Smart Irrigation System Project Report

Project Title: *Smart Irrigation System*
Trainer: *WatchIt Group Rwanda*
By: *Julius Aries Kanneh, Jr*
Date: *September 21, 2021*

1.0 Project Overview/Introduction

As a project to enhance my understanding of the IoT studied at WatchIT Group Rwanda, I have designed a Smart Irrigation System using knowledge learned throughout my studies. Smart Irrigation is an IoT based Agriculture system that uses the Capacitive Soil Moisture Sensor to detect the soil moisture (amount of water in the soil) in order to trigger the pump for irrigation automatically.

2.0 Problem our proposed solution is solving

The issue of manually performing irrigation without even knowing if the soil actually needs water or not has been a challenge for local farmers. Sometimes, they use instance and experience to guess when their crop needs water or not. Some farmers set specific time during the day they perform irrigation on their farm which they at times are either late or too early. They are never going to be exact with the time as we are humans and humans are subject to error.
The irrigation is also done by farmers manually which requires a lot of work and is time consuming. In this report, I am providing a recommended solutions to these problems using IoT technology.

3.0 Technology and Components Used

Because this is an IoT based project, I have used the Menya IoT development kit with other components installed on it. Here is a list of physical components used during the execution of the project along with their usages and roles in the project.

3.1 Hardware Components Used Chart

Components	Quantity	Usage	Role in this project
Menya IoT kit	1	Holds all other components used.	It was used to hold all other components used in this project. It is like a mother board or bread board
NodeMCU ESP8266	1	Board for IoT development. It can be used as Client and WebServer.	I use this component to serve as Web server and web client. We use it to connect to our local WiFi to send sensor data read from the NodeMCU ESP8266

			to my local hosting service(XAMPP), phpMyAdmin database and CayenneMyDevice.
DHT11 Sensor	1	Used to calculate temperature and relative humidity.	I am using this sensor to calculate the temperature and relative humidity of the farm which is then send to the database and Cayenne for visualization on the web-based dashboard.
Capacitive Soil Moisture V1.2 Sensor	1	Used to Calculate soil moisture (quantity of water in the soil)	I am using this sensor to measure the soil moisture of my farm which will be used to trigger the irrigation pump on or off based on a the condition specified.
LED	1	Light Emitting Diode, emits light when current flows through it.	I am using LED in our project to prototype the irrigation pump. If the LED is on, it means the pump is on otherwise, the pump is off.
Wires	11	Used for connections.	I am using the wires to connect all our components used in this project with the NodeMCU board for effective working.

3.2 Software Components Used

IoT uses the effective combination of both software and hardware components to produce an effective working product, therefore, I combine both software and hardware to implement this project. Below is a list of the software used in the implementation of the Smart Irrigation System.

Software Component	**Usage**	**Role in this project**
Arduino Integrated Environment (IDE)	Used for writing Arduino sketches with file extension ".ino"	I used this IDE to develop the sketch which reads data from all of our sensors and control actuators like the LED(Irrigation Pump). The sketch also includes the web server code to interact with the internet to send

		data to my local server(XAMP) and Cayenne for data visualization and trigger implementation.
Visual Studio Code IDE	Used for multipurpose coding.	I have used this IDE to develop the web-based application and dashboard using HTML, CSS, JAVASCRIPT & PHP.
XAMP Local Host	Used to host web applications on my local workstation.	I have used XAMP local host to host the web-based dashboard application and the database which holds the sensor values using phpMyAdmin which is embedded in it.
Web Browser	For serving information over the internet.	I have used it during the project development for testing purposes. Chrome, Firefox and Edge are the three I used widely.
Cayenne my Device	Also used by IoT developers to visualize data in real time from anywhere at anytime and to create triggers.	I have used Cayenne to specifically visualize the soil moisture value and to trigger email notification and SMS notification if the soil moisture reaches 700.

4.0 Technical Details of the Project

At this point, I am going into a little detail of the technologies used for the success of this project. I will explain how I have implemented the entire solution breaking it down into modules.

Module1: Developing the Arduino Sketch

In this module, I develop the sketch which is used to read the sensor values, control the irrigation pump(LED) and setup a web-server and client to send (POST) and receive (GET) data using httpClient.

