

**INTERNET OF THINGS
SMART WEATHER INFORMATION
FINAL PROJECT REPORT**



MEMBERS:

Aarclaine Phanta	(001202100106)
Algifari Abdurahman	(001202100051)
Hebrew Z. Yesly Sagay	(001202100018)
Ricky Chandrean	(001202100077)

**INFORMATION TECHNOLOGY STUDY PROGRAM
FACULTY OF COMPUTER SCIENCE
PRESIDENT UNIVERSITY**

May, 2023

Table of Content

Table of Content	ii
Project Description.....	1
Project Title	1
How the Project Works.....	1
Circuit Diagram	2
List of the Devices	2
Schematic of Wire Configuration.....	2
Blynk App Configuration	5
Sketch.....	11
Code Development	11
Result.....	14
Photos of The Project.....	14
Video Demo Project	15
Worklog.....	16

I. Project Description

a. Project Title

“Smart Weather Information”

b. How the Project Works

For our final project, we create **“Smart Weather Information”** with **“Blynk App”** for Manage all Information about the weather information includes Temperature (BMP180), Altitude (BMP180), Pressure (BMP180), information on the state of the day (sunny, cloudy, dark) (LDR Sensor), Rain Information (Rain Sensor).

The project we are creating has the following flow: when the rain sensor detects water, the NodeMCU will send data to the Blynk application. Then, the Blynk application will display a notification to inform the user that it is raining. In addition, the buzzer will also be activated as an alarm to warn the user that it is raining.

Another feature is the detection of weather conditions on a particular day, whether it is sunny, cloudy, or dark. An LDR sensor will detect the brightness level of the day and categorize it as sunny, cloudy, or dark. If the sensor detects a sunny weather condition, the data will be sent to the Blynk application and display information that it is a sunny day. However, if the weather condition is cloudy or dark, the Blynk application will display information that it is a cloudy or dark day, and the buzzer will be activated to warn the user about the possibility of rain so that the user can prepare if it does rain.

The main idea and objective of this project is to assist users in detecting rain. For example, if a user is drying clothes and is inside a room, so they cannot determine the real-time weather conditions whether it is sunny or rainy, this project will help users by providing notifications if it is raining so that the user can quickly take their clothes off the line to avoid getting wet in the rain.

