National Textile University, Faisalabad

Department of Computer Science

Name:	Hiba Fatima
Class:	BSCS 5th _A
Registration No:	23-NTU-CS-1037
Course Name:	Embedded IoT Systems
Submitted To:	Sir Nasir Mahmood
Submission Date:	20/12/2025

ESP32 Webserver (webserver.cpp)

SHORT QUESTONS

1. What is the purpose of Webserver server (80); and what does port 80 represent?

Webserver server (80); // creates an instance of a web server object that listens for incoming HTTP requests on the ESP32 device.

Port 80 is the default port number for HTTP communication. This is because if the client, such as the web browser, communicates with the ESP32 using the IP address, the client transfers data using port 80. However, if another port number is specified, the client transfers data using the specified port number.

In essence: It configures the ESP32 to run web pages using the standard HTTP port.

2. Explain the role of server. On ("/", handle Root); in this program

server. On () sets a route on the web server.

"/" refers to the root URL (http://192.168.1)

Handle Root is the function that will be called each time the server receives a request for the given route from the client.

This is what tells the server what to do when a user accesses the home page of the ESP32 web server.

3. Why is server. handle Client (); placed inside the loop () function? What will happen if it is removed?

server. Handle Client (): This function checks if any client has sent any request. Then it is called the function associated with it (handle Root).

If it is removed:

It will ignore all requests from the client. The web page does not load because the server is not handling incoming requests.

4. In handle Root (), explain the statement: server. Send (200, "text/html", html);

200 → HTTP status code 200 OK, which means that the operation was successful.

"text/html" → The content type that is being transmitted; it is an HTML page.

html → The actual HTML content that will be rendered on the client browser.

This line sends the full HTTP response containing the webpage data back to the client.

5. What is the difference between displaying last measured sensor values and taking a fresh DHT reading inside handle Root ()?

Showing last measured values:

Showing previously stored sensor readings. This is quick and prevents triggering a sensor on page load.

Fresh read:

Reads the sensor in real time when the page loads. It provides the latest data but may result in a delay in the response and may not work if the sensor is busy or not ready. This means that using cache values is quicker and more accurate, and real-time results can also be achieved with fresh values.

LONG QUESTIONS

Q :Describe the complete working of the ESP32 webserver-based temperature and humidity : monitoring system. Working on ESP32 Webserver-Based Temperature and Humidity Monitoring System ?

1. ESP32 Wi-Fi Connection and IP Address Assignment First, the ESP32 connects to a Wi-Fi network using SSID and password. Once connected, the router assigns an IP address to the ESP32, which is used to open the web page in a browser.

2. Web Server Initialization and Request Handling After Wi-Fi connection, a web server is started on port 80 using the Webserver library. When a user opens the ESP32 IP address in a browser, the server receives the request and sends the required webpage as a response.

3. Button-Based Sensor Reading and OLED Update Mechanism A push button is used to control sensor readings. When the button is pressed, the ESP32 reads new temperature and humidity values from the DHT sensor and updates them on the OLED display.

4. Dynamic HTML Webpage Generation the ESP32 creates an HTML page in the program using strings. Sensor values are inserted into the HTML code, so the webpage shows live temperature and humidity data instead of fixed text.

5. Purpose of Meta Refresh in the Webpage The meta refresh tag is used to automatically reload the webpage after a few seconds. This helps display updated sensor values without manually refreshing the browser.

6. Common Issues in ESP32 Webserver Projects and Their Solutions Wi-Fi not connecting: Check SSID, password, and network range. Webpage not loading: Ensure correct IP address and port 80 is used. Sensor gives Nan values: Check DHT sensor wiring and add a delay between readings OLED not updating: Confirm correct I2C address and proper library installation.

Blynk Cloud Interfacing (blynk.cpp)

Part-A: Short Questions

1. What is the role of Blynk Template ID in an ESP32 IoT project? Why must it match the cloud template?

Function: The Blynk Template ID defines the exact Blynk Template that has been designed on Blynk Cloud. It helps to identify which pre-designed layout, widgets, data types, and virtual pin configurations the ESP32 should abide by.

Why it should be matched: If the Template ID in coding is not matched with the cloud template, there would not be correct communication between ESP32 and cloud because ESP32 would not be aware of what layout should be followed in widgets and which virtual pins should be used.

Briefly: The Template ID connects the hard device to the right cloud template.

2. Differentiate between Blynk Template ID and Blynk Auth Token.

Feature	Template ID	Auth Token
Purpose	Identifies the cloud template associated with the device	Authenticates the device to the Blynk Cloud
Scope	Shared across all devices using the same template	Unique for each device instance
Use	Defines widget structure, virtual pins, and data types	Grants permission for a specific ESP32 to send/receive data
Change	Rarely changes unless template changes	Can be regenerated if needed

3. Why does using DHT22 code with a DHT11 sensor produce incorrect readings? Mention one key difference between the two sensors.

Error of the reading: DHT22 and DHT11 operate in different timings and data transfer systems. DHT22 can detect larger ranges and provide more precise data transmission. Code from DHT22 cannot decode the data bits properly from DHT11 and will give an error of readings of the temperature and humidity.

Main difference:

- DHT11: 0 to 50°C with 1°C resolution.
- DHT22: -40°C to 80°C with 0.1°C resolution.

4. What are Virtual Pins in Blynk? Why are they preferred over physical GPIO pins for cloud communication?

Virtual Pins: These are software-defined pins within the Blynk system, which are totally cloud- and app-based. They are not linked to the ESP32 physical pins.

Benefits over physical pins:

Provide flexible data transfer from the ESP32 to the Blynk widget without the use of actual GPIO.

