**Part1**

**Connecting Esp32 to Blynk to Stream Humidity and Temperature Data**

**Introduction**

This guide will take you through the process of connecting Esp32 to the Blynk platform to stream humidity and temperature data from DHT11 sensor. Blynk is a popular IoT platform that enables users to remotely monitor and control devices through the internet.

**Hardware Requirements:**

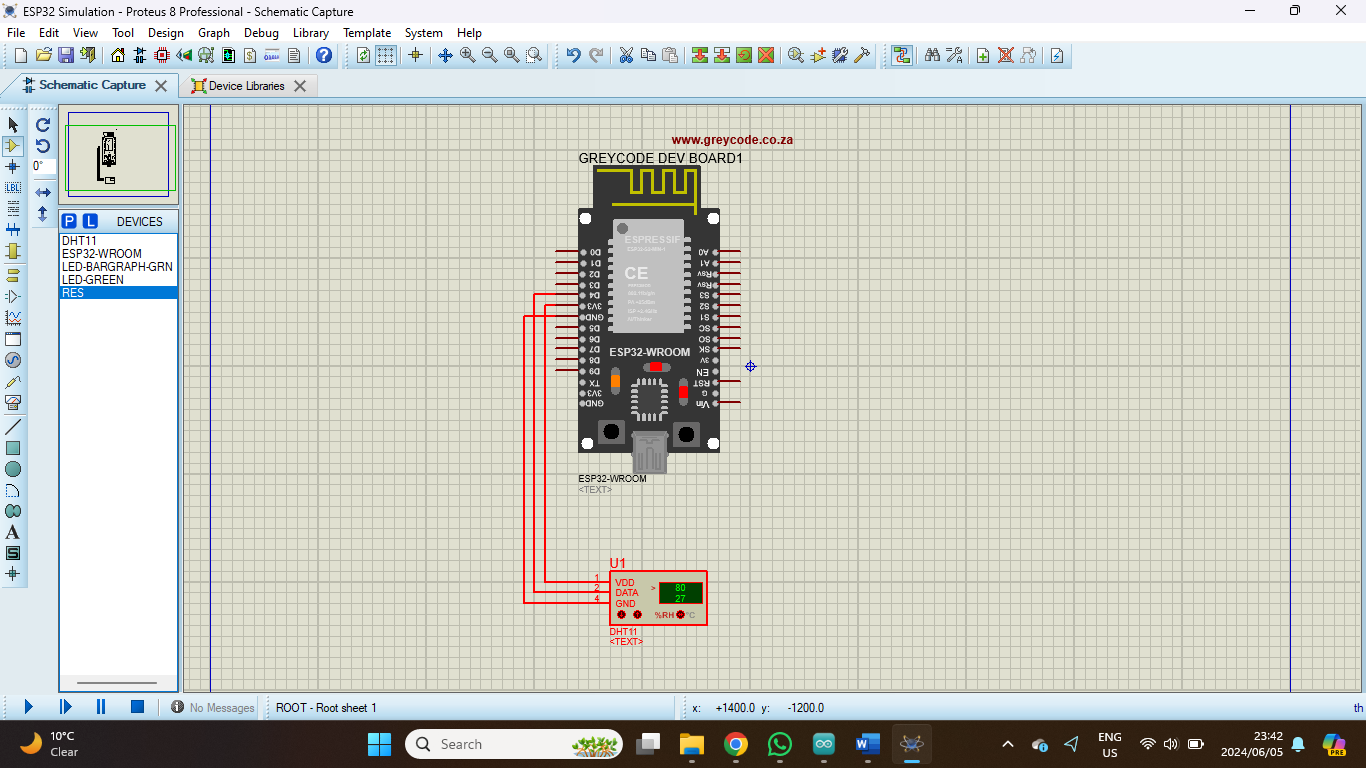
* Esp32
* Humidity sensor (e.g., DHT11 or DHT22)
* Breadboard and jumper wires
* Power source (e.g., USB cable or battery)

**Software Requirements:**

* Arduino IDE
* ESP32i library
* Blynk library

1. Set up the Hardware

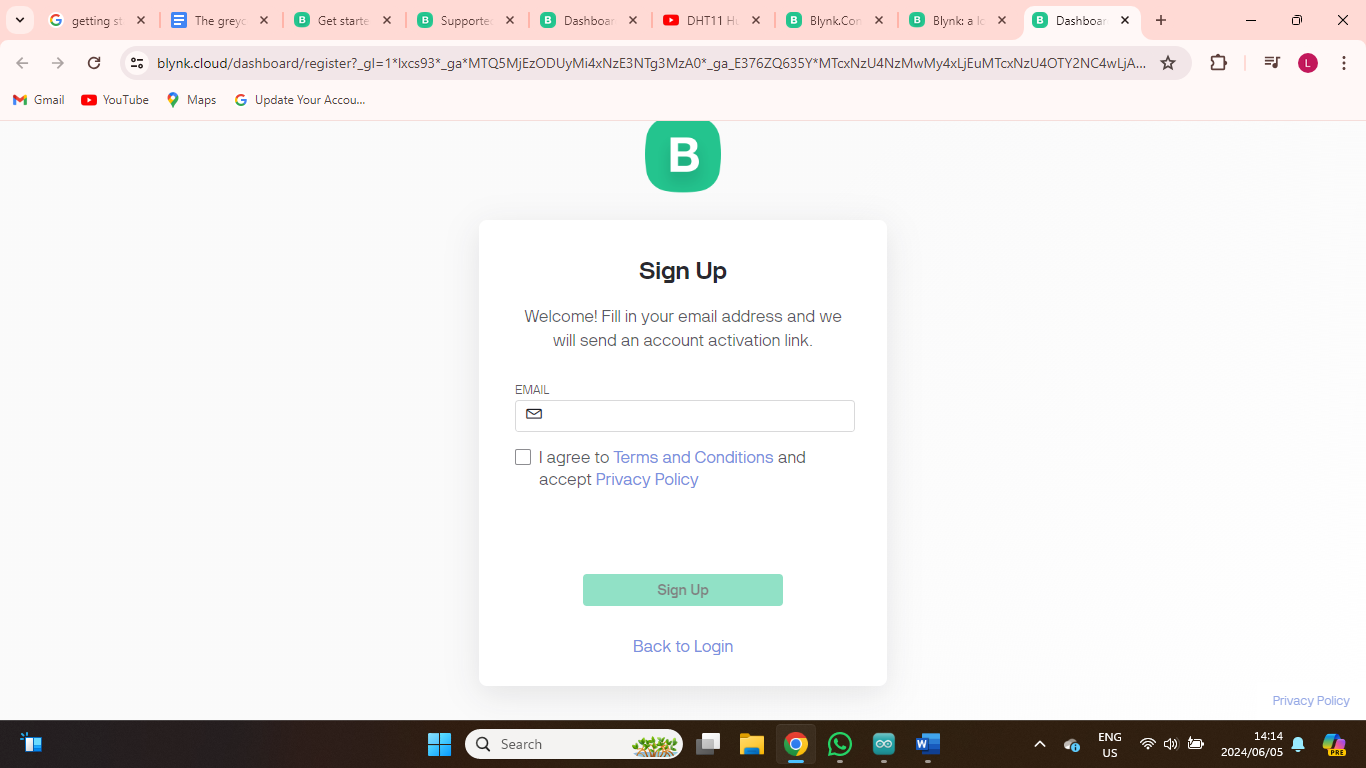
* Connect the humidity sensor to Esp32 according to the sensor's pinout.



