# SMART AGRICULTURE SYSTEM

-Gayathri Madduri
-Poojitha Mittapalli
-Srividya Srigiri
-Sana

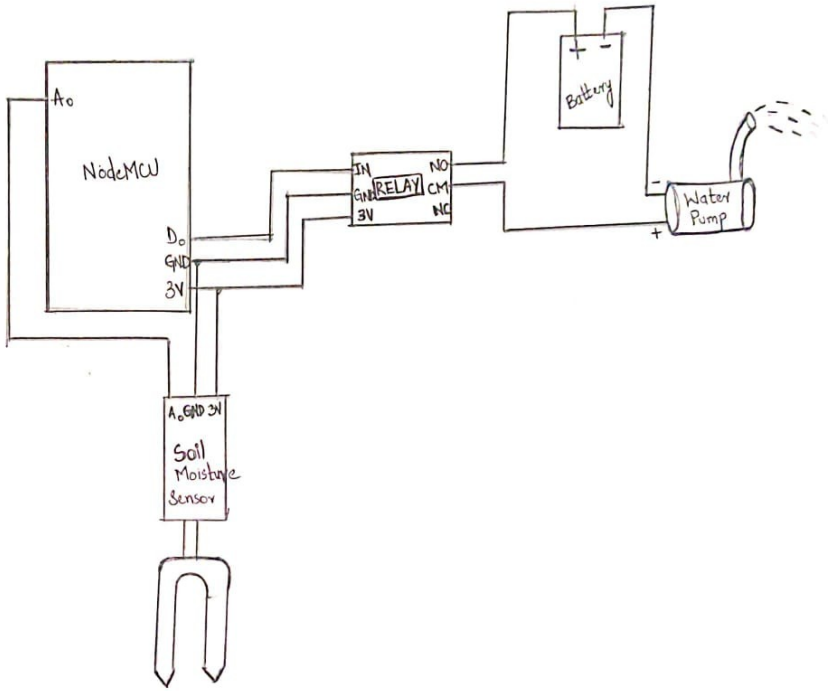
# Project Idea

- This Project is an attempt to improve the efficiency of Irrigation System.
- Monitoring the moisture present in the soil
- If the soil moisture is less than the required then the water pump is switched on for irrigation.
- Once the moisture becomes adequate the water pump  will be switched off.
- EVAPOTRANSPIRATION (ET0)

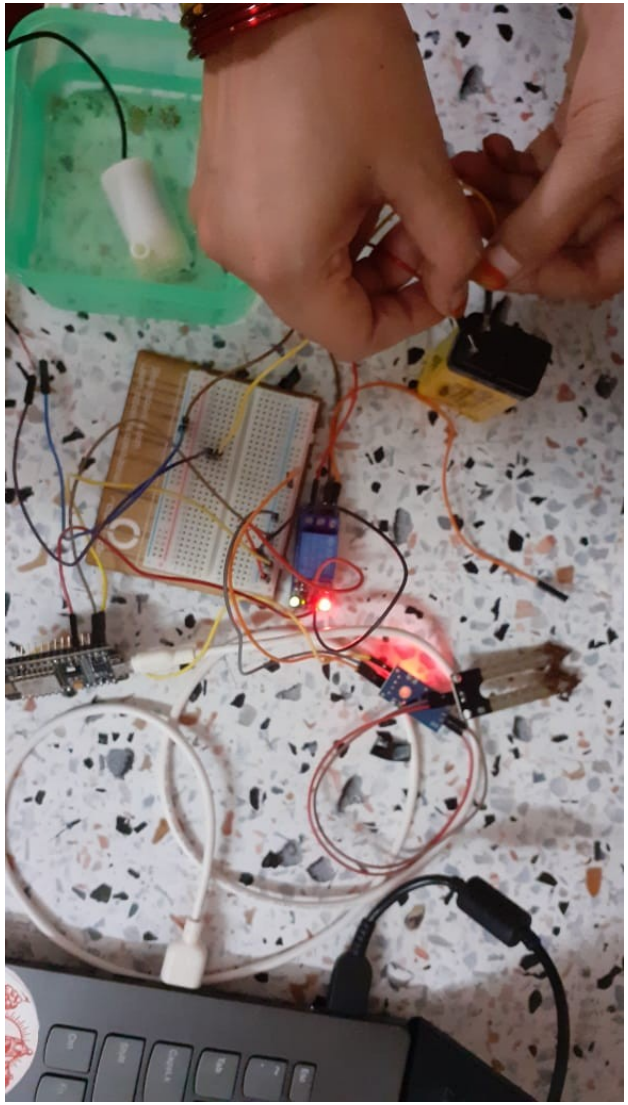
# Components

- NodeMCU ESP8266
- Soil Moisture Sensor Module
- Water Pump Module
- Relay Module
- Connecting wires
- 9 V Battery
- Breadboard