Module1:1 : Adding Libraries needed for the project

Firstly, I added the following libraries into the Arduino sketch:

- <ESP8266HTTPClient.h>
- <CayenneMQTTESP8266.h>
- <DHT.h>

Module1:2 : Connecting to WiFi and Cayenn

Firstly, we make the ESP8266 to connect to our local WiFi and then setup our sketch to interact with Cayenne specifying the username, password and client id from my Cayenne account when I login.

Module1:2 : Reading Sensors values

Capacitive Soil Moisture V1.2 sensors: This sensor is used to calculate the soil moisture. To do this, I firstly connected the **VCC** of the sensor to a 5V pin on the Arduino uno board since the maximum output voltage of the ESP8266 is 3.3V. I than connected the analog output to A0 on the ESP8266 board. I then connected the ground pin of the sensor to the ground pin of the board. See the schematic below:

Now to determine the soil moisture of the soil, I used the following sketch:

```
  moisture_value = analogRead(sensor_pin);

//  Displaying soil moisture value to Serial Monitor
  Serial.print("Moisture Value: ");
  Serial.println(moisture_value);
```

DHT11 Sensor: We use this sensor to measure the relative humidity and temperature of the farm. To do that, we firstly include the define the dht_pin and type and than create an instance of the dht by passing the *dht_pin* and *dht_type* as parameters. Now, to read the temperature and humidity, we create variables to hold them and use the functions as to read the temperature and humidity as shown the snapshot below:

```
t = dht.readTemperature();
h = dht.readHumidity();

Serial.print("Temperature: ");
Serial.print(t);
Serial.println("C");
Serial.print("Humidity: ");
Serial.print(h);
Serial.println("%");
```

Module 1:3 : Sending Data to Cayenne Dashboard

In order to visualize our data anywhere at anytime, I have used Cayenne my Device. I send the soil moisture value, temperature and relative humidity value to my Cayenne live dashboard. To achieve this, I have connected to my device on Cayenne using the following the username and password of the device on cayenne and my client id. See screenshot below with the code:

```
void setup() {
  Serial.begin(9600);
  Cayenne.begin(username, password, clientID, ssid, wifi_password);
  dht.begin();
  pinMode(A0, OUTPUT);
  pinMode(actuator, OUTPUT);
}
```

I than push values read from the sensors to my cayenne device I have connected to above. See snapshot of code below:

```
//  Pushing sensors values to Cayene dashboard
//-------------------------------------------------------------//
  Cayenne.virtualWrite(0, moisture_value);
  Cayenne.virtualWrite(1, t);
  Cayenne.virtualWrite(2, h);
```

Module 1:4 : Sending Data to phpMyAdmin Database of XAMPP

During this module, I am sending the sensor data to phpMyAdmin database of the XAMP every 3 to 4 seconds. To do this, I created a WifiClient and HTTPClient instance which I used to make http POST request.
See screenshot below:

```
// Pushing sensors values to MySql database
//-------------------------------------------------------------//
  HTTPClient http;

  String send_t, send_h, s_moisture, postData;
  send_t = String(t);
  send_h = String(h);
  s_moisture = String(moisture_value);

  postData = "send_t=" + send_t + "&send_h=" + send_h + "&s_moisture=" + s_moisture;

  http.begin(client, "http://192.168.1.219/Smart_Agri/db_push.php");
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");
  int httpCode = http.POST(postData);

  if(httpCode == 200){
    Serial.println("Connected to local server successfully");
    String playload = http.getString();
    Serial.println(playload + "\n");
  }else{
    Serial.println(httpCode);
    Serial.println("Failed to upload valules");
    http.end();
  }
```