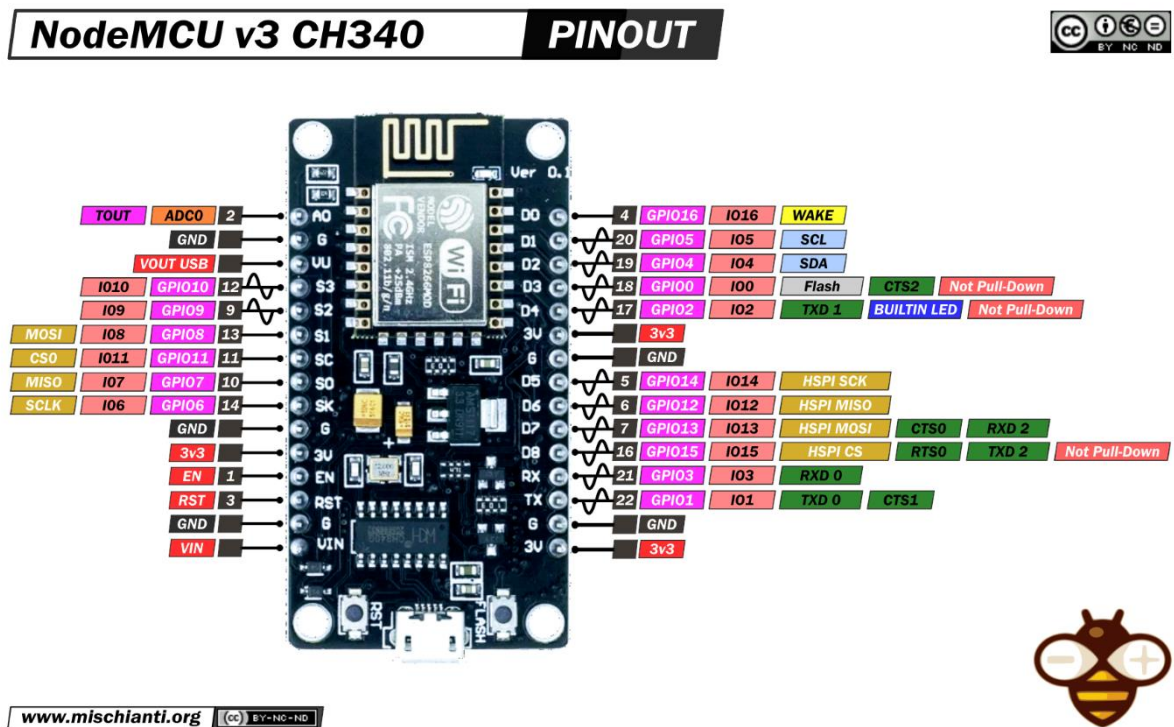
II. Circuit Diagram

a. List of the Devices

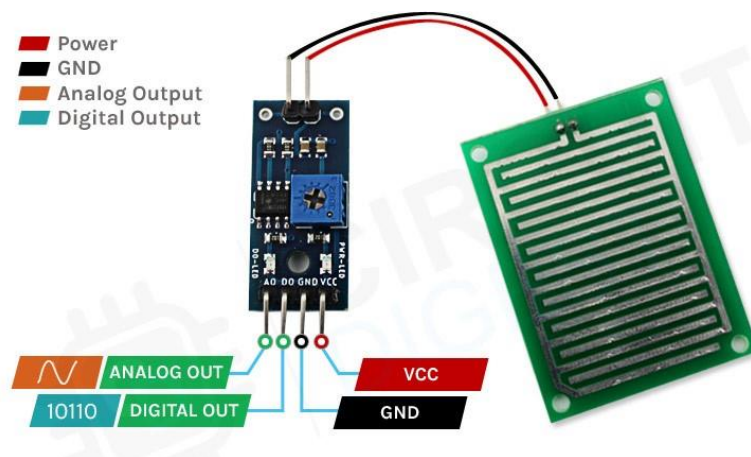
- NodeMCU V3 CH340 Lolin
- Rain Sensor
- LDR Sensor
- BMP180 Sensor
- Buzzer
- LCD 16X2 With I2C Module
- Breadboard

b. Schematic of Wire Configuration

- NodeMCU PINOUT



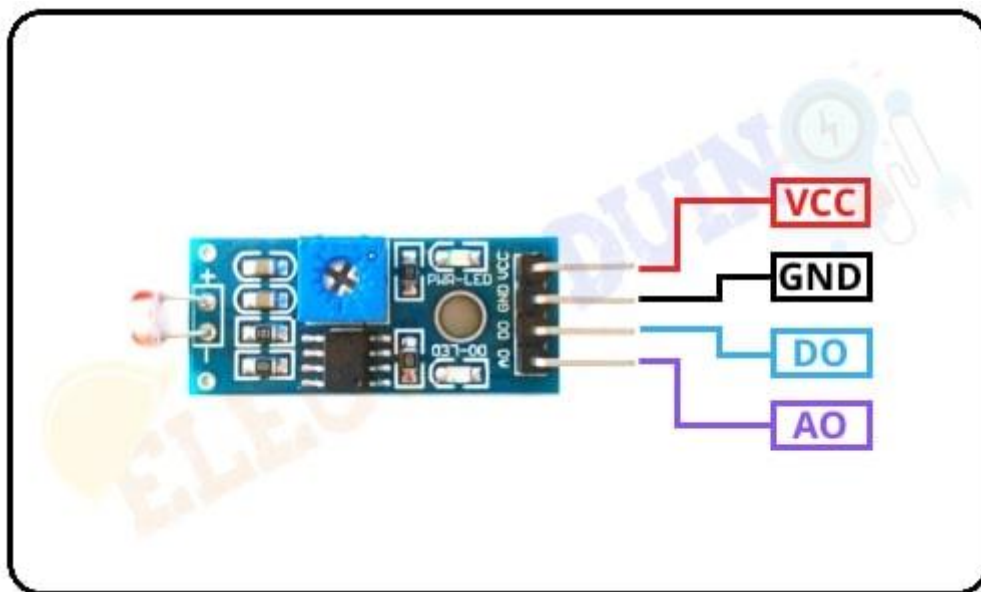
- Rain Sensor Configuration



Rain Sensor → NodeMCU (Rain sensor pin connect directly to NodeMCU)

- VCC → 3V
- GND → GND
- DO → TX (GPIO1)

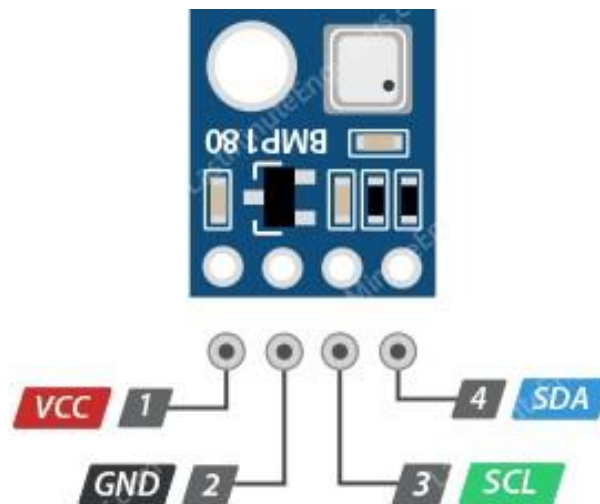
- LDR Sensor Configuration



LDR Sensor → NodeMCU (Ldr sensor pin connect directly to NodeMCU)

