In the name of GOD

Title:

Automate irrigation and online reading data

Supervisor:

Dr Vahid Ghasemzadeh

Dr Ali Mosahebfard

Creator:

Farzin Sharifzade Javidi

Fall 2022

Contents

[Abstract 3](#_Toc116619719)

[Hardware 3](#_Toc116619720)

[Wiring circuit 7](#_Toc116619721)

[Arduino Code 8](#_Toc116619722)

[Save the Data on local system 10](#_Toc116619723)

[Django 10](#_Toc116619724)

[Online with internet 14](#_Toc116619725)

[Online website 17](#_Toc116619726)

[Conclusion 18](#_Toc116619727)

[References 18](#_Toc116619728)

Abstract

In this project, we want to automate irrigation using Arduino and create a website to show the moisture level of the soil.to do so we need to create a control system with Feedback using an Arduino microcontroller. For the website, We can use Django and python or use any CMS we want. in this project, we use both ways And also talk about the API for sending data to the desired web service whether it is CMS or Django.

HardwareIn the beginning, let’s talk about the Hardware of the project. In this project, we use Arduino UNO However the same project can be done with any other kind of micro such as Arduino Nano or Raspberry pi or Node MCU, etc. Moisture sensor to sense the humidity of the Soil,5 Volt relay to connect the Arduino to our pomp, And an adaptor for our pomp, It’s better to use an adaptor to power the pomp the alternative way is to connect to pomp directly to the Arduino but In this scenario, You may burn Your Arduino so It’s better to get a separated power source.

* Arduino uno
* Moisture sensor
* Single-Channel 5V Relay
* 12V DC pomp
* 12V Adaptor
* Jumper Wire

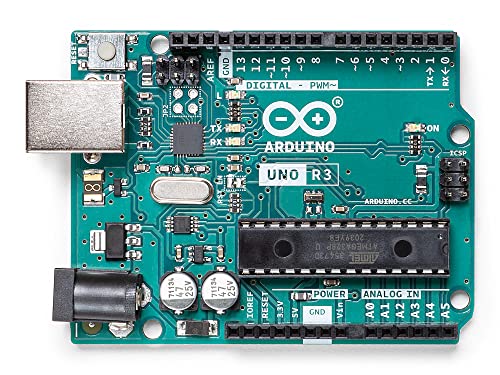
In the following, I will explain each component

Arduino UNO:

The Arduino Uno is an [open-source](https://en.wikipedia.org/wiki/Open-source) [microcontroller board](https://en.wikipedia.org/wiki/Microcontroller_board) based on the [Microchip](https://en.wikipedia.org/wiki/Microchip_Technology) [ATmega328P](https://en.wikipedia.org/wiki/ATmega328P) microcontroller developed by [Arduino. cc](https://en.wikipedia.org/wiki/Arduino) and was initially released in 2010. The board is equipped with sets of digital and analog [input/output](https://en.wikipedia.org/wiki/Input/output) (I/O) pins that may be interfaced with various [expansion boards](https://en.wikipedia.org/wiki/Expansion_board) (shields) and other circuits. The board has 14 digital I/O pins (six capable of [PWM](https://en.wikipedia.org/wiki/Pulse-width_modulation) output), and 6 analog I/O pins, and is programmable with the [Arduino IDE](https://en.wikipedia.org/wiki/Arduino#Software) (Integrated Development Environment), via a type B [USB cable](https://en.wikipedia.org/wiki/USB_cable). It can be powered by the USB cable or by an external [9-volt battery](https://en.wikipedia.org/wiki/9-volt_battery), though it accepts voltages between 7 and 20 volts. It is similar to the [Arduino Nano](https://en.wikipedia.org/wiki/Arduino_Nano) and Leonardo. The hardware reference design is distributed under a [Creative Commons](https://en.wikipedia.org/wiki/Creative_Commons) Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available.