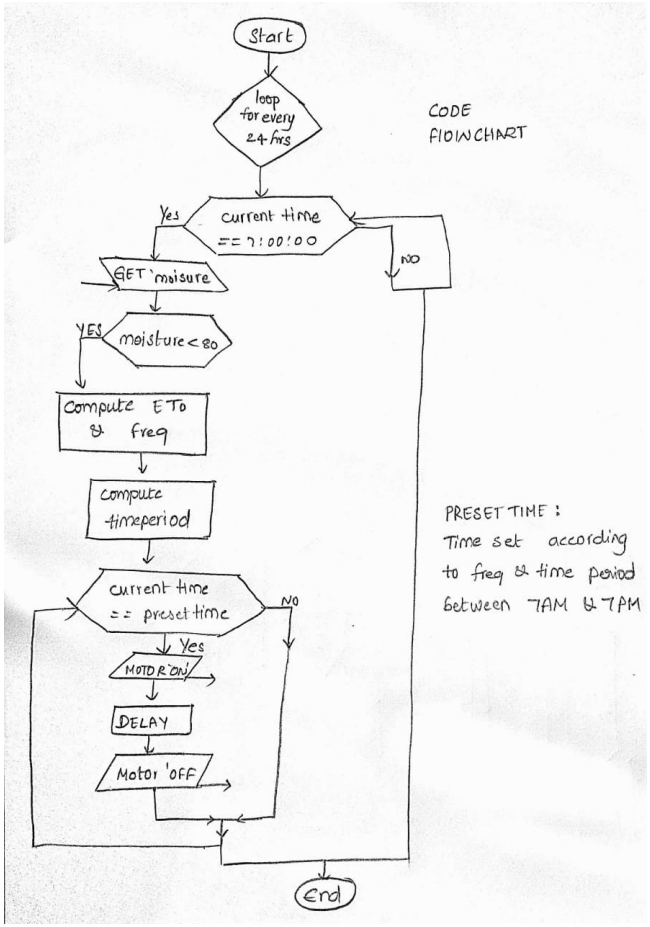
# Circuit



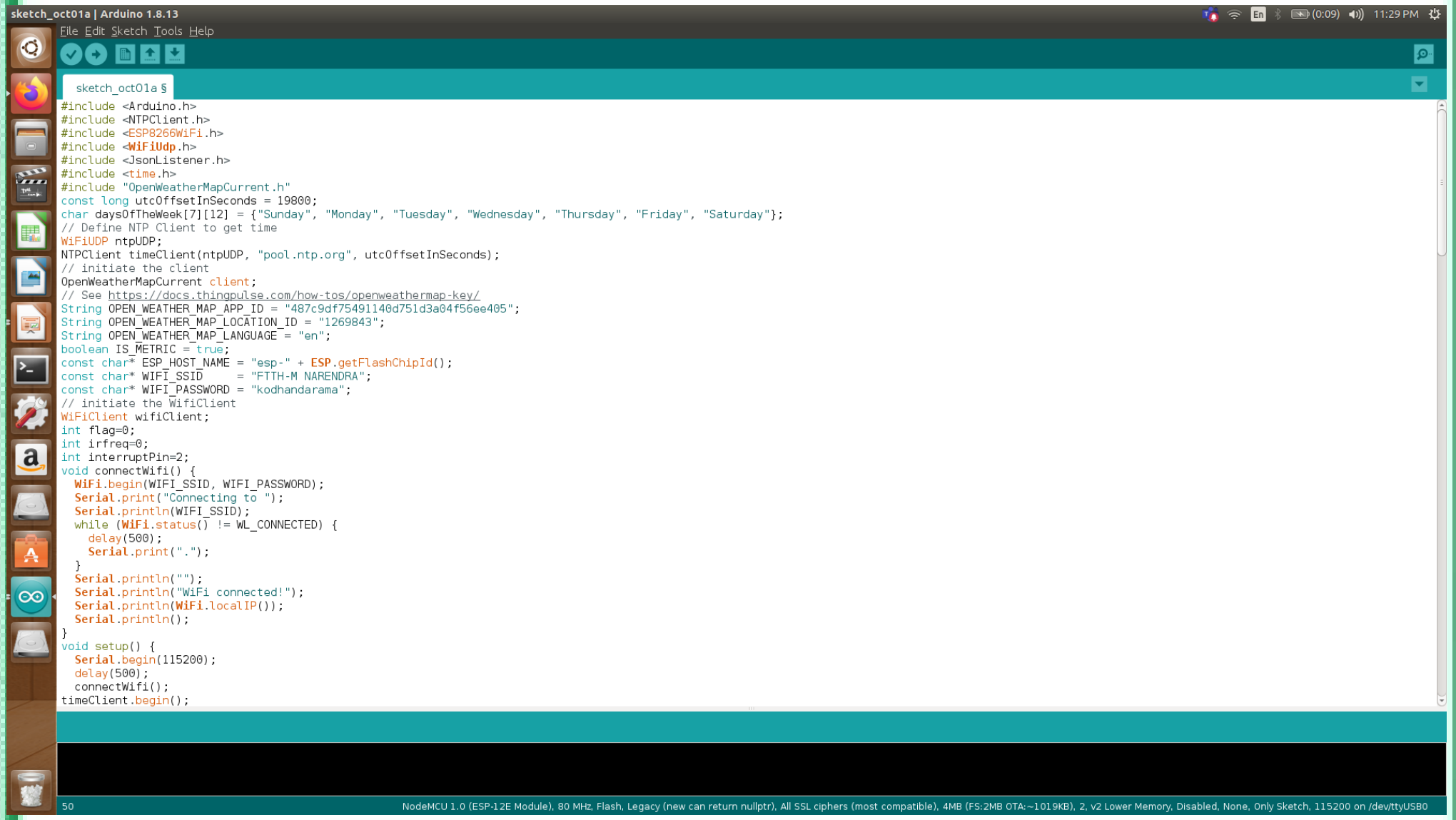


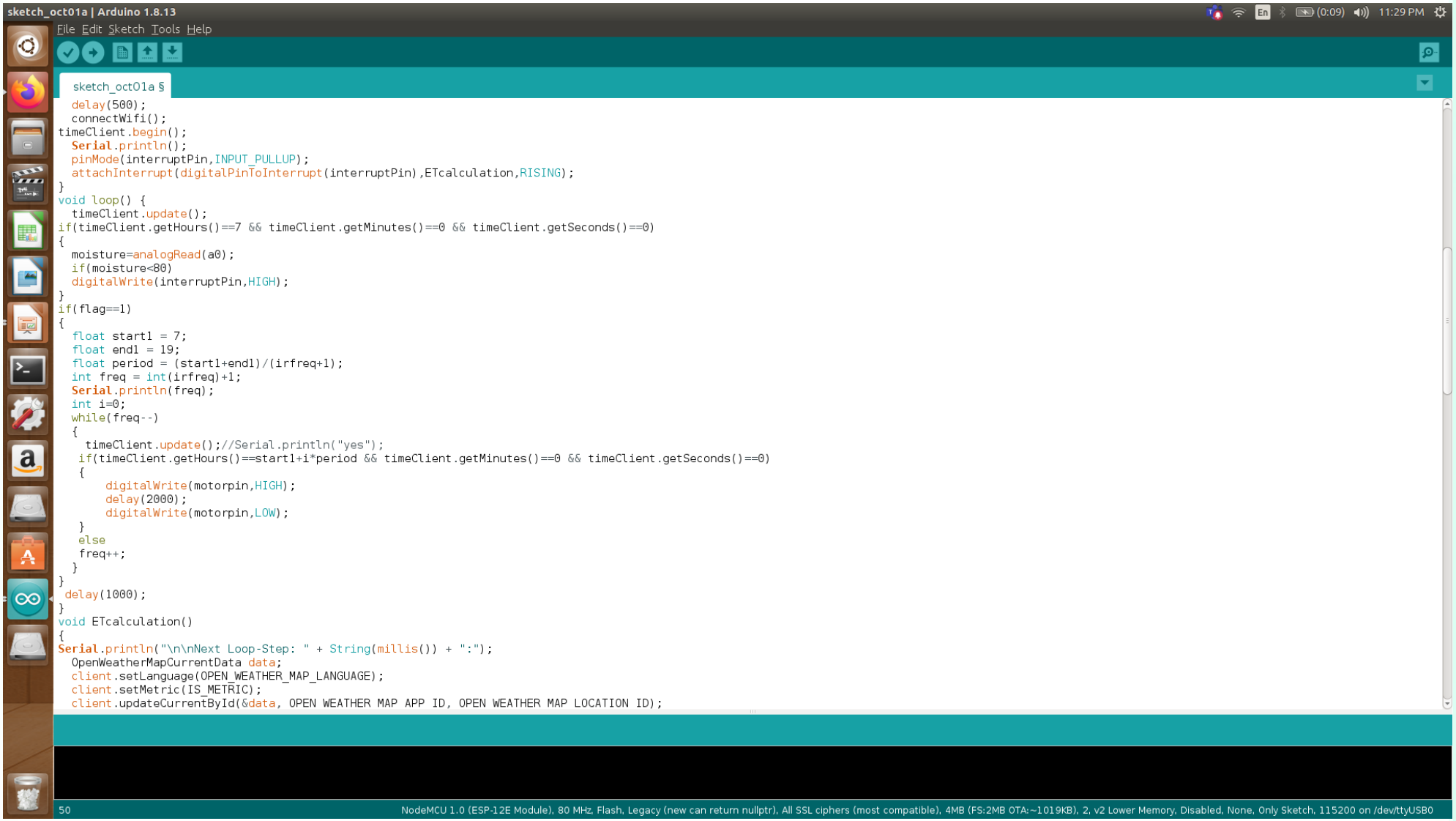



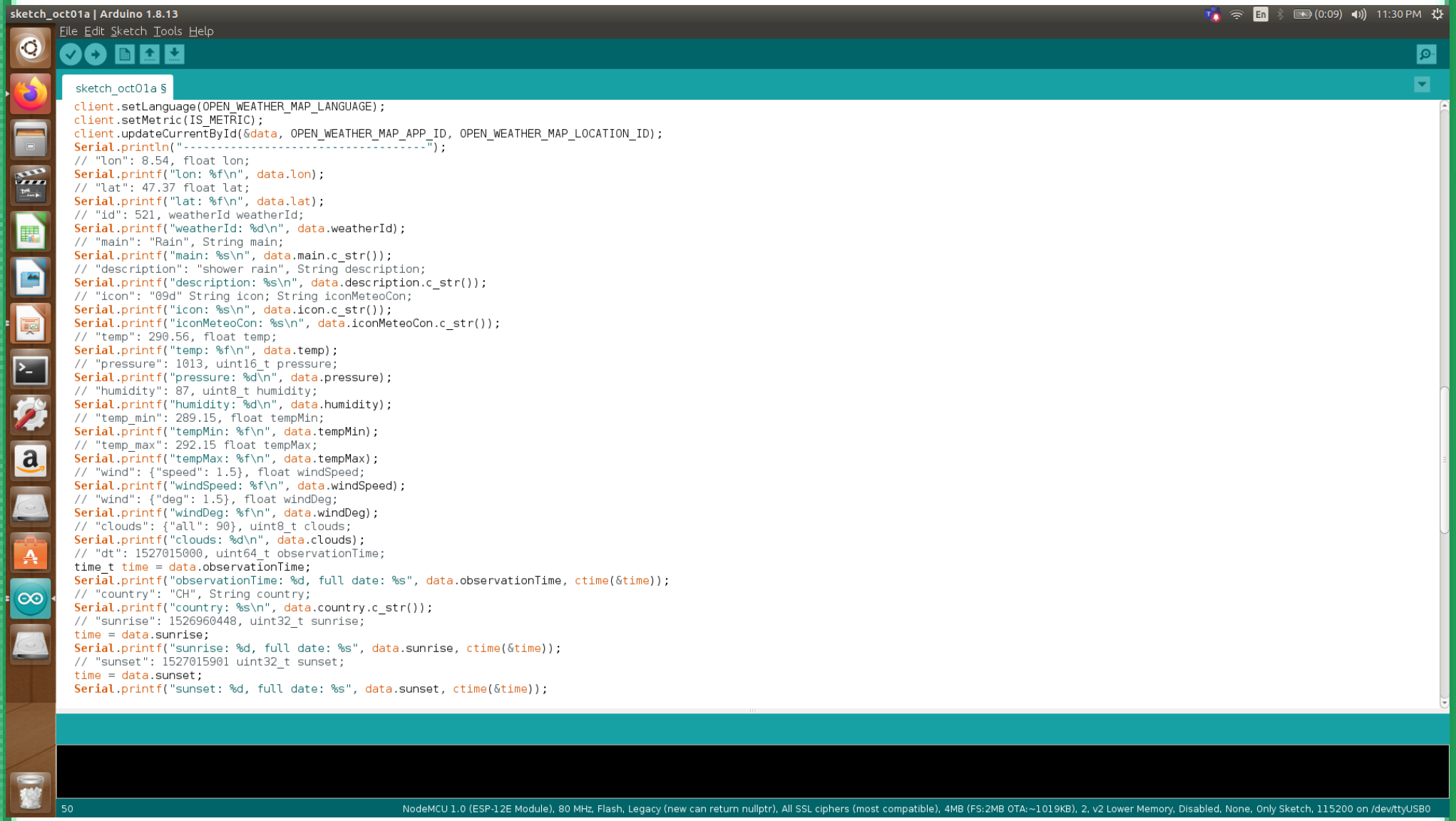
# Flow chart


Start
loop for every 24 hrs
CODE FLOWCHART
Yes
current time == 7:00:00
NO
GET 'moisure
YES
moisture < 80
Compute ETo & freq
compute timeperiod
current time == preset time
NO
Yes
MOTOR 'ON'
DELAY
Motor 'OFF'
End
PRESET TIME :
Time set according to freq & time period between 7AM & 7PM

# CODE


sketch_oct01a | Arduino 1.8.13
File Edit Sketch Tools Help
sketch_oct01a §
#include <Arduino.h>
#include <NTPClient.h>
#include <ESP8266WiFi.h>
#include <WiFiUdp.h>
#include <JsonListener.h>
#include <time.h>
#include "OpenWeatherMapCurrent.h"
const long utcOffsetInSeconds = 19800;
char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"};