- VCC → 3V
- GND → GND
- AO → AO

- BMP180 Sensor Configuration



BMP180 → NodeMCU (BMP180 sensor pin connect directly to NodeMCU)

- VCC → 3V
- GND → GND
- SCL → D1
- SCL → D2

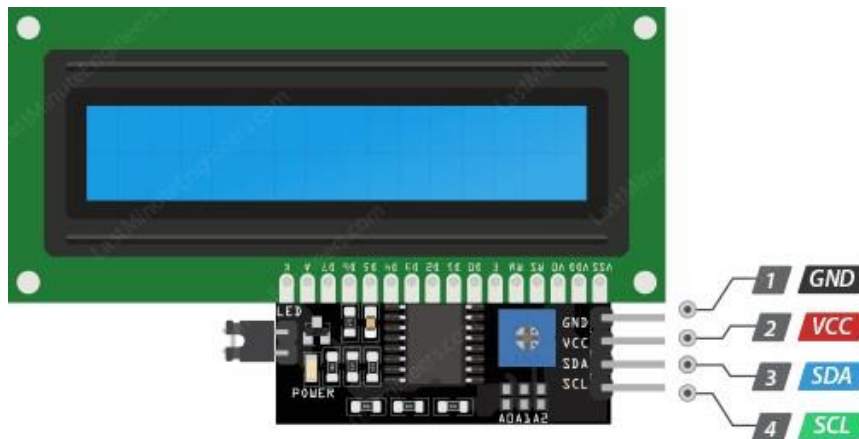
- Buzzer Configuration



Buzzer → Breadboard → NodeMCU (Because our group doesn't have jumper cable Female to Female, so we use breadboard to link Buzzer with NodeMCU)

- Negative (-) → GND (NodeMCU)
- Positive (+) → D5 (GPIO14 NodeMCU)

- **LCD 16X2 With I2C Module Configuration**



LCD → Breadboard ← NodeMCU (Because our group doesn't have jumper Female to Female, so we use breadboard to link LCD with NodeMCU)

- GND → (-) Breadboard ← GND (NodeMCU)
- VCC → (+) Breadboard ← 5V (NodeMCU)
- SDA → Breadboard ← D2 (SDA Pin NodeMCU)
- SCL → Breadboard ← D1 (SCL Pin NodeMCU)

c. **Blynk App Configuration**

- First make configuration in Web App Blynk
- Create Project Template

- Click done

The screenshot shows the Blynk Smart Weather Information template configuration page. The left sidebar contains navigation icons. The main content area has tabs for Info, Metadata, Datastreams, Events, Automations, Web Dashboard, and Mobile Dashboard. The Info tab is active, showing fields for Template Name (Smart Weather Information), Hardware (ESP8266), Connection Type (WiFi), Description (Internet of Things Project: 2023), Template ID (THPL6TUIVFmOE), Manufacturer (My organization 683150), Offline Ignore Period (00 hrs 00 mins 00 secs), and Hotspot Prefix (Hotspot Prefix). There is an 'Add image' section for the Template Image (Optional) and a 'Firmware Configuration' section with a code block containing Blynk template definitions. A note states: 'Template ID and Device Name should be included at the top of your main firmware'. The bottom right corner shows 'Region: sgp1' and a 'Privacy Policy' link.

- Click Datastreams → New Datastream → VirtualPin

Datastream is an important platform component that allows to configure the way data is sent between the device and Blynk.Cloud. We use virtual pin, so the data that we have will send to the pin that we configure for the sensor.

The screenshot shows the Blynk Virtual Pin Datastream configuration page for the LDRSensor. The page has a 'NAME' field (LDRSensor) and an 'ALIAS' field (LDRSensor). The 'PIN' is set to V0 and the 'DATA TYPE' is Integer. The 'UNITS' are set to None. The 'MIN' value is 0 and the 'MAX' value is 1023. The 'DEFAULT VALUE' is Default Value. There is an 'ADVANCED SETTINGS' section. The bottom right corner shows 'Cancel' and 'Save' buttons. The bottom status bar shows 'Region: sgp1' and a 'Privacy Policy' link.

The screenshot shows the Blynk Virtual Pin Datastream configuration page for the rainSensor. The page has a 'NAME' field (rainSensor) and an 'ALIAS' field (rainSensor). The 'PIN' is set to V1 and the 'DATA TYPE' is Integer. The 'UNITS' are set to None. The 'MIN' value is 0 and the 'MAX' value is 1. The 'DEFAULT VALUE' is 0. There is an 'ADVANCED SETTINGS' section. The bottom right corner shows 'Cancel' and 'Save' buttons. The bottom status bar shows 'Region: sgp1' and a 'Privacy Policy' link.