1. Install the Necessary Libraries

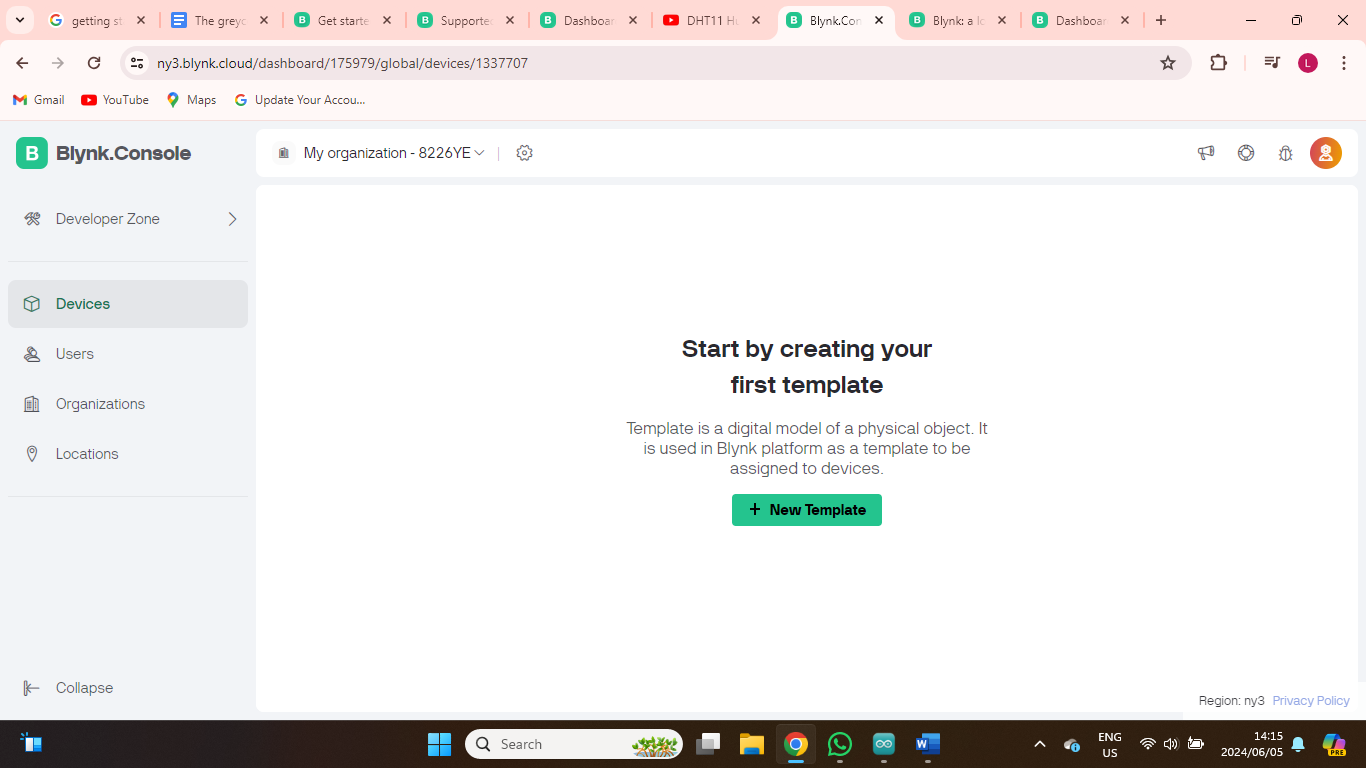
* Open the Arduino IDE and install the Blynk library.