// Define NTP Client to get time
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org", utcOffsetInSeconds);
// initiate the client
OpenWeatherMapCurrent client;
// See https://docs.thingpulse.com/how-tos/openweathermap-key/
String OPEN_WEATHER_MAP_APP_ID = "487c9df75491140d751d3a04f56ee405";
String OPEN_WEATHER_MAP_LOCATION_ID = "1269843";
String OPEN_WEATHER_MAP_LANGUAGE = "en";
boolean IS_METRIC = true;
const char* ESP_HOST_NAME = "esp-" + ESP.getFlashChipId();
const char* WIFI_SSID = "FTTH-M NARENDRA";
const char* WIFI_PASSWORD = "kodhandarama";
// initiate the WifiClient
WiFiClient wifiClient;
int flag=0;
int irfreq=0;
int interruptPin=2;
void connectWifi() {
WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
Serial.print("Connecting to ");
Serial.println(WIFI_SSID);
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected!");
Serial.println(WiFi.localIP());
Serial.println();
}
void setup() {
Serial.begin(115200);
delay(500);
connectWifi();
timeClient.begin();
50
NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Legacy (new can return nullptr), All SSL ciphers (most compatible), 4MB (FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on /dev/ttyUSB0


sketch_oct01a | Arduino 1.8.13
File Edit Sketch Tools Help
sketch_oct01a §
delay(500);
connectWifi();
timeClient.begin();
Serial.println();
pinMode(interruptPin,INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(interruptPin),ETcalculation,RISING);
}
void loop() {
timeClient.update();
if(timeClient.getHours()==7 && timeClient.getMinutes()==0 && timeClient.getSeconds()==0)
{
moisture=analogRead(a0);
if(moisture<80)
digitalWrite(interruptPin,HIGH);
}
if(flag==1)
{
float start1 = 7;
float end1 = 19;
float period = (start1+end1)/(irfreq+1);
int freq = int(irfreq)+1;
Serial.println(freq);
int i=0;
while(freq--)
{
timeClient.update();//Serial.println("yes");
if(timeClient.getHours()==start1+i*period && timeClient.getMinutes()==0 && timeClient.getSeconds()==0)
{
digitalWrite(motorpin,HIGH);
delay(2000);
digitalWrite(motorpin,LOW);
}
else
freq++;
}
}
delay(1000);
}
void ETcalculation()
{
Serial.println("\n\nNext Loop-Step: " + String(millis()) + ":");
OpenWeatherMapCurrentData data;
client.setLanguage(OPEN_WEATHER_MAP_LANGUAGE);
client.setMetric(IS_METRIC);
client.updateCurrentById(&data, OPEN WEATHER MAP APP ID, OPEN WEATHER MAP LOCATION ID);
50
NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Legacy (new can return nullptr), All SSL ciphers (most compatible), 4MB (FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on /dev/ttyUSB0