The word "[no](https://en.wiktionary.org/wiki/uno)" means "one" in [Italian](https://en.wikipedia.org/wiki/Italian_language) and was chosen to mark the initial release of [Arduino Software](https://en.wikipedia.org/wiki/Arduino_Software). The Uno board is the first in a series of USB-based Arduino boards; it and version 1.0 of the Arduino [IDE](https://en.wikipedia.org/wiki/Integrated_development_environment) were the reference versions of Arduino, which have now evolved to newer releases. The ATmega328 on the board comes preprogrammed with a [bootloader](https://en.wikipedia.org/wiki/Bootloader) that allows uploading new code to it without the use of an external hardware programmer.

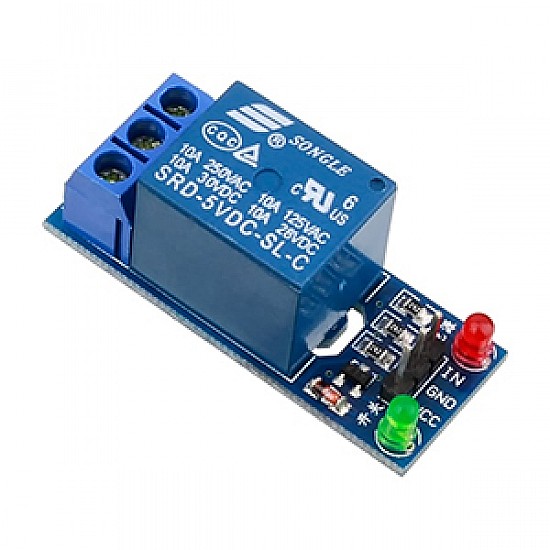
While the Uno communicates using the original STK500 protocol, it differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.



Moisture sensor:

 The soil moisture sensor consists of two probes that are used to measure the volumetric content of water. The two probes allow the current to pass through the soil, which gives the resistance value to measure the moisture value.

Single-Channel 5V Relay:

Relay is an electromechanical device that uses an electric current to open or close the contacts of a switch. The single-channel relay module is much more than just a plain relay, it comprises components that make switching and connection easier and act as indicators to show if the module is powered and if the relay is active or not.

12V DC pomp:

Any kind of pomp can be used. In this project, I’ve used Pride Car Windshield.  
The important Tip in Choosing Pomp is, to find pomp that would work on the 15V DC at max.

12V DC Adaptor:

As I said earlier It’s optional to use an adaptor to supply the power for pomp, It’s better to do so, cause it will prevent over-current draw In our Arduino.

In the following, I will explain how to bypass the adaptor to pomp so that the pomp will gain its Currents from the adaptor and not Arduino.

Jumper wire:

Since we’re working with Arduino We need some jumper wire to connect our modules to Arduino

Wiring the circuit

In the following, I will explain the wiring and schematic of the circuit

* 5V from Power Arduino to VCC of sensor
* GND from Power Arduino to GND of sensor
* A3 from Analog pin to AO pin of sensor (AO stands for Analog Output)
* Vin from Power Arduino to VCC of Relay
* GND from Power Arduino to GND of Relay
* Digital pin 7 from Arduino to IN pin of Relay
* NO port of Relay to Positive pole of Pomp (NO stands for Normally open)
* Com port of Relay to Positive Wire of Adaptor
* Negative Wire of the pomp to Negative of adaptor

Now that the wiring is finished Let’s talk about Arduino Code

Arduino Code

// if the soil is dryer than this number, then start watering

const int dry = 450;

const int pumpPin = 7;

const int soilSensor = A4;

void setup() {

pinMode(pumpPin, OUTPUT);

pinMode(soilSensor, INPUT);

Serial.begin(9600);

digitalWrite(pumpPin, HIGH);

delay(5000);

}

void loop() {

// read current moisture

int moisture = analogRead(soilSensor);

Serial.println(moisture);

delay(5000);

if (moisture >= dry) {

// the soil is too dry, water!