1. Create a Blynk Account and Project <https://blynk.io/>
2. Download the Blynk IoT mobile app on your phone.

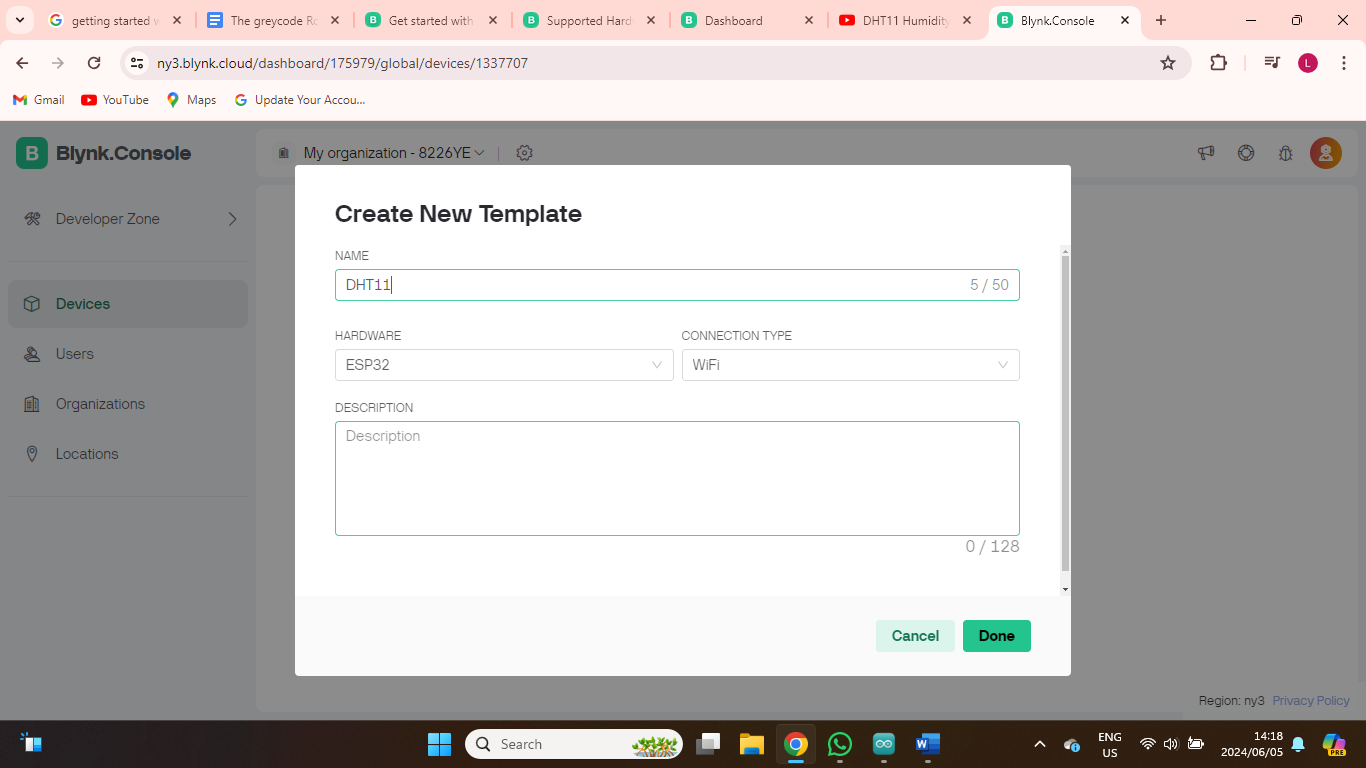


1. Create a new project

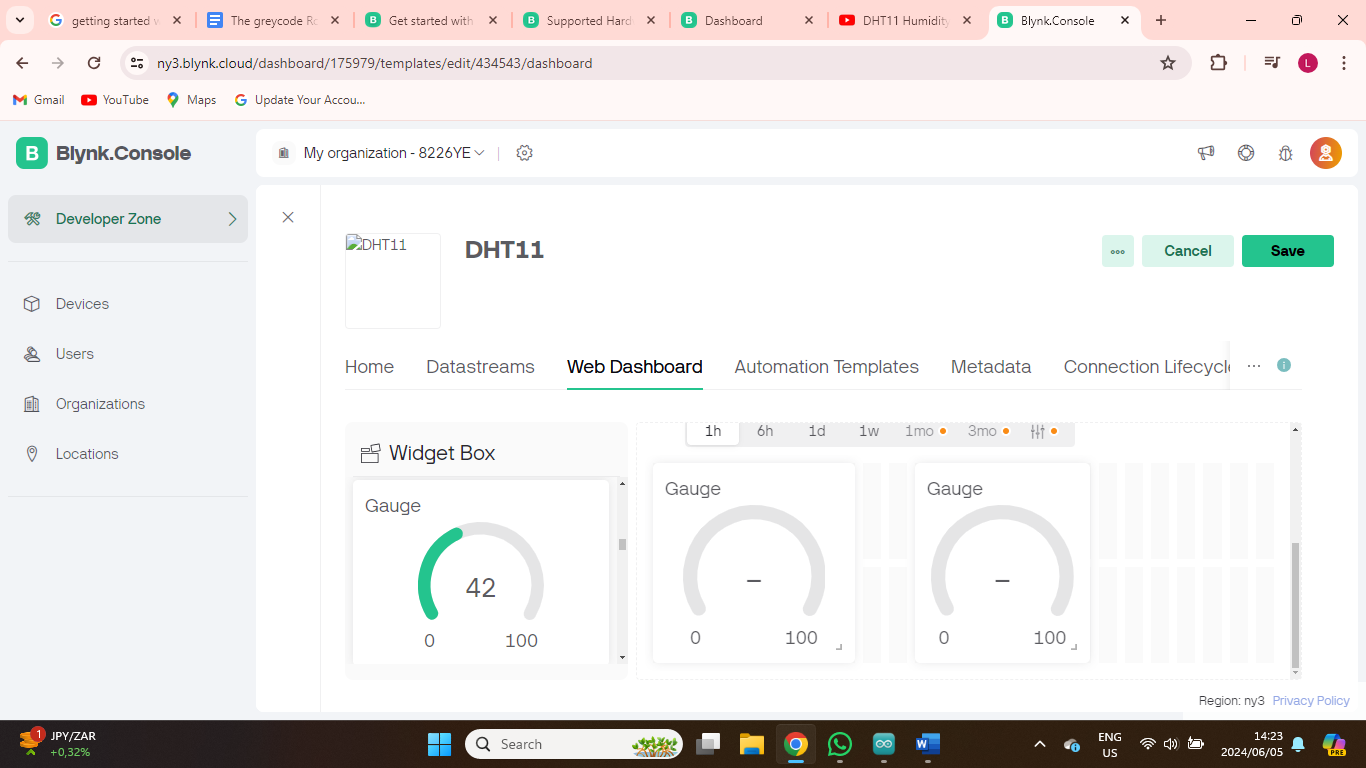
* Go to device
* Add New Template



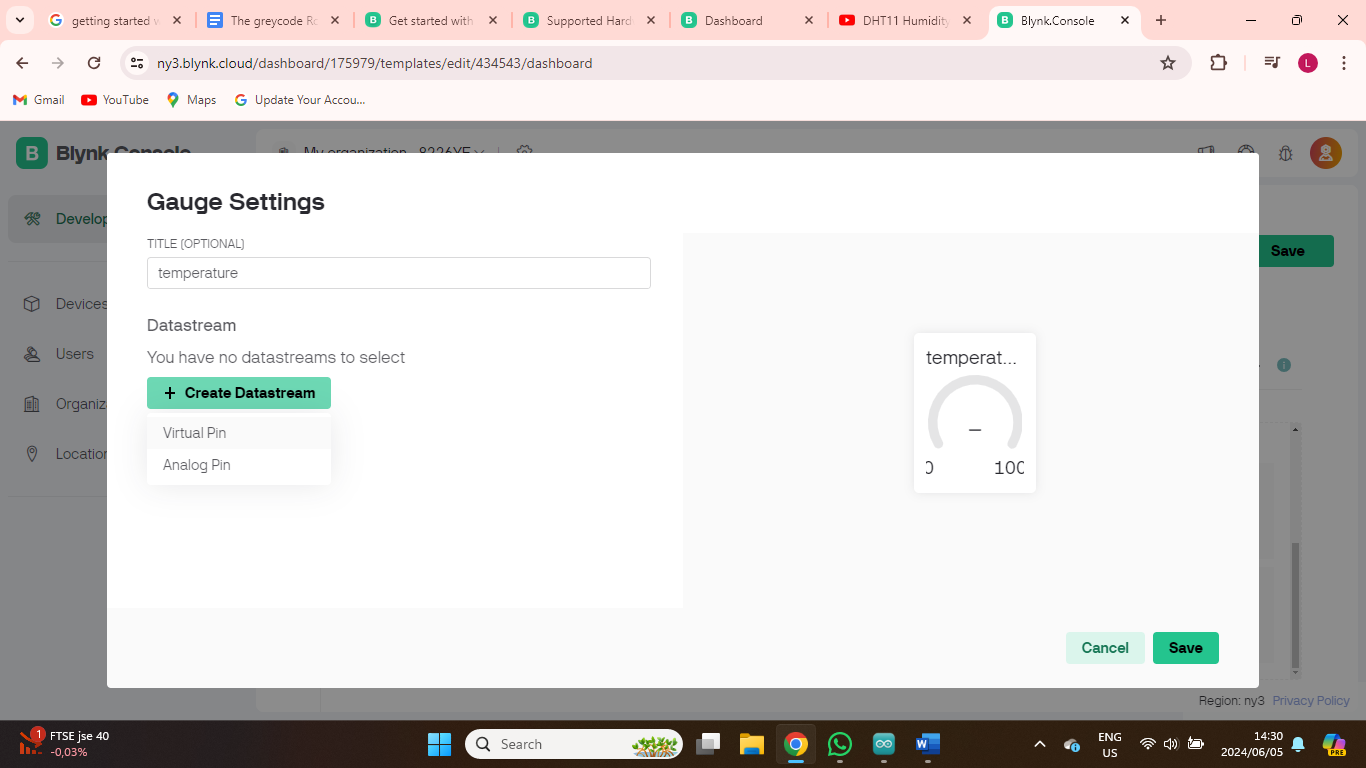
* Give a name to the template click done

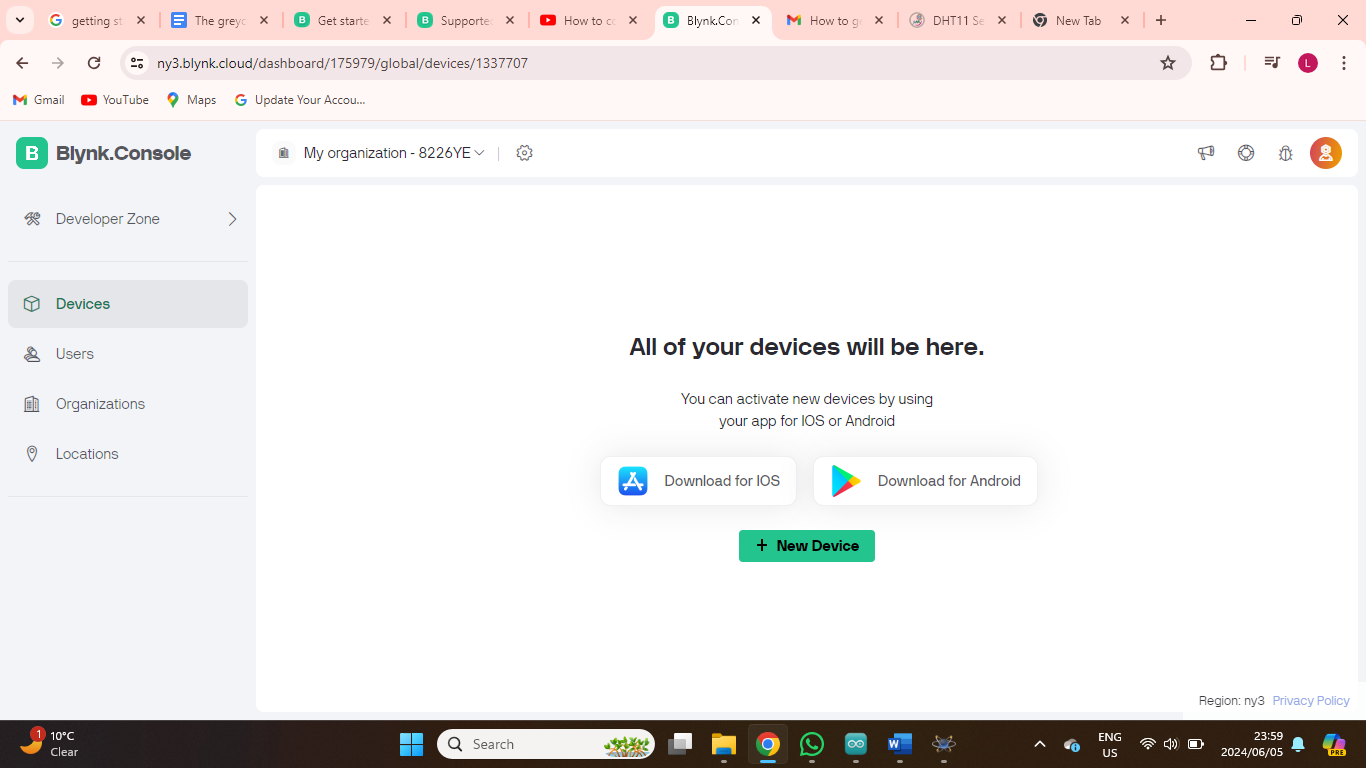


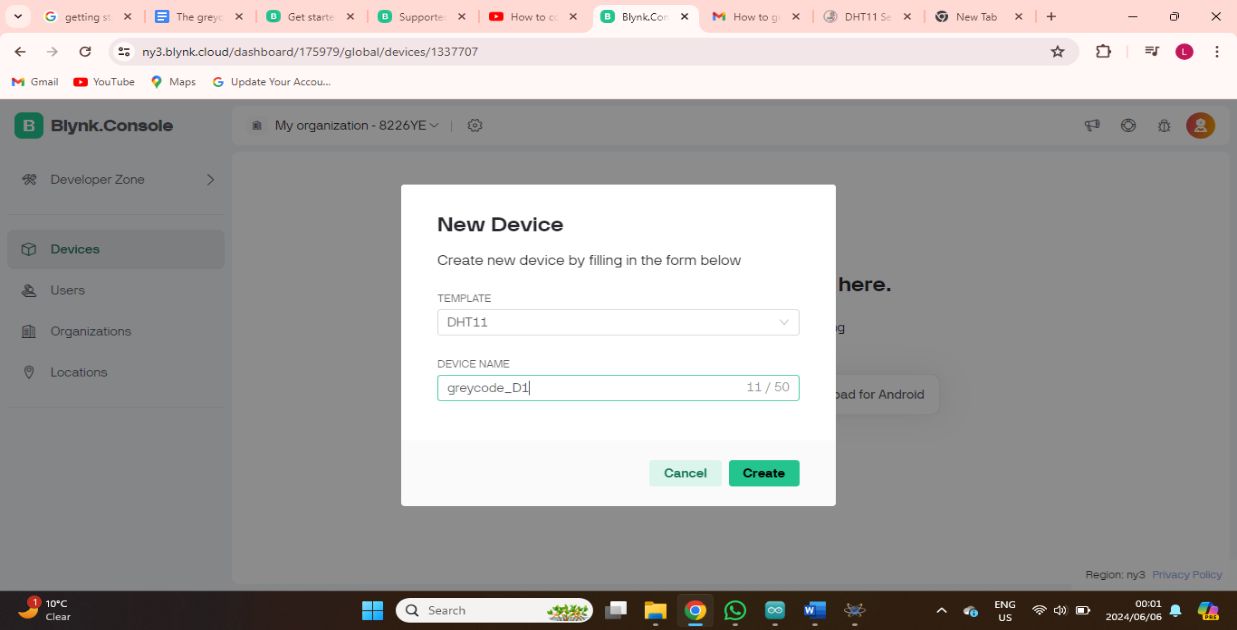
* Navigate to Web Dashboard
* On the Widget Box select Gauge widget, drag and place the widget on the dashboard.
* Double the widget for humidity data.