B

Smart Weather Informati...

Cancel

Save

Virtual Pin Datastream

NAME

weatherColor

ALIAS

weatherColor

PIN

V2

DATA TYPE

Integer

UNITS

None

MIN

0

MAX

1

DEFAULT VALUE

0

ADVANCED SETTINGS

Cancel

Save

6

currentDate

currentDate

Region: sgp1

Privacy Policy

B

Smart Weather Informati...

Cancel

Save

Virtual Pin Datastream

NAME

weatherInfo

ALIAS

weatherInfo

PIN

V3

DATA TYPE

String

DEFAULT VALUE

Default Value

ADVANCED SETTINGS

Cancel

Save

6

currentDate

currentDate

Region: sgp1

Privacy Policy

B

Smart Weather Informati...

Cancel

Save

Virtual Pin Datastream

NAME

currentTime

ALIAS

currentTime

Use letters, digits, spaces, and underscores only

PIN

V4

DATA TYPE

String

DEFAULT VALUE

Default Value

ADVANCED SETTINGS

Cancel

Save

6

currentDate

currentDate

Region: sgp1

Privacy Policy

B

Smart Weather Informati...

Cancel

Save

Virtual Pin Datastream

NAME

currentDate

ALIAS

currentDate

PIN

V5

DATA TYPE

String

DEFAULT VALUE

Default Value

ADVANCED SETTINGS

Cancel

Save

6

currentDate

currentDate

Region: sgp1

Privacy Policy

B

Smart Weather Informati...

Cancel

Save

Virtual Pin Datastream

NAME

weatherEmoji

ALIAS

weatherEmoji

PIN

V6

DATA TYPE

String

DEFAULT VALUE

Default Value

ADVANCED SETTINGS

Cancel

Save

10

temperatureBMP180

temperatureBMP180

Region: sgp1

Privacy Policy

B

Smart Weather Informati...

Cancel

Save

Virtual Pin Datastream

NAME

pressureBMP180

ALIAS

pressureBMP180

PIN

V7

DATA TYPE

Double

UNITS

None

MIN

0

MAX

1000000

DECIMALS

###

DEFAULT VALUE

Default Value

ADVANCED SETTINGS

Cancel

Save

10

temperatureBMP180

temperatureBMP180

Region: sgp1

Privacy Policy

Virtual Pin Datastream

NAME: altitudeBMP180 ALIAS: altitudeBMP180

PIN: V8 DATA TYPE: Double

UNITS: None

MIN: 0 MAX: 255 DECIMALS: ### DEFAULT VALUE: Default Value

ADVANCED SETTINGS

Buttons: Cancel, Save

Virtual Pin Datastream

NAME: temperatureBMP180 ALIAS: temperatureBMP180

PIN: V9 DATA TYPE: Double

UNITS: None

MIN: 0 MAX: 255 DECIMALS: ### DEFAULT VALUE: Default Value

ADVANCED SETTINGS

Buttons: Cancel, Save

- After that Click the Automation. Automation is for show the notification
- Change Type of Automation to Sensor and the condition to on

Smart Weather Information

Info Metadata Datastreams Events **Automations** Web Dashboard Mobile Dashboard

Define which Datastreams will be available in Automation actions and conditions. Only Virtual Pin, Enumerable and Location Datastreams supported.

Search datastream

10 Datastreams

Name	Pin	Data Type	Type Of Automation	Condition	Action
LDRSensor	V0	Integer	Switch	<input type="checkbox"/>	<input type="checkbox"/>
rainSensor	V1	Integer	Sensor	<input checked="" type="checkbox"/>	<input type="checkbox"/>
weatherColor	V2	Integer	Switch	<input type="checkbox"/>	<input type="checkbox"/>

Region: sgp1 Privacy Policy

- After that we go to search and klik Device and create device from the template that we already created before.