Serial.println("Watering starts now..moisture is " + String(moisture));

digitalWrite(pumpPin, LOW);

// keep watering for 3 sec

delay(3000);

// turn off water

digitalWrite(pumpPin, HIGH);

Serial.println("Done watering.");

} else {

digitalWrite(pumpPin, HIGH);

Serial.println("Moisture is adequate. No watering needed " + String(moisture));

}

}

Now that We write the code let’s explain it a little bit:

At first, we Declare our Variable such as the desired Moisture Level and Arduino Pins we want to work with For Example in this case Digital pin 7 and Analog pin 4

Then in Void setup(), we must determine the output or input of the chosen pin

And finally, in the Void loop() section, we must write the down code

Since in this project we want to observe the moisture level online it’s better to print the moisture level each time we irrigate the soil.

The logic of the code is that when the sensor senses the Moisture lower than the Const int Declare in the first, the Arduino will send a command to pomp and pomp will start working for 3 Sec, that sensor will sense the Moisture again if the Moisture is higher than the declared desired level the irrigation will stop else: it will keep watering till moisture level is equal or higher than the declared variable.

Now if You connect Arduino to the laptop and upload the code into it and put the sensor in the soil and connect the adaptor to the Power outlet you can see that Your micro is working properly also you can see the serial monitor of the Moisture Level.

Save the Data on the local system

As I said earlier one of the goals of this project is to upload the Data online into the website we create. The easy way of doing this is to use Module Name ESP8226 And connect the module to Arduino and it will upload the data automatically to Arduino cloud All you need to do is to sign up in the Arduino Cloud : )

In our scenario, we want to do it the hard way.

First, we will create our website, As I said earlier we can use the CMS or Django python I’ve done both so I explain both here:

First Let’s talk about Django Webservice

Django

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

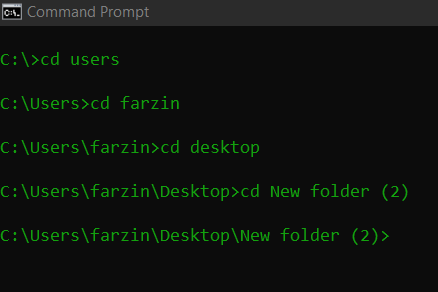
To create our first web page with Django first you must Download Python and install Django. to do that after you install your python you must go to Your command prompt And type in this command:

* “Pip install Django”

After you did that You are almost ready to create your first web page. To do so

You must create a folder and target that folder in your command prompt using the command : “cd”

For example, in my case, I create a folder on my desktop and Named it a New folder (2)

And to target that directory I must go to it using the cd command As did in the below picture:

When you did that (going to the directory), You must type this code in your command prompt:

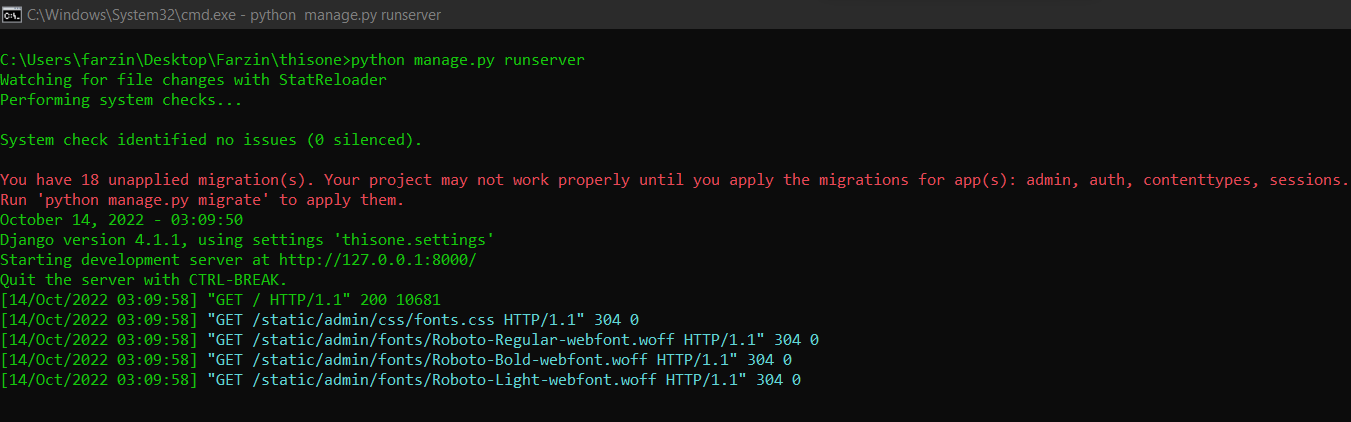
* “django-admin startproject mysite”