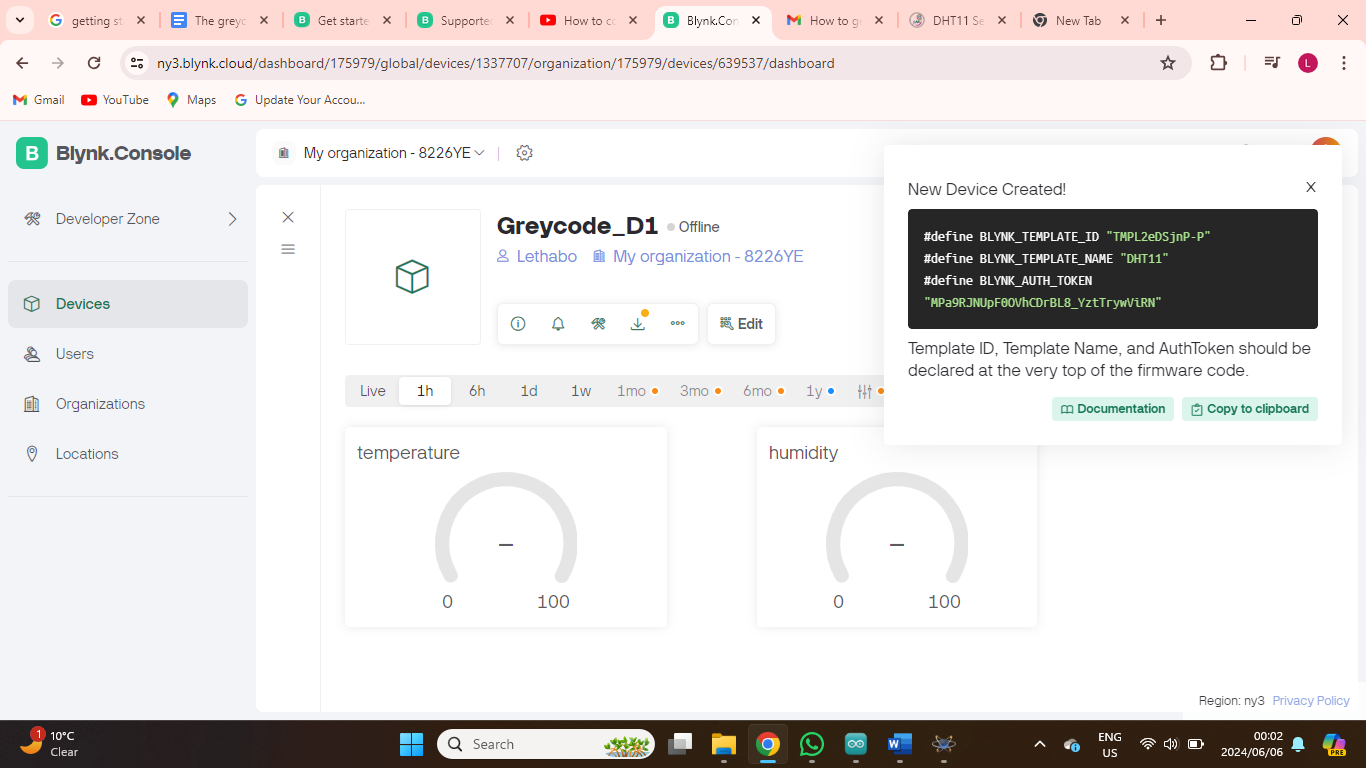
* On each widget click on settings and name it temperature and humidity respectively.
* Click Create Datastream and select virtual pin.
* Fill in the form that follows:
* PIN: Select V0 for humidity and V1 for temperature
* Data type: Select Boolean on both humidity and temperature
* UNITS: Select percentage% for humidity and °C for temperature
* MIN: 0 on both humidity and temperature
* MAX: 100 on both humidity and temperature
* DEFAULT VALUE: 0 on both humidity and temperature
* Click save



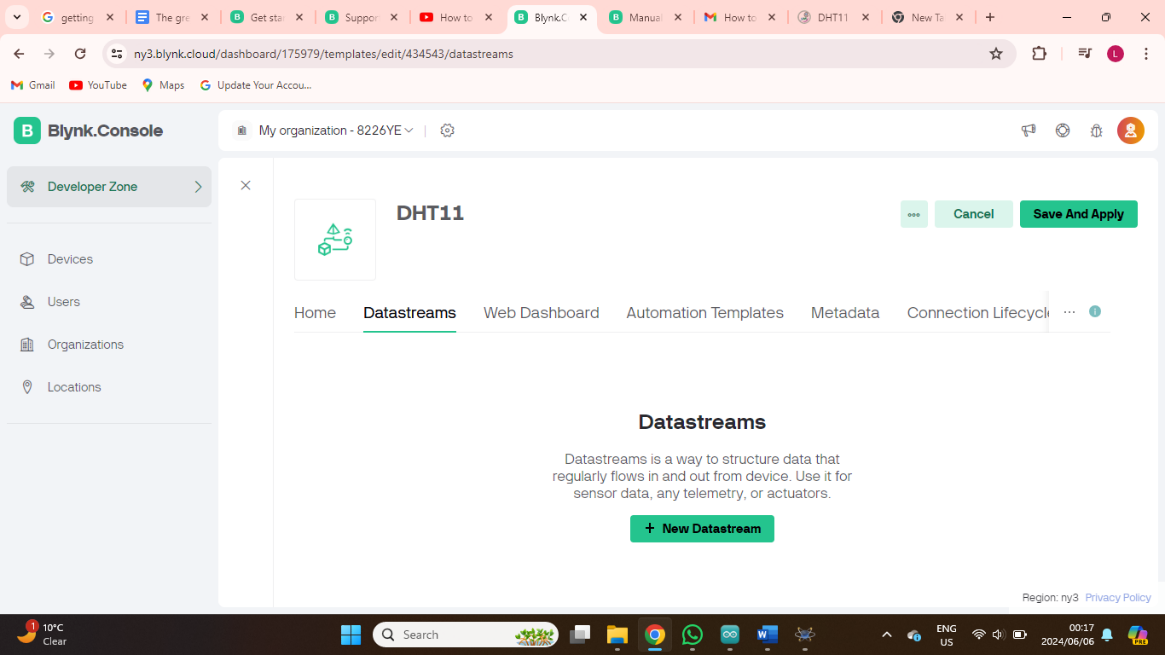
* Add New Device
* Select From Template
* Give the device a name and Create



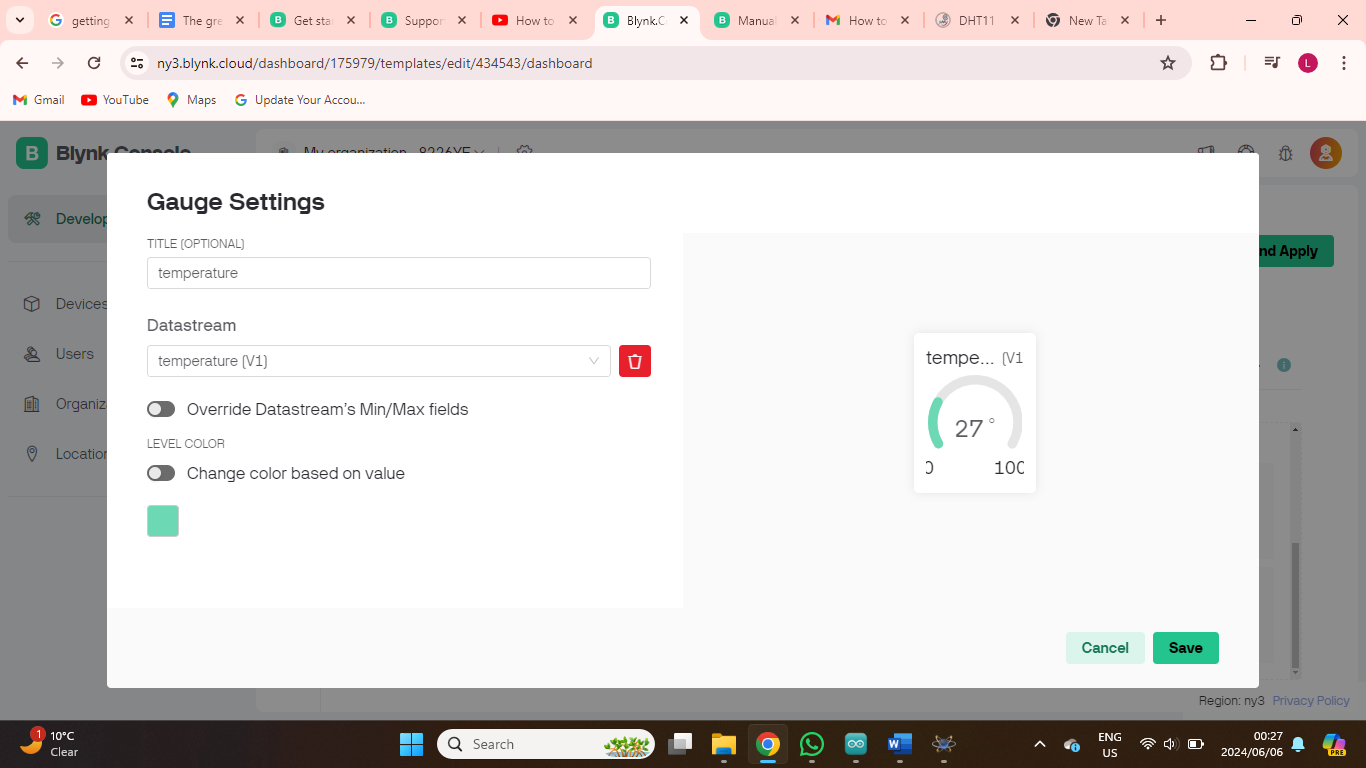
* Save the data for the device as it will be used on the code