sketch_oct01a | Arduino 1.8.13
File Edit Sketch Tools Help
sketch_oct01a §
client.setLanguage(OPEN_WEATHER_MAP_LANGUAGE);
client.setMetric(IS_METRIC);
client.updateCurrentById(&data, OPEN_WEATHER_MAP_APP_ID, OPEN_WEATHER_MAP_LOCATION_ID);
Serial.println("------------------------------------");
// "lon": 8.54, float lon;
Serial.printf("lon: %f\n", data.lon);
// "lat": 47.37 float lat;
Serial.printf("lat: %f\n", data.lat);
// "id": 521, weatherId weatherId;
Serial.printf("weatherId: %d\n", data.weatherId);
// "main": "Rain", String main;
Serial.printf("main: %s\n", data.main.c_str());
// "description": "shower rain", String description;
Serial.printf("description: %s\n", data.description.c_str());
// "icon": "09d" String icon; String iconMeteoCon;
Serial.printf("icon: %s\n", data.icon.c_str());
Serial.printf("iconMeteoCon: %s\n", data.iconMeteoCon.c_str());
// "temp": 290.56, float temp;
Serial.printf("temp: %f\n", data.temp);
// "pressure": 1013, uint16_t pressure;
Serial.printf("pressure: %d\n", data.pressure);
// "humidity": 87, uint8_t humidity;
Serial.printf("humidity: %d\n", data.humidity);
// "temp_min": 289.15, float tempMin;
Serial.printf("tempMin: %f\n", data.tempMin);
// "temp_max": 292.15 float tempMax;
Serial.printf("tempMax: %f\n", data.tempMax);
// "wind": {"speed": 1.5}, float windSpeed;
Serial.printf("windSpeed: %f\n", data.windSpeed);
// "wind": {"deg": 1.5}, float windDeg;
Serial.printf("windDeg: %f\n", data.windDeg);
// "clouds": {"all": 90}, uint8_t clouds;
Serial.printf("clouds: %d\n", data.clouds);
// "dt": 1527015000, uint64_t observationTime;
time_t time = data.observationTime;
Serial.printf("observationTime: %d, full date: %s", data.observationTime, ctime(&time));
// "country": "CH", String country;
Serial.printf("country: %s\n", data.country.c_str());
// "sunrise": 1526960448, uint32_t sunrise;
time = data.sunrise;
Serial.printf("sunrise: %d, full date: %s", data.sunrise, ctime(&time));
// "sunset": 1527015901 uint32_t sunset;
time = data.sunset;
Serial.printf("sunset: %d, full date: %s", data.sunset, ctime(&time));
50
NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Legacy (new can return nullptr), All SSL ciphers (most compatible), 4MB (FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on /dev/ttyUSB0