My Devices

+ New Device

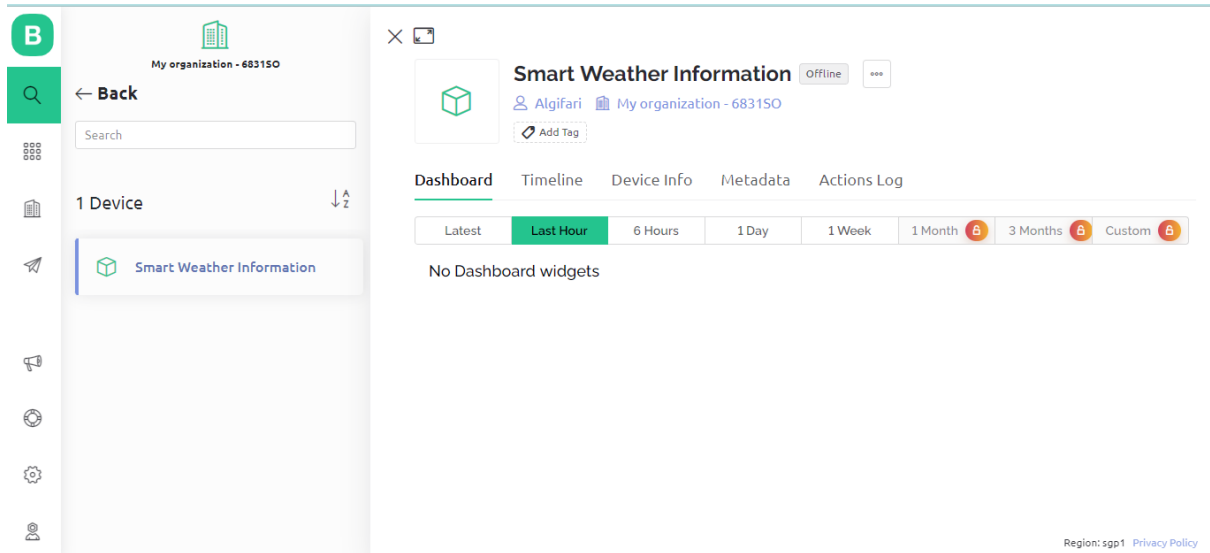
New Device

Create new device by filling in the form below

TEMPLATE: Smart Weather Information

DEVICE NAME: Smart Weather Information

Buttons: Cancel, Create



- In Device Info we can copy the FIRMWARE CONFIGURATION to our Code.

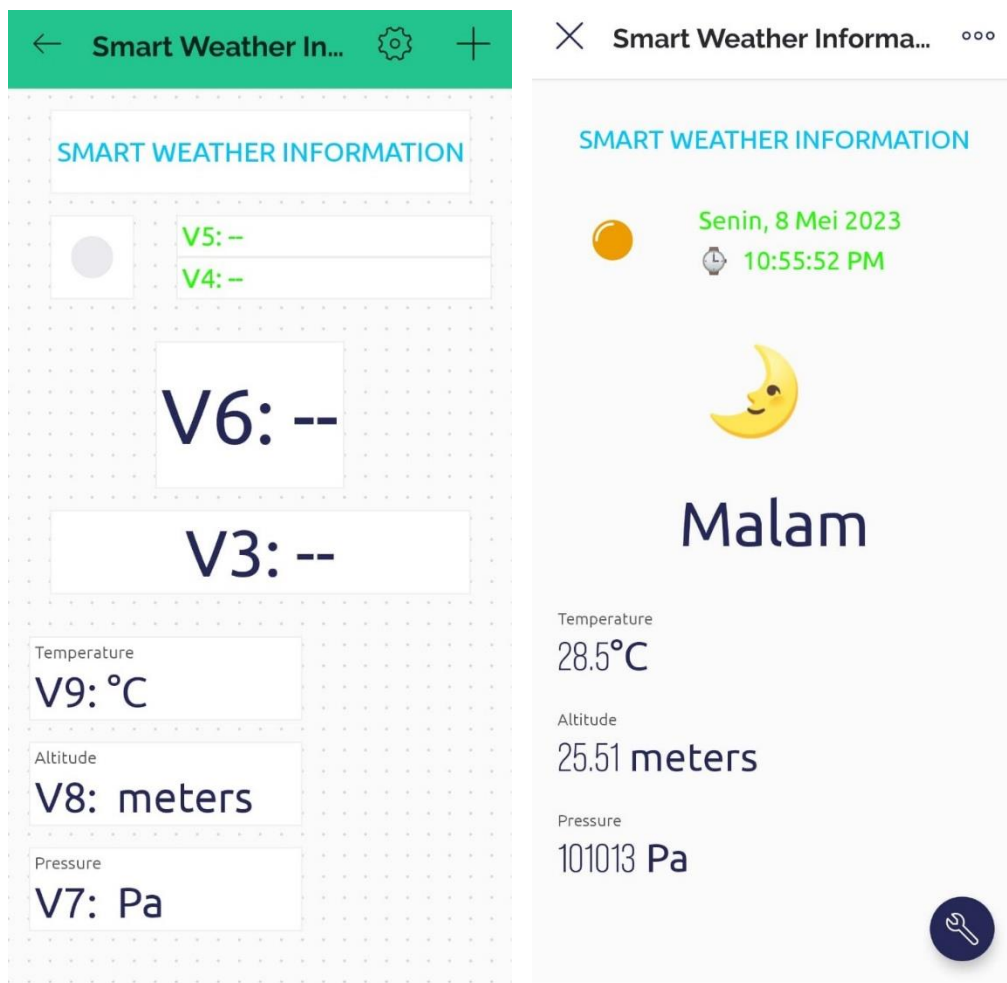
FIRMWARE CONFIGURATION

```
#define BLYNK_TEMPLATE_ID
"TMPL6TUIVFm0E"
#define BLYNK_TEMPLATE_NAME "Smart
Weather Information"
#define BLYNK_AUTH_TOKEN
"l7XFtoHgLFv3Z5nyRnsWw6zwAHdhUZKW"
```

Template ID, Device Name, and AuthToken should be declared at the very top of the firmware code.

