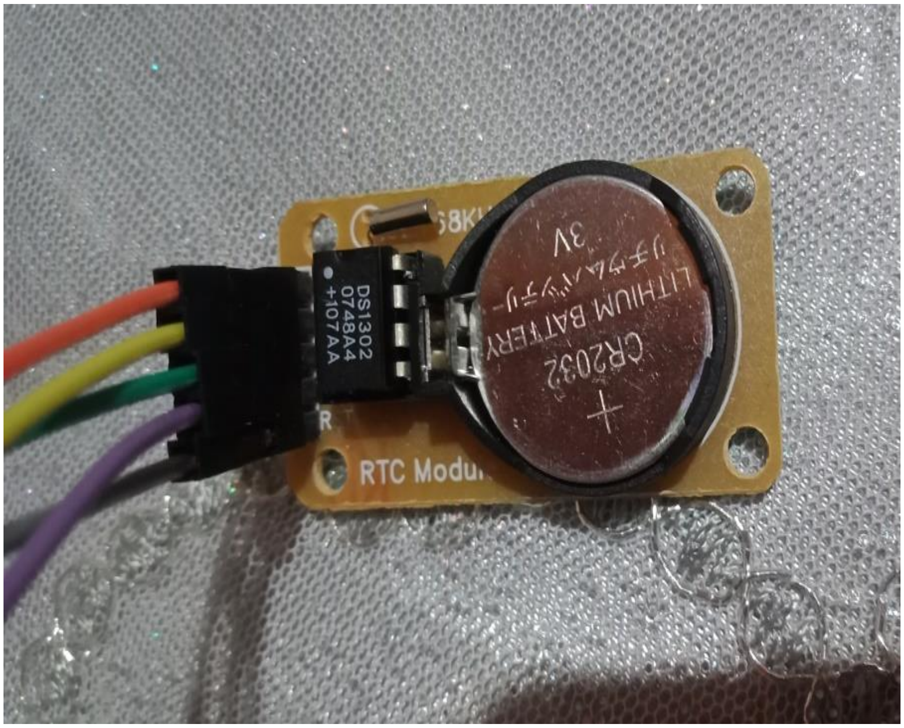
[4] Meola, A. (Jan 24, 2020). *Smart Farming in 2020: How IoT sensors are creating a more efficient precision agriculture industry*. Retreived from https://www.businessinsider.com/smart-farming-iot-agriculture

[5] Team, D. (JUN 17, 2019). *Is IoT the Future of Agriculture.* Retreived from https://www.digiteum.com/iot-agriculture

[6] TexasInstriments.(2017). *LM35 Precision Centigrade Temperature Sensors.* Retreived from http://www.ti.com/lit/ds/symlink/lm35.pdf

Appendix :

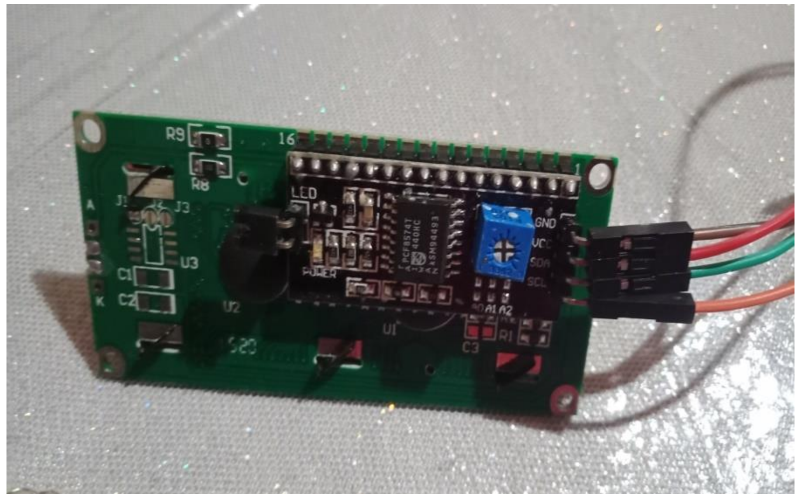
The figures of the components used in this project:



Figures 7. RTC Module [1]

Figure 8. Relay

Pictures of the LCD used in my prototype.