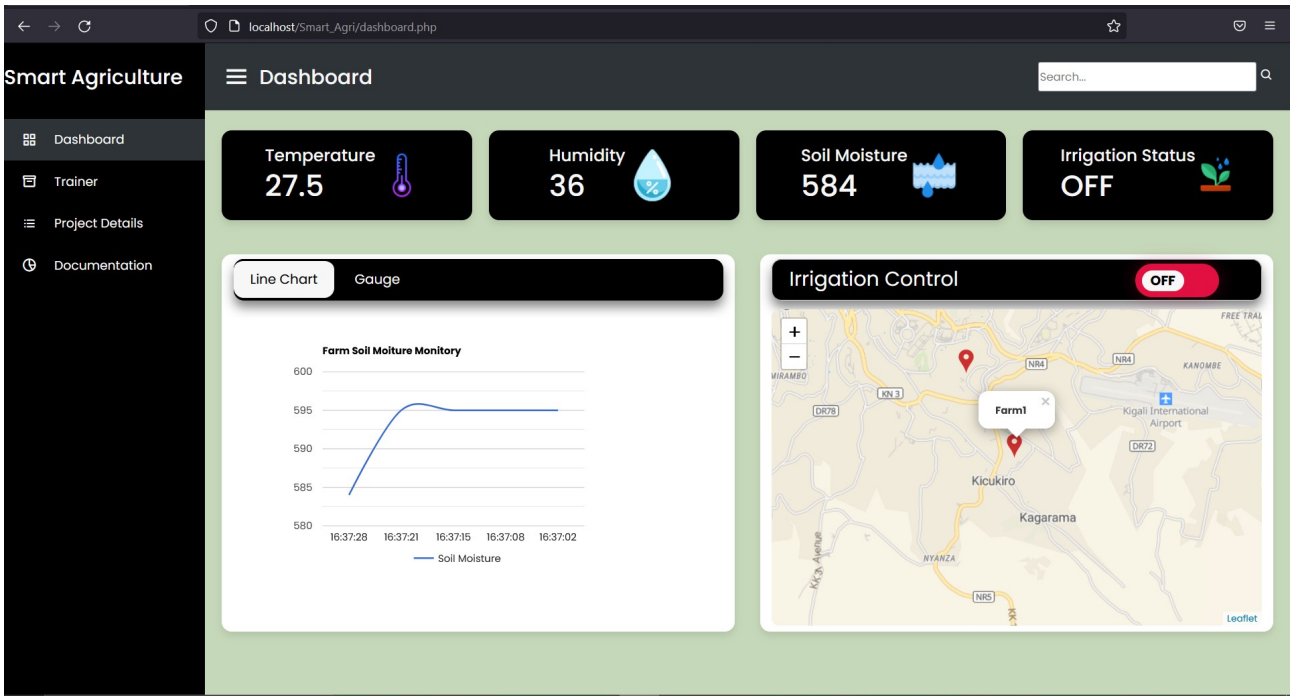
Module 1:5 : Receiving Data from phpMyAdmin Database of XAMPP

Because I want to trigger on and off the irrigation pomp (LED) based on the irrigation status from the dashboard. I used first send alter the status of irrigation in the database as either 1 or 0 based on the state of the button on the web dashboard I designed. I than read the value from the database into my arduino sketch so that I can be able to power on or off the pomp(LED). See the code snapshot below:

```
//http to get data from the server(database)
  HTTPClient httpGet;

  httpGet.begin(client1, "http://192.168.1.219/Smart_Agri/getData.php");
  httpGet.addHeader("Content-Type", "application/x-www-form-urlencoded");

//  getData = "ID=" + String(id);
//  int httpCodeGet = httpGet.POST(getData);

  int httpCode1 = httpGet.GET();
  if(httpCode1 == 200){
    Serial.println("Connected successfully");
    String payloadGet = httpGet.getString();
    Serial.print("Irrigaion status from db: ");
    Serial.println(payloadGet);

//Turn on irrigation if value returned from db is 1 and soil_moisture value is greater than or equal 700.
//Turn off irrigation if value returned from the db is 0 and soil moisture value is less than or equal to 580.

    if(payloadGet == "1"){
        digitalWrite(actuator, HIGH);
      }else{
        digitalWrite(actuator, LOW);
      }

  }else{
    Serial.println("Not Connected");
  }
```

Module2: Designing the Web-Application Dashboard

I have built a web application using *html, css, javascript, and php* to give the owner of the farm remote control to the irrigation of the farm and visualization of the farm data(temperature, relative humidity and soil moisture).
I have used html to design the pages' structure, css and javascript to style and add interactivity to the web-app and finally php to handle the back-end functionalities like sending data to database and retrieving record from database. See screenshot of the dashboard below:



Module3: Cayenne Dashboard and Triggers

Data pushed to the cayenne dashboard can be visualize using the cayenne mobile app or the cayenne web application.
I also created triggers to send automatic email and sms notification to my email and local number if the soil moisture value reaches 700.

Conclusion

Thanks to WatchIT Group Rwanda for the training and guidance which has given me the knowledge to implement this solution.