Congratulation you’ve created your first python web service

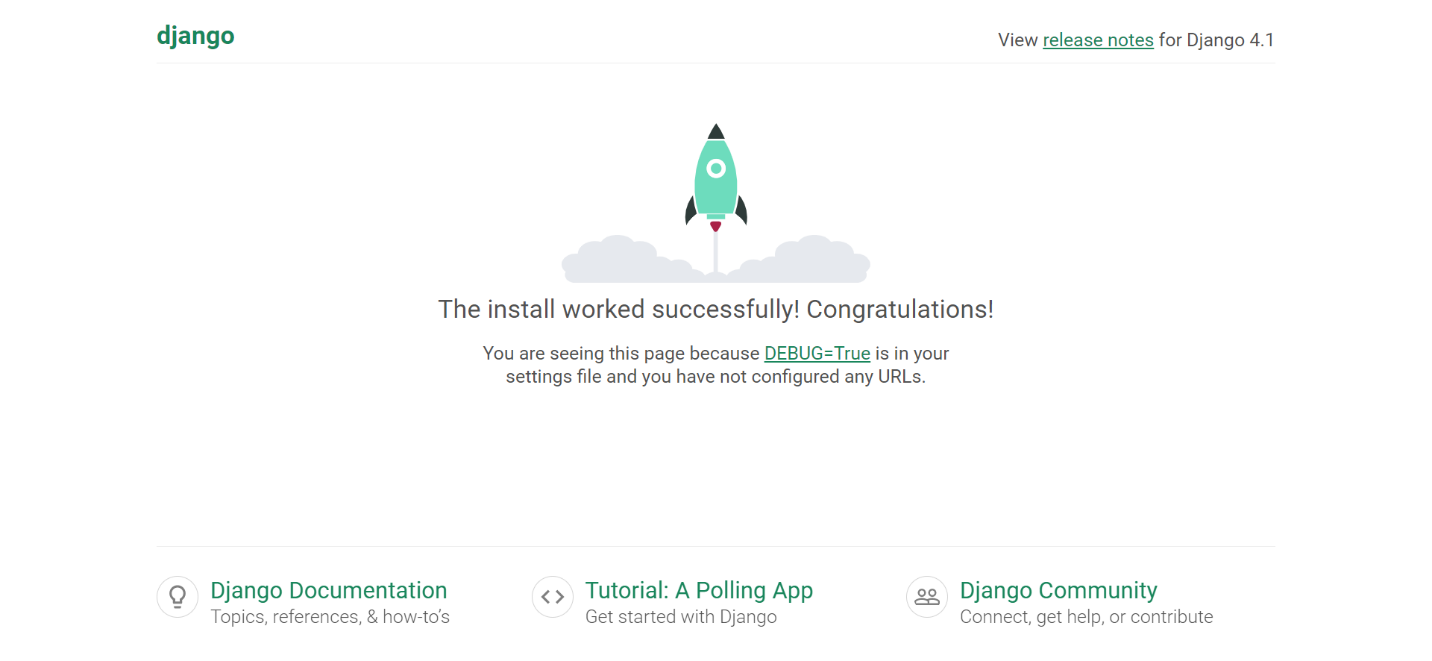
Now if you type this command You will run your server on the local machine

* “python manage .py runserver”

After typing this command in Your cmd You must see such feedback giving you the address of your localhost:



And If you go the 127.0.0.1 on your browser You must see your first webpage like this:



Now that we have created our first webpage There are a few more steps to finish the project.