sketch_oct01a | Arduino 1.8.13
File Edit Sketch Tools Help
sketch_oct01a §
time = data.sunset;
Serial.printf("sunset: %d, full date: %s", data.sunset, ctime(&time));
// "name": "Zurich", String cityName;
Serial.printf("cityName: %s\n", data.cityName.c_str());
float Tmin=data.tempMin;
float Tmax=data.tempMax;
float Tmean = (Tmin+Tmax)/2;
float z=542;
float u2 = data.windSpeed;
float delta = (4098 * 0.6108*exp((17.27*Tmean)/(Tmean+237.3)))/pow((Tmean+237),2);
float p = 101.3*pow(((293-0.0065*z)/293),(5.26));
float g = 0.000665*p;
float DT = delta/(delta+g*(1+0.34*u2));
float PT = g/(delta+g*(1+0.34*u2));
float tt = (900/(Tmean+273))*u2;
float eTmax = 0.6108*exp((17.27*Tmax)/(Tmax+237.3));
float eTmin = 0.6108*exp((17.27*Tmin)/(Tmin+237.3));
float es = (eTmax+eTmin)/2;
float rhmean=data.humidity;
float ea = (rhmean/100)*es;
int day=274;
float pi=3.14;
float dr = 1+0.033*cos(2*pi*day/365);
float solard = 0.409*sin((2*pi*day/365)-1.39);
//step13
float degree=data.lat;
float rad = (pi/180)*degree;
float Ws = acos(-tan(rad)*tan(solard));
float Ra = (24*60/pi)(0.0820*dr((Ws*sin(rad)*sin(solard))+(cos(rad)*cos(solard)*sin(Ws))));
//step16
float Rs=5.44;
float Rso = (0.75+2*pow(10,-5)*z)*Ra;
float Rns = 0.77*Rs;
float sigma=4.903*pow(10,-9);
float Rnl = sigma*(pow((Tmax + 273.16 ),4)+pow((Tmin+273.16),4)/2)(0.34-0.14*sqrt(ea))(1.35*(Rs/Rso) -0.35);
float Rn = Rns - Rnl;
float Rng = 0.408*Rn;
float ETrad = DT*Rng;
float ETwind = PT*tt*(es-ea);
float ETo = ETwind+ETrad;
Serial.println("ET0 is ");
Serial.println(ETo);
50
NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Legacy (new can return nullptr), All SSL ciphers (most compatible), 4MB (FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on /dev/ttyUSB0

sketch_oct01a | Arduino 1.8.13

File Edit Sketch Tools Help

sketch_oct01a §

```
  float Tmax=data.tempMax;
  float Tmean = (Tmin+Tmax)/2;
  float z=542;
  float u2 = data.windSpeed;
  float delta = (4098 * 0.6108*exp((17.27*Tmean)/(Tmean+237.3)))/pow((Tmean+237),2);
  float p = 101.3*pow(((293-0.0065*z)/293),(5.26));
  float g = 0.000665*p;
  float DT = delta/(delta+g*(1+0.34*u2));
  float PT = g/(delta+g*(1+0.34*u2));
  float tt = (900/(Tmean+273))*u2;
  float eTmax = 0.6108*exp((17.27*Tmax)/(Tmax+237.3));
  float eTmin = 0.6108*exp((17.27*Tmin)/(Tmin+237.3));
  float es = (eTmax+eTmin)/2;
  float rhmean=data.humidity;
  float ea = (rhmean/100)*es;
  int day=274;
  float pi=3.14;
  float dr = 1+0.033*cos(2*pi*day/365);
  float solard = 0.409*sin((2*pi*day/365)-1.39);
  //step13
  float degree=data.lat;
  float rad = (pi/180)*degree;
  float Ws = acos(-tan(rad)*tan(solard));
  float Ra = (24*60/pi)(0.0820*dr((Ws*sin(rad)*sin(solard))+(cos(rad)*cos(solard)*sin(Ws))));
//step16
  float Rs=5.44;
  float Rso = (0.75+2*pow(10,-5)*z)*Ra;
  float Rns = 0.77*Rs;
  float sigma=4.903*pow(10,-9);
  float Rnl = sigma*(pow((Tmax + 273.16 ),4)+pow((Tmin+273.16),4)/2)(0.34-0.14*sqrt(ea))(1.35*(Rs/Rso) -0.35);
  float Rn = Rns - Rnl;
  float Rng = 0.408*Rn;
  float ETrad = DT*Rng;
  float ETwind = PT*tt*(es-ea);
  float ETo = ETwind+ETrad;
Serial.println("ET0 is ");
Serial.println(ETo);
float depth=2.00;
irfreq=depth/ETo;
Serial.print("The number of times it should provide water per day is ");
Serial.println(irfreq);
  Serial.println();
  Serial.println("-------------------------------------------------/\n");
  flag=1;
}
```

50

NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Legacy (new can return nullptr), All SSL ciphers (most compatible), 4MB (FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on /dev/ttyUSB0

# Motivation

- Over irrigation will result in poor crops
- water is a scarce resource.
- We must try to conserve it especially with the continuous depletion of water table that we see around
- Agriculture comprises much of the water consumption world wide. Hence keeping the amount of water optimum can be a crucial step to contribute towards the environment