Figures 9. LCD

Female/Male Color Ribbon Flat Cable Jumper Dupont 40-wire 20cm 2.54mm

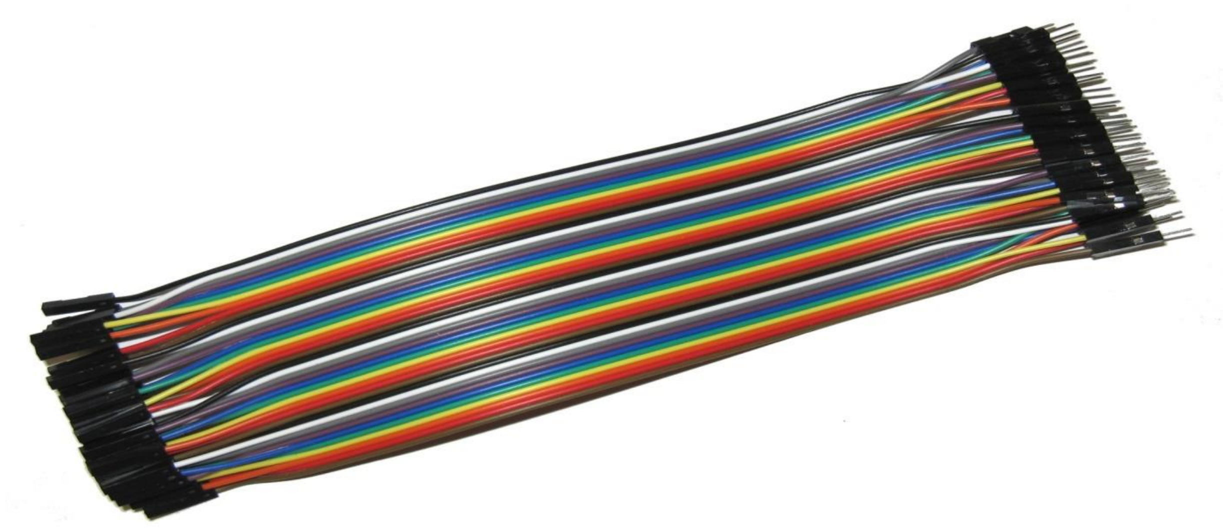


Figure 10

150 mm wire with male connectors on the ends.