After running the server now we need to send our data to a webpage, to do that we need to save the data coming from Arduino. There are some ways to do this

One is to use the putty Application or Cool Termwin in this project I use cool- Termwin

By using this app we can save data from our Arduino.

After we did this and save data from our Arduino to our pc we must send data to the website.  
for example, if we want to show the data on the specific page of our website

in this project I wanted to show data on the polls page of my website so I Created a Polls page and upload data into it.to that you need to do these steps:

first, you must create a poll app page with the following command in Your command prompt:

* “python manage.py startapp polls”

That will create directory polls for your website

Now that we have created our desired directory we must upload the data that we saved from our Arduino. to that we must go to the folder of our website and then the polls folder. In the polls, the folder finds view.py, and click on it.

In View.py of the polls page we can write down the desired HTML or anything we want to see on the polls page, to do so let’s create a function like this:

Code:

from Django. shortcuts import render

from Django.http import HttpResponse

import os

os.chdir(r"C:\Users\farzin\Desktop")

f= open("test.txt","r")

def index(request):

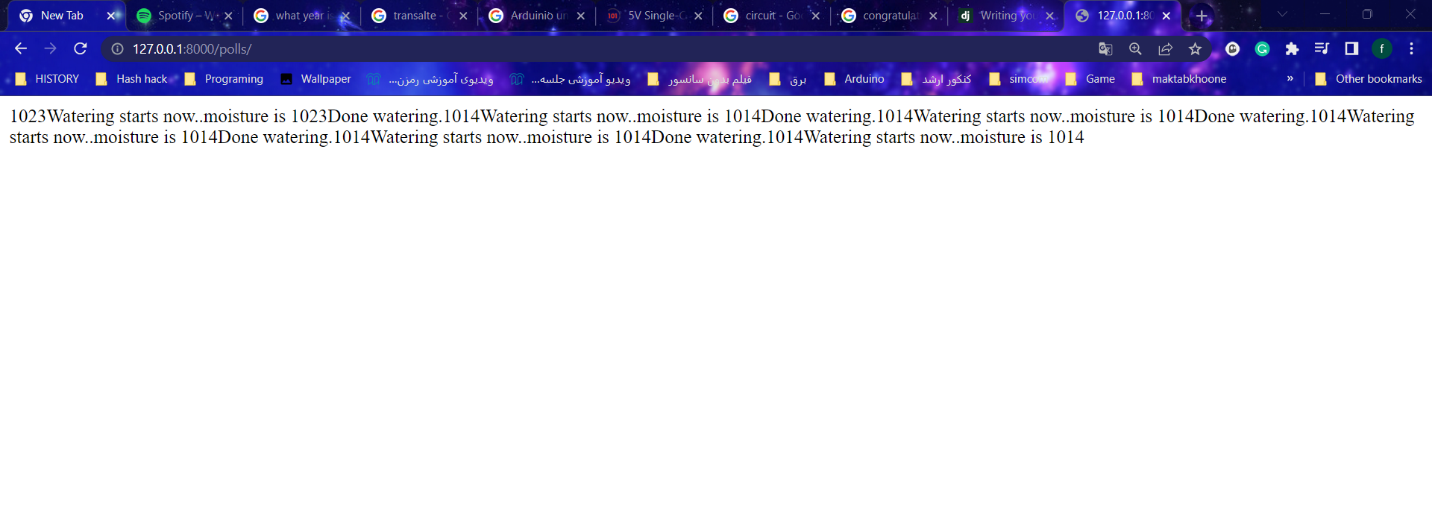
    return HttpResponse(f)

We import the OS library here to tell the python to go to the folder that our coolTermwin will save the data as a Text file, read that file, and append it to our page on our website As you can see in my code first I determine the path that data will be saved on and then open the data and read it to memory and finally return the Value as HttpResponse.