- Copy the firmware configuration (Blynk) above to the code (Arduino IDE)

- After that we make configuration of UI Blynk App in Smartphone



III. Sketch

a. Code Development

- Morning Condition

```
//Weather Info
// Morning, 00:00 AM sd 09:59 AM
if((hour() >= 0)&&(hour() < 10))
{
  if(rainSensorDigital==true) // Not Raining
  {
    if(ldrSensorAnalog <=349) //if the brightness is less than or equal 349
    {
      weatherInfo = "Sunny Morning"; //set the weather info to sunny morning
      Blynk.setProperty(V2, "color", "BLYNK_YELLOW"); //set a button wheather info to yellow
      Blynk.virtualWrite(V3, weatherInfo); //send and display weather info in v3 data
      Blynk.virtualWrite(V6, "\xF0\x9F\x8C\x9E "); //Unicode (UTF-8) Character for ☀
    }
    if((ldrSensorAnalog >=350)&&(ldrSensorAnalog <=449)) //if the brightness is between 350 and 449
    {
      buzzer(); //set buzzer active
      weatherInfo = "Cloudy Morning"; //set the weather info to cloudy morning
      Blynk.setProperty(V2, "color", "#A4A4A4"); //set a button weather info to grey
      Blynk.virtualWrite(V3, weatherInfo); //send and display weather info in V3 data
      Blynk.virtualWrite(V6, "\xE2\x9B\x85 "); //Unicode (UTF-8) Character for ☁
    }
    if(ldrSensorAnalog >=450) //if the brightness is greater than equal 450
    {
      buzzer(); //set buzzer active
      weatherInfo = "Dark Morning"; //set the weather info to dark morning
      Blynk.setProperty(V2, "color", "#000000"); //set a button weather info to black
      Blynk.virtualWrite(V3, weatherInfo); //send and display weather info in V3 data
      Blynk.virtualWrite(V6, "\xE2\x98\x81 "); //Unicode (UTF-8) Character for ☾
    }
  }
}
```

```
else //Raining
{
  buzzer(); //set buzzer active
  weatherInfo = "Rainy Morning"; //set the weather info to rainy morning
  Blynk.virtualWrite(V3, weatherInfo); //send and display weather info in V3 data
  Blynk.virtualWrite(V6, "\xE2\x98\x94 "); //Unicode (UTF-8) Character for 🌧
  if(ldrSensorAnalog <=349) //if the brightness is less than or equal 349
  {
    Blynk.setProperty(V2, "color", BLYNK_YELLOW); //set a button wheather info to yellow
  }
  if((ldrSensorAnalog >=350)&&(ldrSensorAnalog <=449)) //if the brightness between 350 and 449
  {
    Blynk.setProperty(V2, "color", "#A4A4A4"); //set a button wheather info to grey
  }
  if(ldrSensorAnalog >=450) //if the brightness is greater than equal 450
  {
    Blynk.setProperty(V2, "color", "#000000"); //set a button wheather info to black
  }
}
}
```