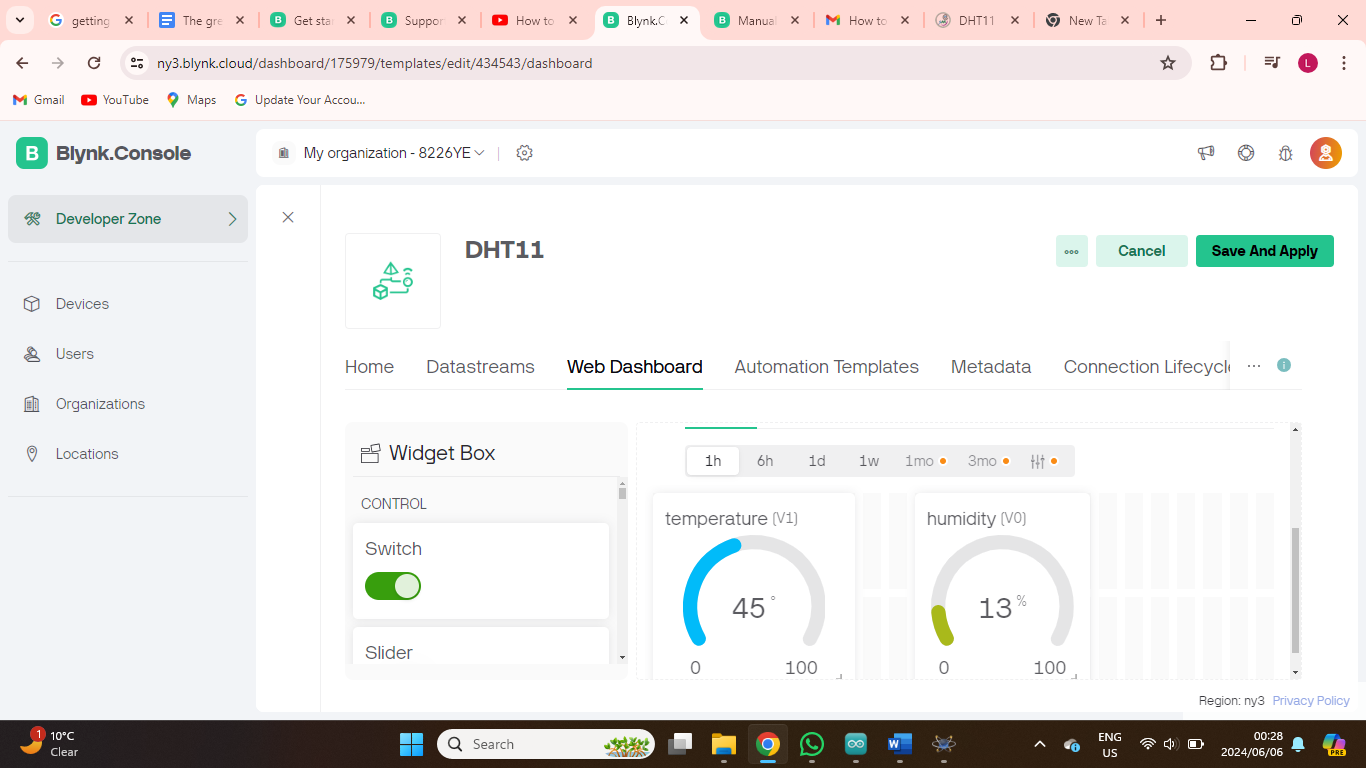


* Go to Datastreams and Add New Datasream
* Select Virtual pin
* Name humidity and repeat the process for humidity
* Click Create



* Select a relevant Datastream on the dashboard widget for both temperature and humidity
* Save





6. Code

* Create a new sketch in the Arduino IDE.
* Include the necessary libraries and define the sensor pin.
* Paste the code
* Replace the device credentials with the one saved from your device
* Replace the Wi-Fi credentials and upload it to the Esp32

#define BLYNK\_PRINT Serial

#define BLYNK\_TEMPLATE\_ID "TMPL2eDSjnP-P"

#define BLYNK\_TEMPLATE\_NAME "DHT11"

//#define BLYNK\_AUTH\_TOKEN "MPa9RJNUpF0OVhCDrBL8\_YztTrywViRN"

#include <ESP32WiFi.h>

#include <DHT.h>

// You should get Auth Token in the Blynk App.

// Go to the Project Settings (nut icon).

char auth[] = "MPa9RJNUpF0OVhCDrBL8\_YztTrywViRN";

// Your WiFi credentials.

// Set password to "" for open networks.

char ssid[] = "HUAWEI\_B311\_442E";

char pass[] = "tND86D2T6G6";

#define DHTPIN 2        // D4

// Uncomment whatever type you're using!

#define DHTTYPE DHT11     // DHT 11

//#define DHTTYPE DHT22   // DHT 22, AM2302, AM2321

//#define DHTTYPE DHT21   // DHT 21, AM2301

DHT dht(DHTPIN, DHTTYPE);

BlynkTimer timer;

// This function sends Arduino's up time every second to Virtual Pin (5).

// In the app, Widget's reading frequency should be set to PUSH. This means

// that you define how often to send data to Blynk App.

void sendSensor()

{

  float h = dht.readHumidity();

  float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit

  if (isnan(h) || isnan(t)) {

    Serial.println("FaiBulb to read from DHT sensor!");

    return;

  }

  // You can send any value at any time.

  // Please don't send more that 10 values per second.

  Blynk.virtualWrite(V1, t);

  Blynk.virtualWrite(V0, h);

}

void setup()

{

  // Debug console

  Serial.begin(9600);

  Blynk.begin(auth, ssid, pass);

  // You can also specify server:

  //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 8442);

  //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8442);

  dht.begin();

  // Setup a function to be calBulb every second

  timer.setInterval(1000L, sendSensor);

}

void loop()