Now if we write this address in our browser we must see the data on our webpage:

<http://127.0.0.1:8000/polls/>

we must see such result as an answer:



Later on, if we want to online this process using the internet we must buy the domain and hosting service, and when we did that we will have this kind of web address

<http://example.com/polls/> instead of localhost.

Online with internet

But what if we want a more user-friendly and also online with the internet website?

In that case, we can use the CMS And Because this is a demo, we can use some free domain and hosting service, in my case I use this hosting service:

<https://profreehost.com/>

It got a good c-panel no limits the size for the site and also provides you with a free domain name.

<http://thisistestsokeepitdownonme.liveblog365.com/>

this is the website that I created for this project and since it’s a free domain service it’s got expire date of my free membership so it’s expected that the website will be shut down after 2023.

Now that we have a friendly user interface website and not just some simple HTML code, the main challenge is to post the data from our local machine to the backend of our online web service

Hopefully, In the world of open source programming and free internet, there is API for that.

Code:

import pandas as pd

import base64

import requests

import JSON

import os

os.chdir(r"C:\Users\farzin\Desktop")

user ='farzin' #Type the username of your cms

password ='\*\*\*\*\*\*\*\*\*\*\*\*\*\*' #tType in your own password here

url ='http://localhost/wordpress/wp-json/wp/v2/pages/'

wp\_connection = user + ':' + password

token = base64.b64encode(wp\_connection.encode())

headers = {'Authorization': 'Basic ' + token.decode('utf-8')}

token.decode('utf-8')

post\_title = "Null:)"

post\_body = open(“Arduino data.txt”,”r”)

page = {'title': post\_title,

            'status': 'publish',

            'content': post\_body,

            'format': 'standard'

            }

print("done")

wp\_request = requests.post(url + '/post', headers=headers, json=page)

print(wp\_request)

explanation of API code:

As the explanation to our code: first, we need the pandas library because we will work with the JSON data.

We also need to request a library because we need to post data to a webserver

And base64 library to encode and decode data

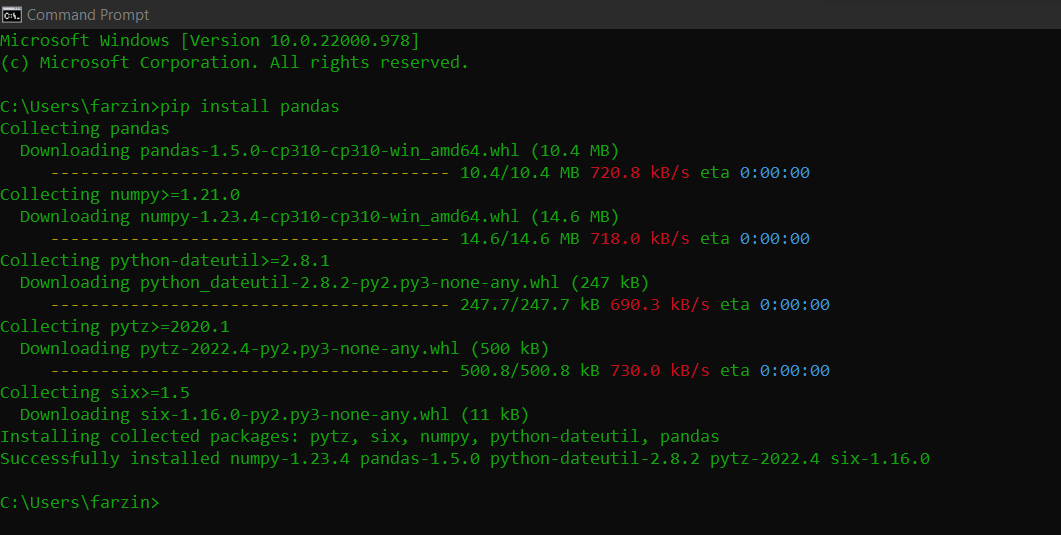
And JSON library to recognize the JSON data