- Afternoon Condition

```
// Afternoon, 10:00 AM sd 05:59 PM
if((hour() >= 9)&&(hour() < 18))
{
    if(rainSensorDigital==true) // Not Raining
    {
        if(ldrSensorAnalog <=249) //if the brightness is greater than equal 249
        {
            weatherInfo = "Sunny Afternoon"; //set the weather info to sunny afternoon
            Blynk.setProperty(V2, "color", BLYNK_YELLOW); //set a button wheather info to yellow
            Blynk.virtualWrite(V3, weatherInfo); //display weather info in V3 data
            Blynk.virtualWrite(V6, "\xE2\x98\x80 "); //Unicode (UTF-8) Character for ☀
        }
        if((ldrSensorAnalog >=250)&&(ldrSensorAnalog <=449)) //if the brightness between 250 and 449
        {
            buzzer(); //set buzzer active
            weatherInfo = "Cloudy Afternoon"; //set the weather info to cloudy afternoon
            Blynk.setProperty(V2, "color", "#A4A4A4"); //set a button wheather info to grey
            Blynk.virtualWrite(V3, weatherInfo); //send and display weather info in V3 data
            Blynk.virtualWrite(V6, "\xE2\x9B\x85 "); //Unicode (UTF-8) Character for ☁
        }
        if(ldrSensorAnalog >=450) //if the brightness greater than 450
        {
            buzzer(); //set buzzer active
            weatherInfo = "Dark Afternoon"; //set the weather info to dark afternoon
            Blynk.setProperty(V2, "color", "#000000"); //set a button wheather info to black
            Blynk.virtualWrite(V3, weatherInfo); //send and display weather info in V3 data
            Blynk.virtualWrite(V6, "\xE2\x98\x81 "); //Unicode (UTF-8) Character for ☾
        }
    }
}
```

```
else //Raining
{
    buzzer(); //set buzzer active
    weatherInfo = "Rainy Afternoon"; //set the weather info to Rainy Afternoon
    Blynk.virtualWrite(V3, weatherInfo); //send and display weather info in V3 data
    Blynk.virtualWrite(V6, "\xE2\x98\x94 "); //Unicode (UTF-8) Character for 🌧
    if(ldrSensorAnalog <=249) //if the brightness is greater than equal 249
    {
        Blynk.setProperty(V2, "color", BLYNK_YELLOW); //set a button wheather info to yellow
    }
    if((ldrSensorAnalog >=250)&&(ldrSensorAnalog <=449)) //if the brightness between 250 and 449
    {
        Blynk.setProperty(V2, "color", "#A4A4A4"); //set a button wheather info to grey
    }
    if(ldrSensorAnalog >=450) //if the brightness greater than 450
    {
        Blynk.setProperty(V2, "color", "#000000"); //set a button wheather info to black
    }
}
}
```

- Evening Condition

```
// Evening, 06:00 PM sd 11:59 PM
if((hour() >= 18)&&(hour() < 24))
{
  if(rainSensorDigital==true) // Not Raining
  {
    weatherInfo = "Evening"; //set the weather info to Evening
    Blynk.setProperty(V2, "color", "#000000"); //set a button wheather info to black
    Blynk.virtualWrite(V3, weatherInfo); //send and display weather info in V3 data
    Blynk.virtualWrite(V6, "\xF0\x9F\x8C\x9B "); //Unicode (UTF-8) Character for 🌙
  }
  else //Raining
  {
    buzzer(); //set buzzer active
    weatherInfo="Rainy Evening"; //set the weather info to Rainy Evening
    Blynk.virtualWrite(V3, weatherInfo); //send and display weather info in V3 data
    Blynk.virtualWrite(V6, "\xE2\x98\x94 "); //Unicode (UTF-8) Character for ☔
    if(ldrSensorAnalog >= 0) //if the brightness greater than or equal 0
    {
      Blynk.setProperty(V2, "color", "#000000"); //set a button wheather info to black
    }
  }
}
}
```

- Send The Data

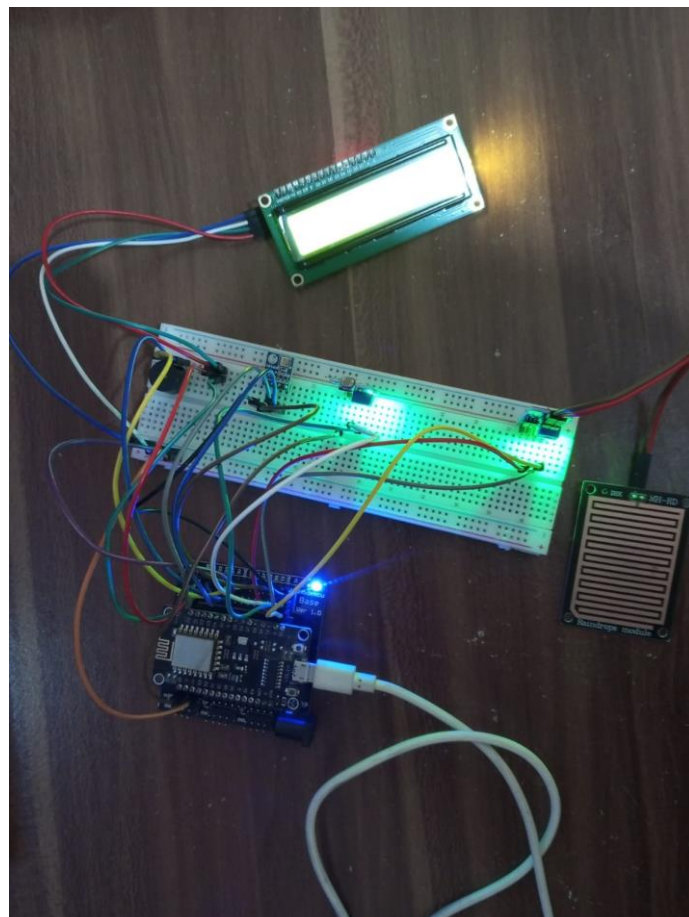
```
//send and display data values to virtual pins in the blynk app
Blynk.virtualWrite(V0, ldrSensorAnalog);
Blynk.virtualWrite(V1, rainSensorDigital);
Blynk.virtualWrite(V2, colorStation);
Blynk.virtualWrite(V4, "\xE2\x8C\x9A ", currentTime);
Blynk.virtualWrite(V5, currentDate);
Blynk.virtualWrite(V7, bp);
Blynk.virtualWrite(V8, ba);
Blynk.virtualWrite(V9, bt);
```