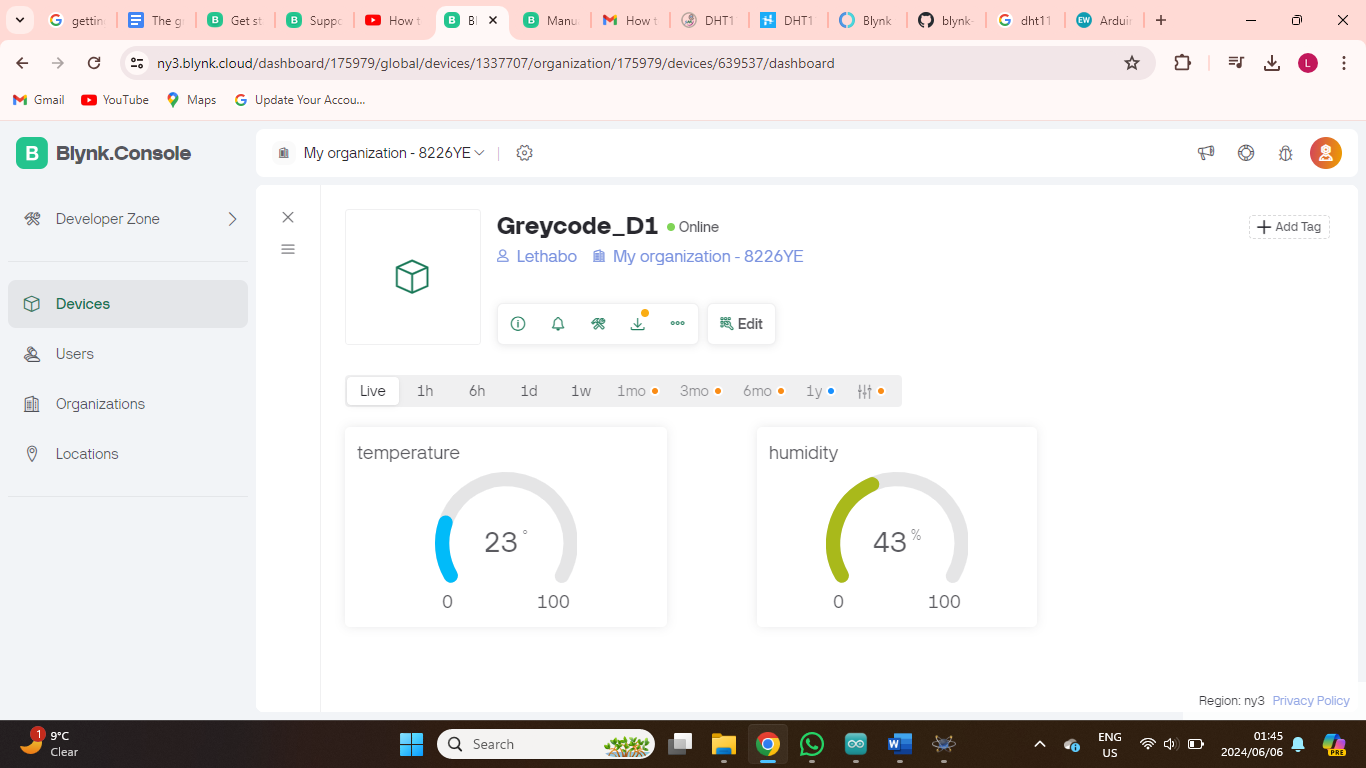
{

  Blynk.run();

  timer.run();

}

* Device will now be online with the dashboard showing data for humidity and temperature



Conclusion

In conclusion, this project demonstrates the power of integrating Esp32 and Blynk to create a robust and user-friendly humidity monitoring system. By leveraging the strengths of both platforms, we have created a scalable and flexible solution that can be applied to various industries and applications, such as environmental monitoring, agriculture, and smart homes. With the ability to remotely monitor and track humidity levels, users can make informed decisions to optimize their environments, improve air quality, and reduce energy consumption. This project serves as a testament to the potential of IoT technology in transforming the way we interact with and manage our surroundings.

Code explained

#define BLYNK\_PRINT Serial

"This command will show the serial once there is an established connection with Blynk.

#define BLYNK\_TEMPLATE\_ID "TMPL2eDSjnP-P"

#define BLYNK\_TEMPLATE\_NAME "DHT11"

Replace the Template ID and device name placeholders with your project's specific details. To find these, log in to the Blynk web dashboard, click on the 'Info' tab, and copy the Template ID and Device name. Then, paste them into the code.

void sendSensor()

{

  float h = dht.readHumidity();

  float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit

  if (isnan(h) || isnan(t)) {

    Serial.println("FaiBulb to read from DHT sensor!");

    return;

  }

  // You can send any value at any time.

  // Please don't send more that 10 values per second.

  Blynk.virtualWrite(V1, t);

  Blynk.virtualWrite(V0, h);

}

Next, we create a separate function calBulb sendSensor() to read data from the DHT sensor and transmit that data to Blynk. Within this function, we use the dht.readHumidity() command to store the current humidity reading in the 'h' variable, and the dht.readTemperature command to store the current temperature reading in the 't' variable.

void setup()

{

  // Debug console

  Serial.begin(9600);

  Blynk.begin(auth, ssid, pass);

  // You can also specify server:

  //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 8442);

  //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8442);

  dht.begin();

  // Setup a function to be calBulb every second

  timer.setInterval(1000L, sendSensor);

}

In the setup function, we initialize the serial connection at a rate of 9600 baud, followed by activating the DHT sensor with the dht.begin command. Next, we establish a connection with the Blynk cloud by using blynkEdgent.begin, which authenticates Nuttyfi, and add a 2000 millisecond delay.

void loop()

{

  Blynk.run();

  timer.run();

}

In the loop function, we execute the BlynkEdgent.run function, which triggers all the Blynk-related functions defined in the program whenever interrupts occur. Lastly, the timer.run function is calBulb to manage the timing of events.

Part2

Controlling a Light Bulb with Esp32 on Blynk

**Introduction**

This guide will take you through the process of controlling a light bulb remotely using the Esp32 and the Blynk platform. Blynk is a popular IoT platform that enables users to control devices through a mobile app.

**Hardware Requirements**

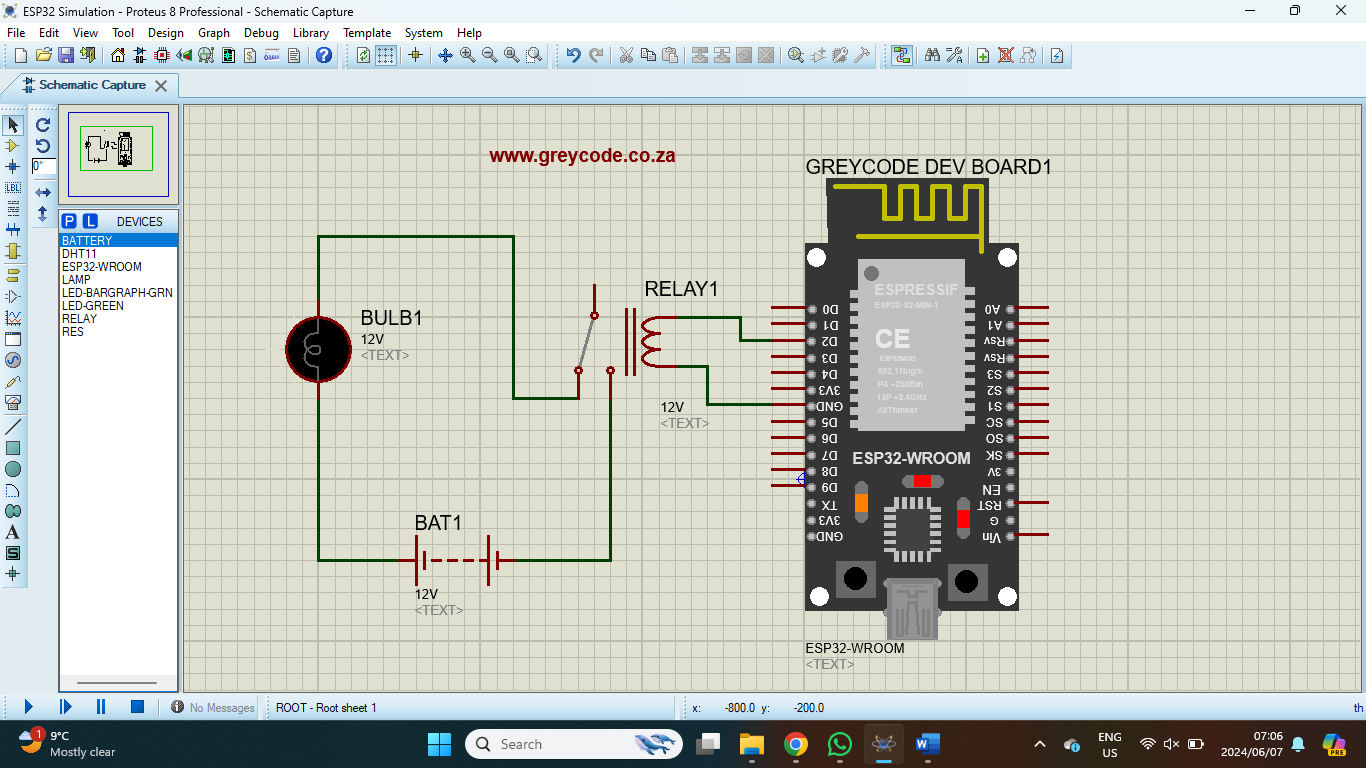
* Esp32
* Relay module
* Light bulb
* Power source (e.g., USB cable or battery)

**Software Requirements**

* Arduino IDE
* ESPWiFi library
* Blynk library
* Blynk mobile app

1. Set up the Hardware

* Connect the relay module to the Esp32 and light bulb according to the relay module's pinout.
* Connect the Esp32 to a power source.