Os library to get the data from our local machine in the directory we’ve saved the data

If this is the first that we run python on the machine we probably need to install some of this library first so we have:

* “pip install pandas”

And as result we expect such things:



In the end, we will print the wp\_request Var because we want to see the feedback from the HTTP

If the HTTP response was 404 it means we post the data to the wrong URL if it was 403 it means it’s forbidden this error probably happens when your country bans your hosting server IP or the server IP provider put you under sanction

If the error was 401 it means your web page can’t handle JSON data

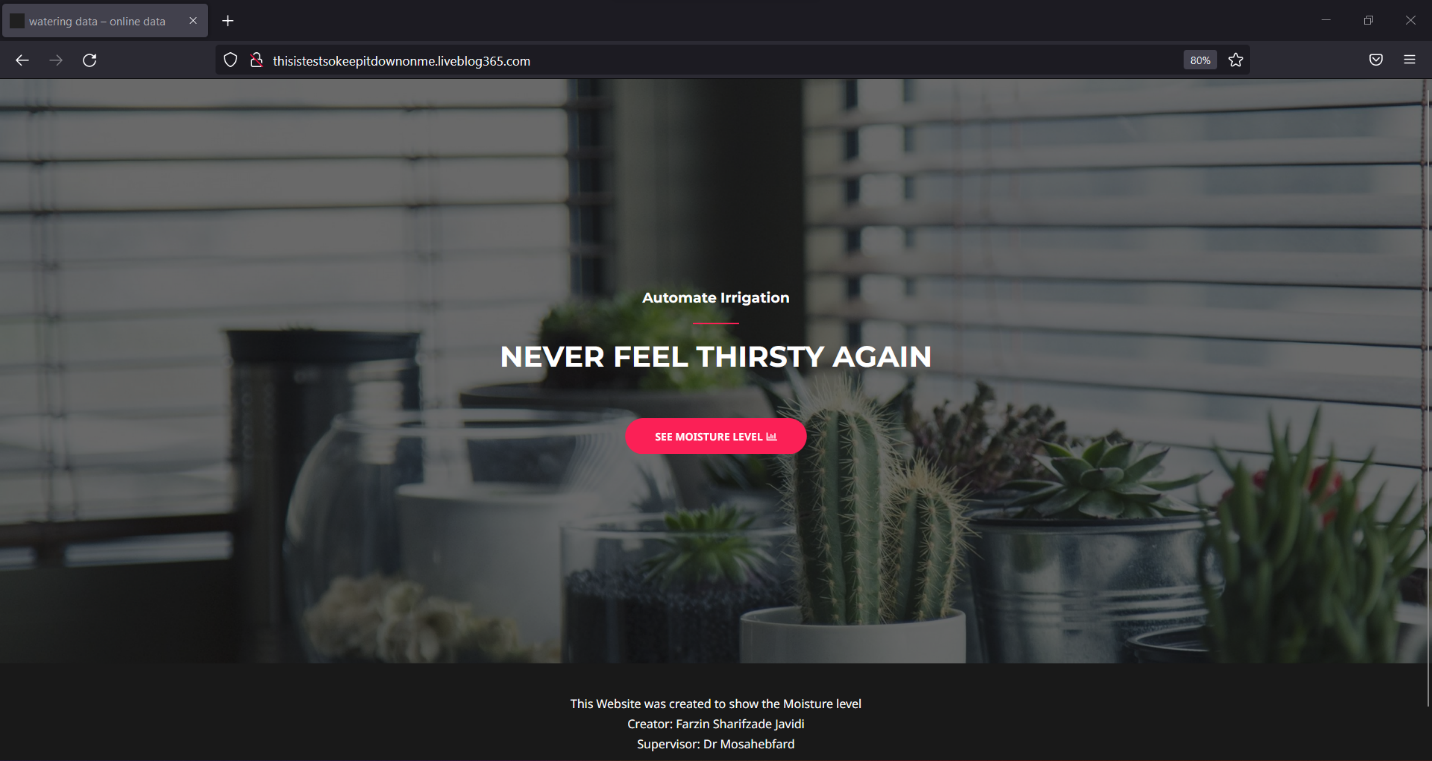
The message we expect to show in our terminal as feedback is 200

HTTP response 200 ok!