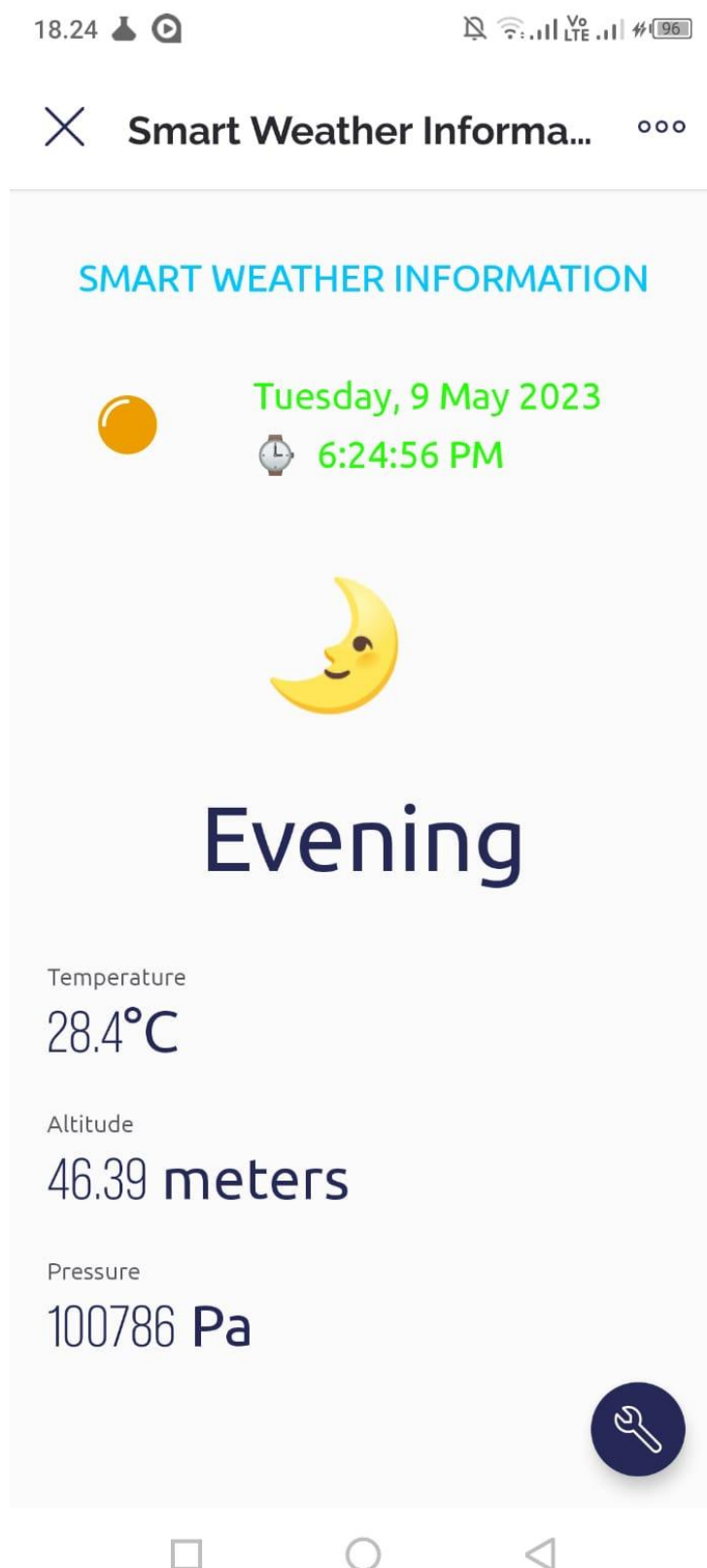
- Below is the link for full code

https://drive.google.com/file/d/12rlyKNOU-s3OsbTnA6gaqThDWTYmPJTH/view?usp=share_link

IV. Result

a. Photos of The Project





b. Video Demo Project

https://drive.google.com/file/d/1DVcV2LiFRu9CFhByelYMC9EXFa7o4kDH/view?usp=share_link

V. Worklog

- Algifari, Aarclaine, Hebrew → Make a Schematic of Wire Configuration
- Algifari, Ricky → Develop the code
- Algifari → Create the Report