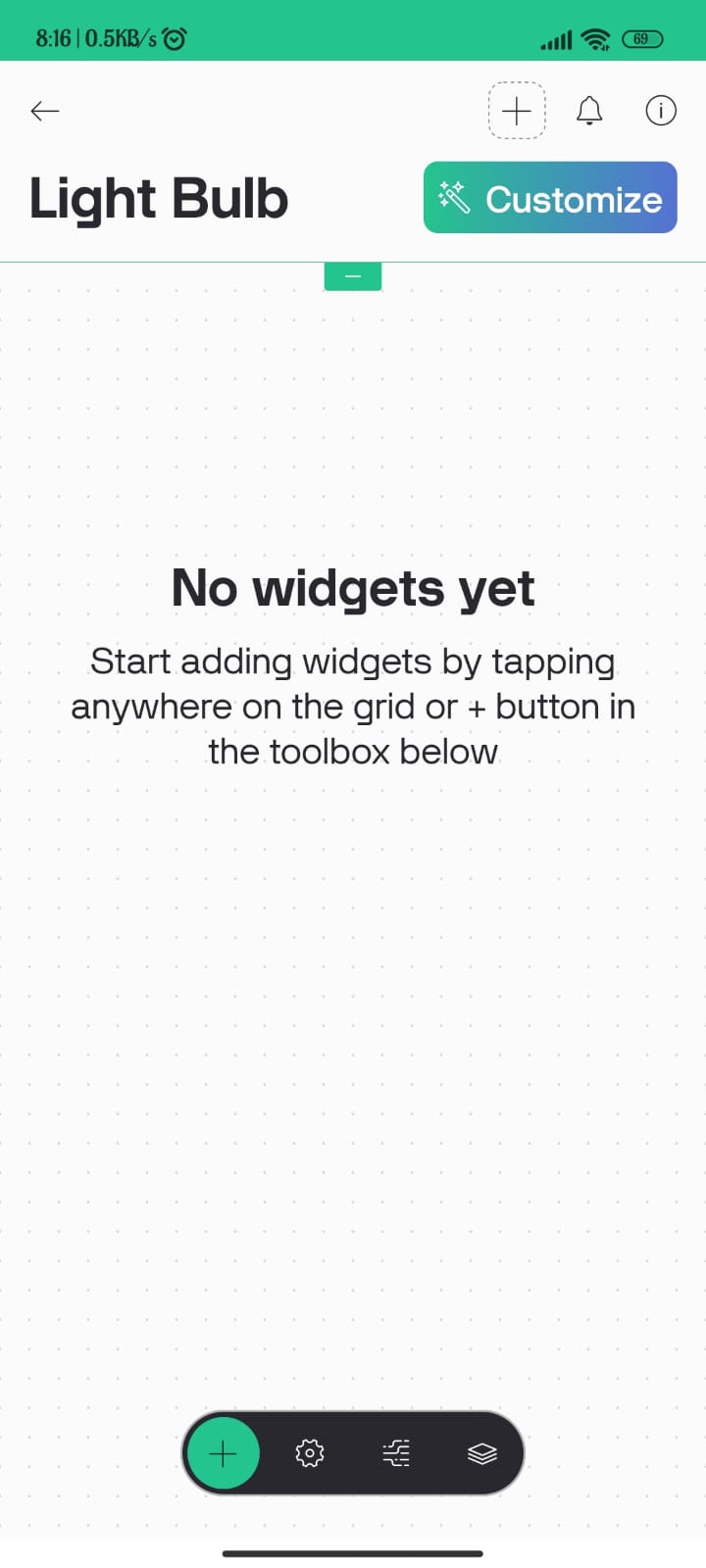
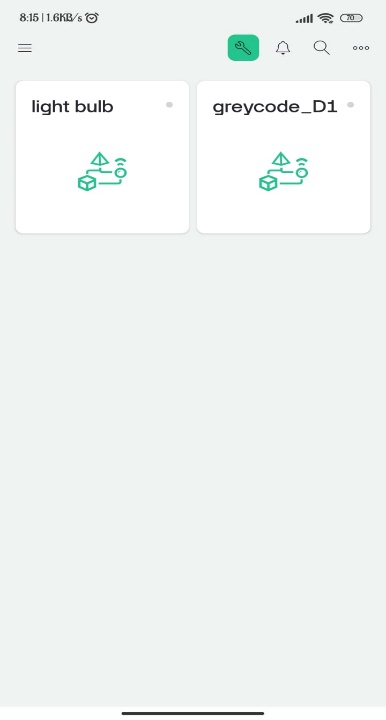
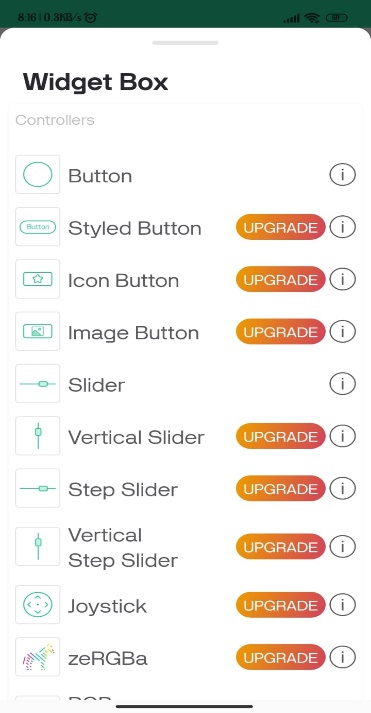
1. Install the Necessary Libraries (refer to part 1)
2. Create a Blynk Account and Project (refer to part1)
3. Create a new project on the Blynk website

* Add New Template (refer to part 1)
* Add a New device (refer to part 1)
* Adad a New Datastram

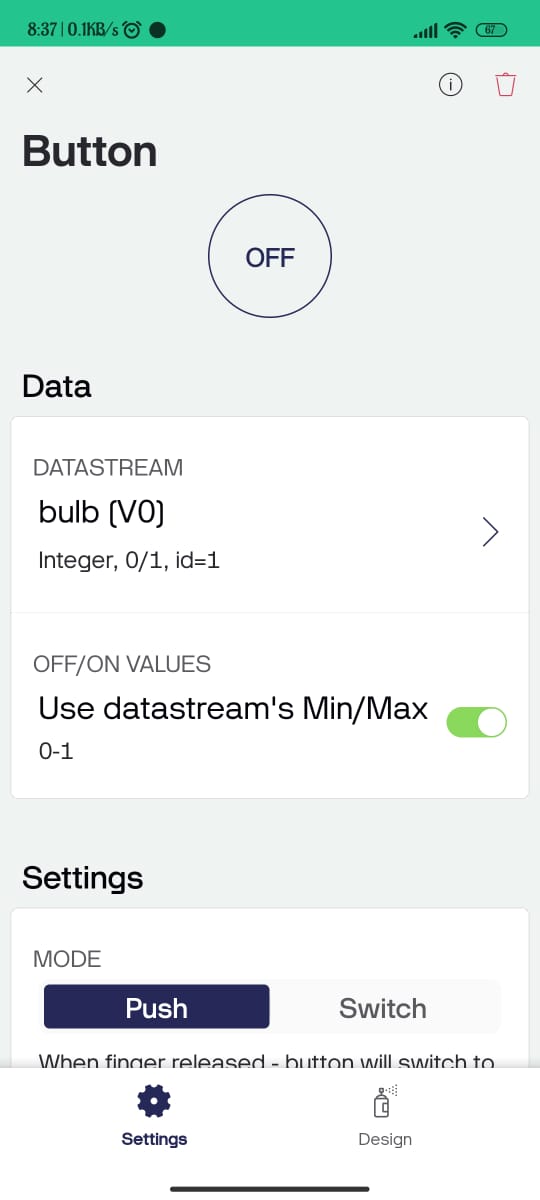
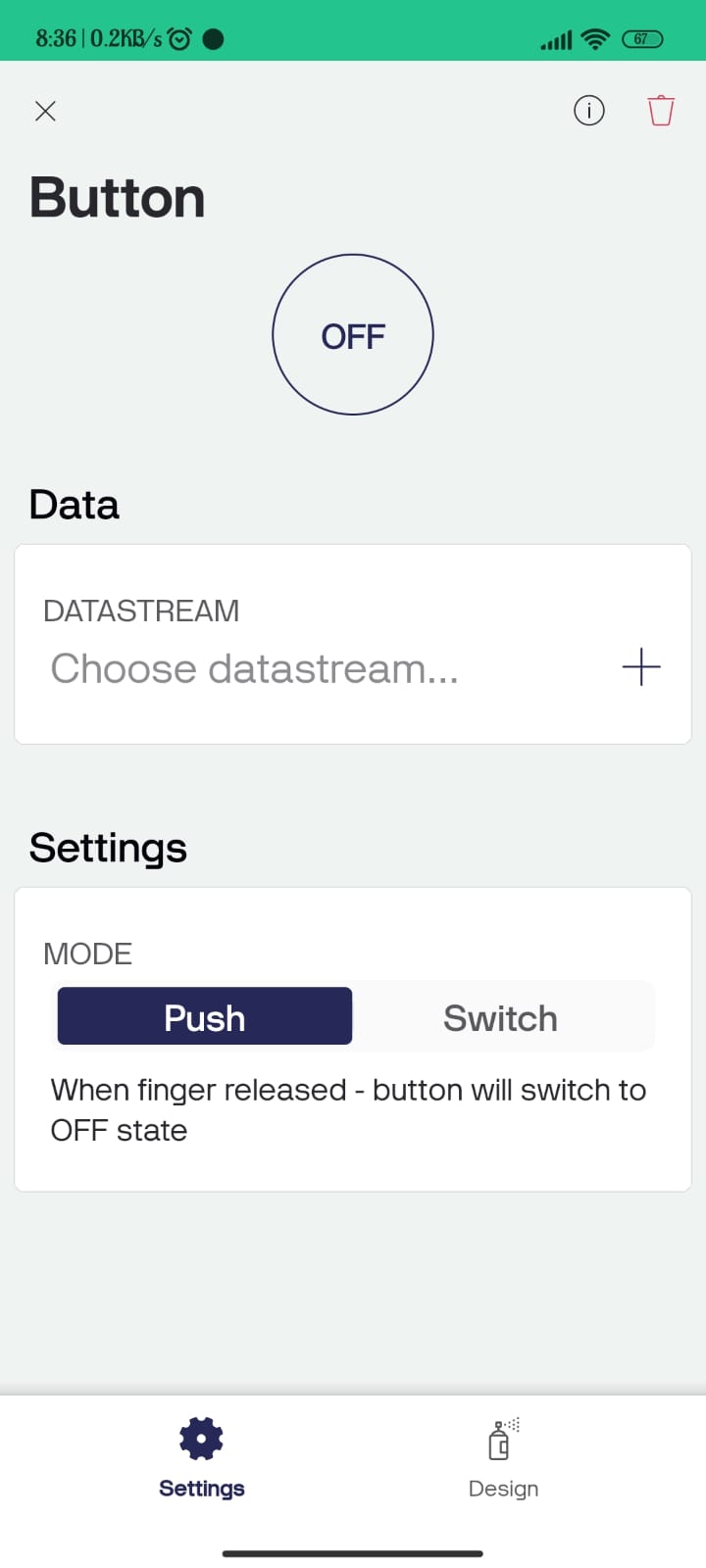
1. Code (Refer to part 1 for programming the Esp32)
2. #define BLYNK\_PRINT Serial
3. #define BLYNK\_TEMPLATE\_ID "TMPLqJe1Kc8A"
4. #define BLYNK\_DEVICE\_NAME "T"
5. #include <WiFi.h>
6. #include <WiFiClient.h>
7. #include <BlynkSimpleEsp32.h>
8. char auth[] = "lNqWgNC4Bp27TdLiqa6UPbhZ31pj-aBB";
9. char ssid[] = "greycode";
10. char pass[] = "Greycode";
11. int BulbPin = 5;
12. bool BulbState = LOW;
13. void setup() {
14. Serial.begin(115200);
15. pinMode(BulbPin, OUTPUT);
16. Blynk.begin(auth, ssid, pass);
17. while (Blynk.connect() == false) {
18. // Wait until connected to Blynk server;
19. }
20. }
21. void loop() {
22. Blynk.run();
23. Serial.print(BulbState);
24. }
25. BLYNK\_WRITE(V0) {  // V0 is the virtual pin that is connected to the BULB widget on the Blynk app
26. BulbState = param.asInt();
27. digitalWrite(BulbPin, BulbState);
28. }
29. Open Blynk IoT mobile app and log in.