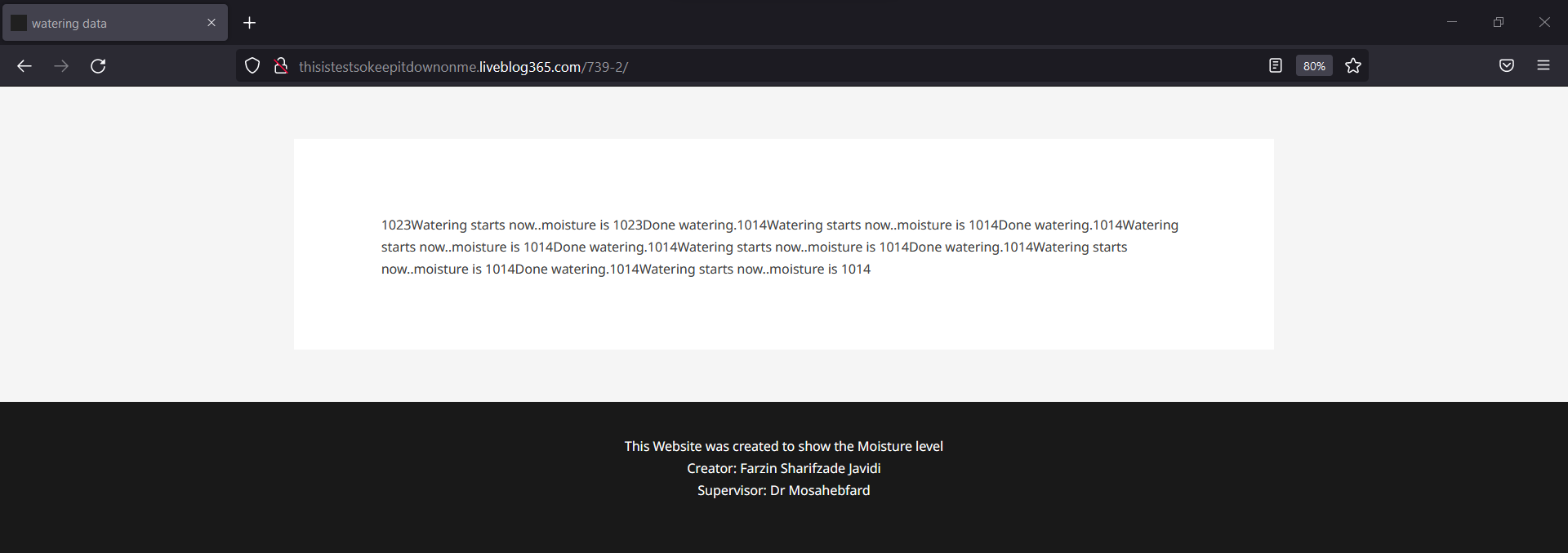
After we manage the backend API and post the data on our website we must see the data on the link we send data to it.

Online website

This is the front of my website and if You click on the Button you will head to the page I created especially for representing data



This is how the second page looks like:



To see the HTML and CSS and javascript code simply click on inspect in your browser to see the code

Conclusion

We create a close loop control system with feedback And create a website to observe the Data from our feedback. The main challenge in this project was to connect the local data on our machine to the backend of our website, The solution to this problem was The API.

If need be there is always a way to draw a diagram, In python, we can do it with matplotlib library. But since it was not the goal of this project we just pass it.

References

* [Arduino Documention](https://www.arduino.cc/)
* [Django Documantion](https://docs.djangoproject.com/en/4.1/)
* [HTTP Respnose](https://www.ibm.com/docs/en/cics-ts/5.2?topic=protocol-http-responses)