Figure 11

20cm Female to Female Circuit Board Cable Colorful Wire

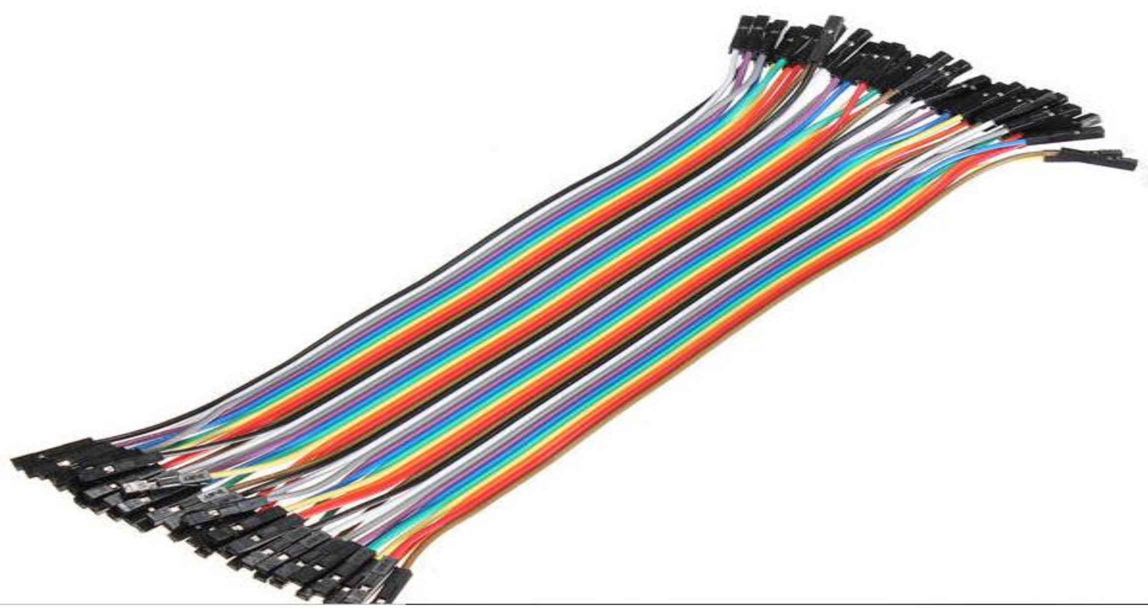


Figure 12

Mini USB cable of Nodemcu

Solar panel

Figure 14. Solar panel

- The Electric box :

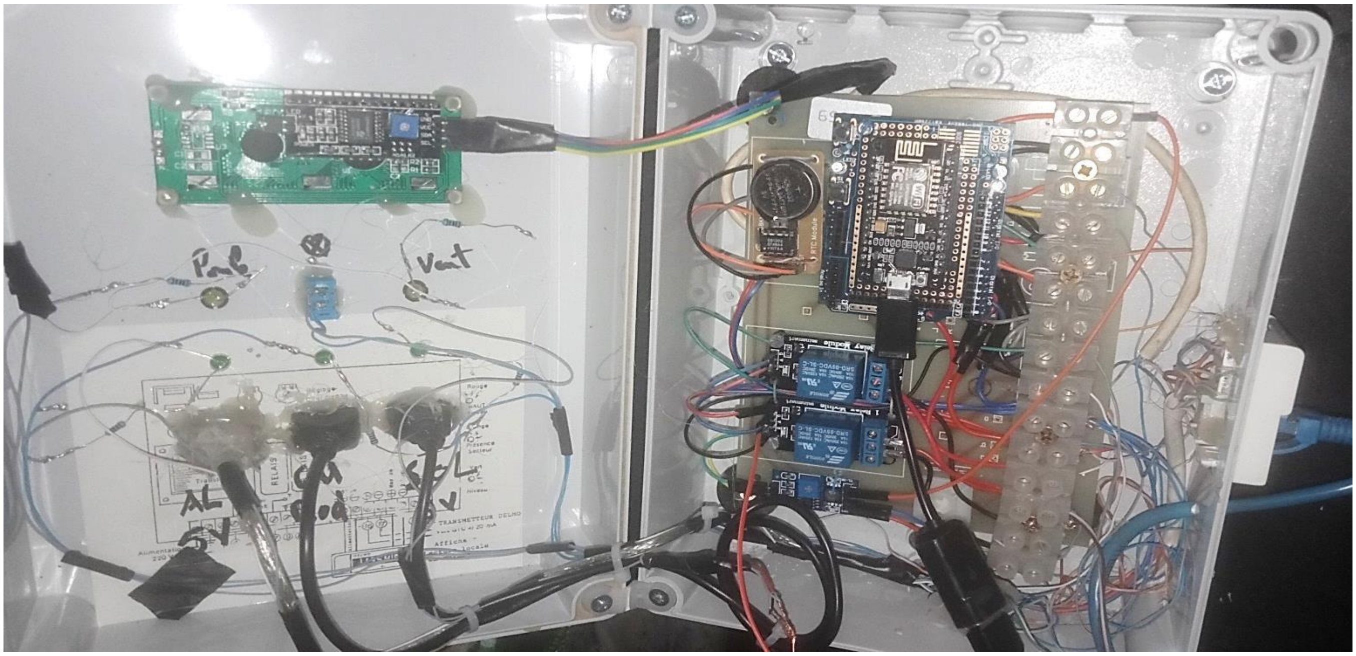


Figure 16. Electric Box

- The Green house :

Figure 17. Greenhouse