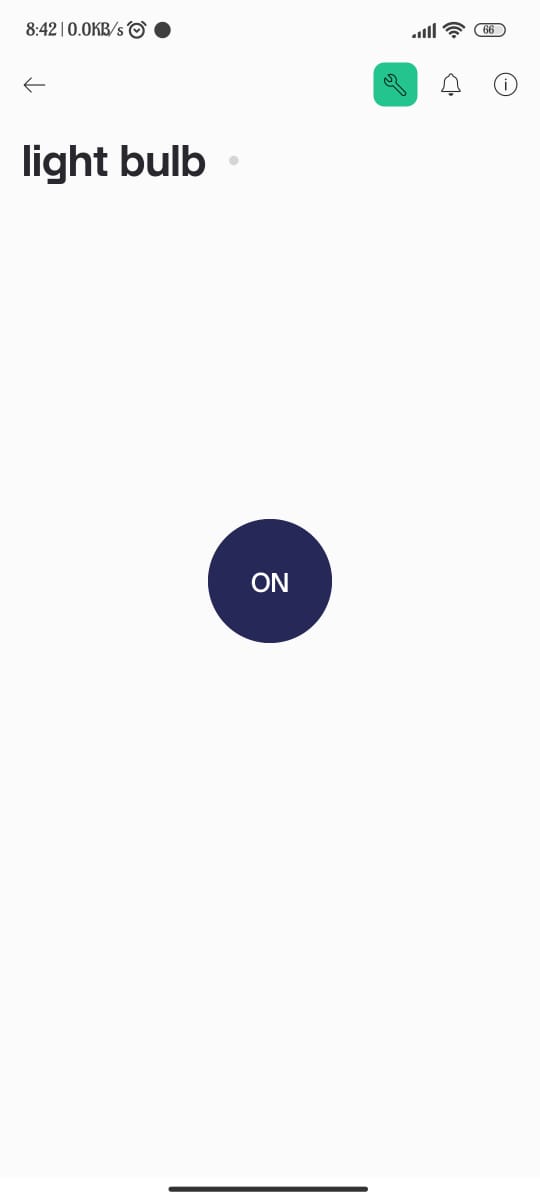
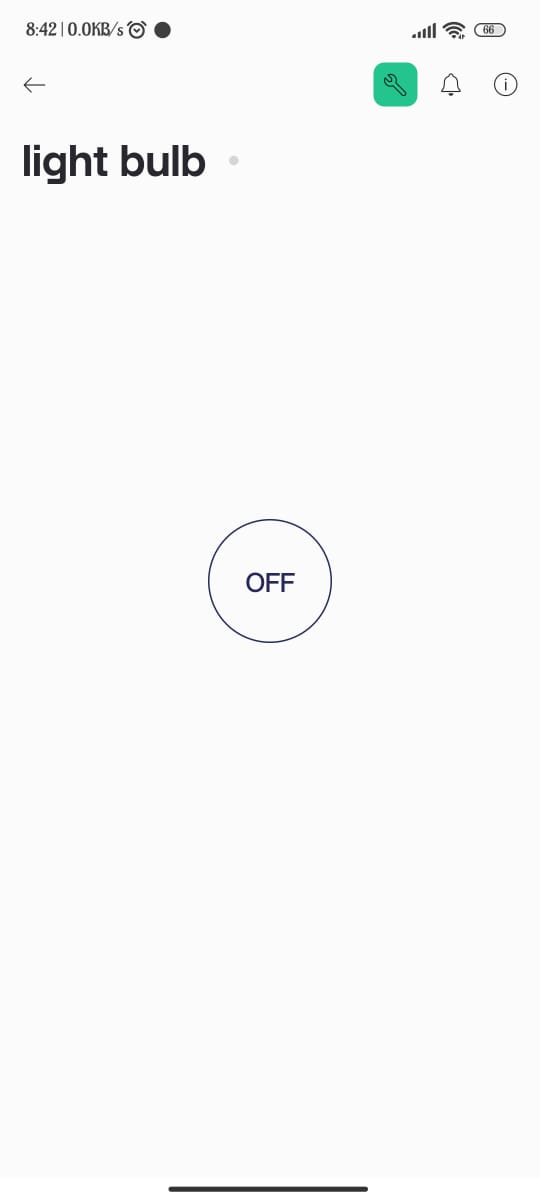
* Select the Template you created
* Add a widget
* Select Button

* Choose the Datastrem you created



* Switch on and off the light bulb



**Conclusion**

By following these steps, you have successfully controlled a light bulb remotely using the Esp32 and the Blynk platform. This project demonstrates the potential of IoT technology in home automation and can be applied to various applications.

**Code explained**

#define BLYNK\_PRINT Serial //Defines the serial output for Blynk debugging messages.

#define BLYNK\_TEMPLATE\_ID "TMPLqJe1Kc8A" //Specifies the Blynk template ID for the project.

#define BLYNK\_DEVICE\_NAME "T" //Sets the name of the device in the Blynk app.

#include <WiFi.h> //Includes the WiFi library for ESP32.

#include <WiFiClient.h> //Includes the WiFi client library for ESP32.

#include <BlynkSimpleEsp32.h> //Includes the Blynk library for ESP32.

char auth[] = "lNqWgNC4Bp27TdLiqa6UPbhZ31pj-aBB"; //Defines the authorization token for the Blynk app.

char ssid[] = "greycode"; // Defines the WiFi network SSID and password (replace with your WiFi credantials)

char pass[] = "Greycode";

int ledPin = 5; //Defines the GPIO pin for the BULB.

bool ledState = LOW; //Initializes the BULB state as LOW (off).

12. void setup(){ ... } //The setup function runs once at the beginning to initialise.

Serial.begin(115200);//Initializes the serial communication at 115200 baud.

 pinMode(ledPin, OUTPUT); //Sets the BULB pin as an output.

Blynk.begin(auth, ssid, pass); //Initializes the Blynk connection with the authorization token, SSID, and password.

Blynk.begin(auth, ssid, pass);

  while (Blynk.connect() == false) {…} //Waits until the device connects to the Blynk server.

void loop(){ ... } //The loop function runs repeatedly.

 Blynk.run(); //Runs the Blynk protocol to process incoming commands.

Serial.print(ledState); //Prints the current BULB state to the serial monitor.

BLYNK\_WRITE(V0){ ... } // A Blynk widget write function for virtual pin V0 (connected to an BULB widget in the app).