Allow multiple widgets to share a common data stream.

Make IoT projects easier because you don't have to rewire the hardware when you change widgets.

Point: Virtual pins enable software control, cloud flexibility, and ease of use.

5. What is the purpose of using Blynk Timer instead of delay () in ESP32 IoT applications?

Purpose: The purpose of Blynk Timer is to schedule tasks to be performed repeatedly without interrupting other tasks.

Using delay () prevents the entire program from proceeding, stopping the ESP32 from:

- Handling commands from Blynk
- Mobile Device or Wi-Fi/Cloud Communication

Blynk Timer enables smooth multitasking and interaction with the cloud in real time

Long Question

Q: Explain the complete workflow of interfacing ESP32 with Blynk Cloud to display temperature and humidity values. Your answer should include: - Creation of Blynk Template and Datastreams - Role of Template ID, Template Name, and Auth Token - Sensor configuration issues (DHT11 vs DHT22) - Sending data using Blynk.virtualWrite() - Common problems faced during configuration and their solutions.

Answer:

Workflow of Interfacing ESP32 with Blynk Cloud to Display Temperature and Humidity

1. Creating a Blynk Template and Datastreams

The process begins by creating a new template in the Blynk Cloud platform. Within this template, virtual pin datastreams are configured for temperature and humidity. These datastreams act as channels through which sensor data from the ESP32 is sent to the Blynk application.

2. Importance of Template ID, Template Name, and Auth Token

The Template ID uniquely links the ESP32 firmware to its corresponding project on the Blynk Cloud. The Template Name identifies the project in the Blynk app interface, while the Auth Token serves as a security key that allows the ESP32 to communicate securely with the cloud server.

3. Sensor Configuration Considerations (DHT11 vs DHT22)

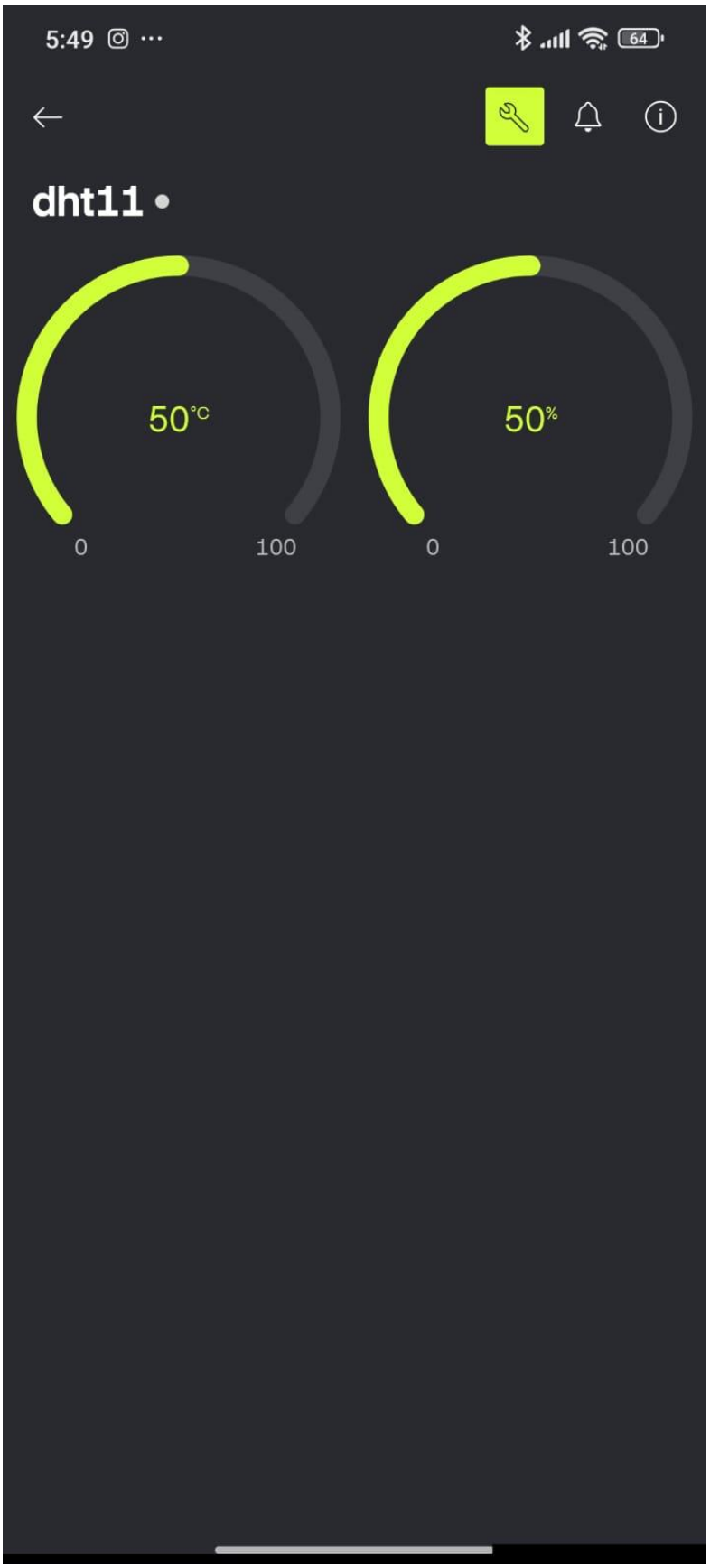
DHT11 and DHT22 sensors differ in accuracy, range, and data format. If the incorrect sensor type is defined in the code, the readings may be inaccurate or invalid. Therefore, it is essential to select the correct sensor model in the program according to the hardware used.

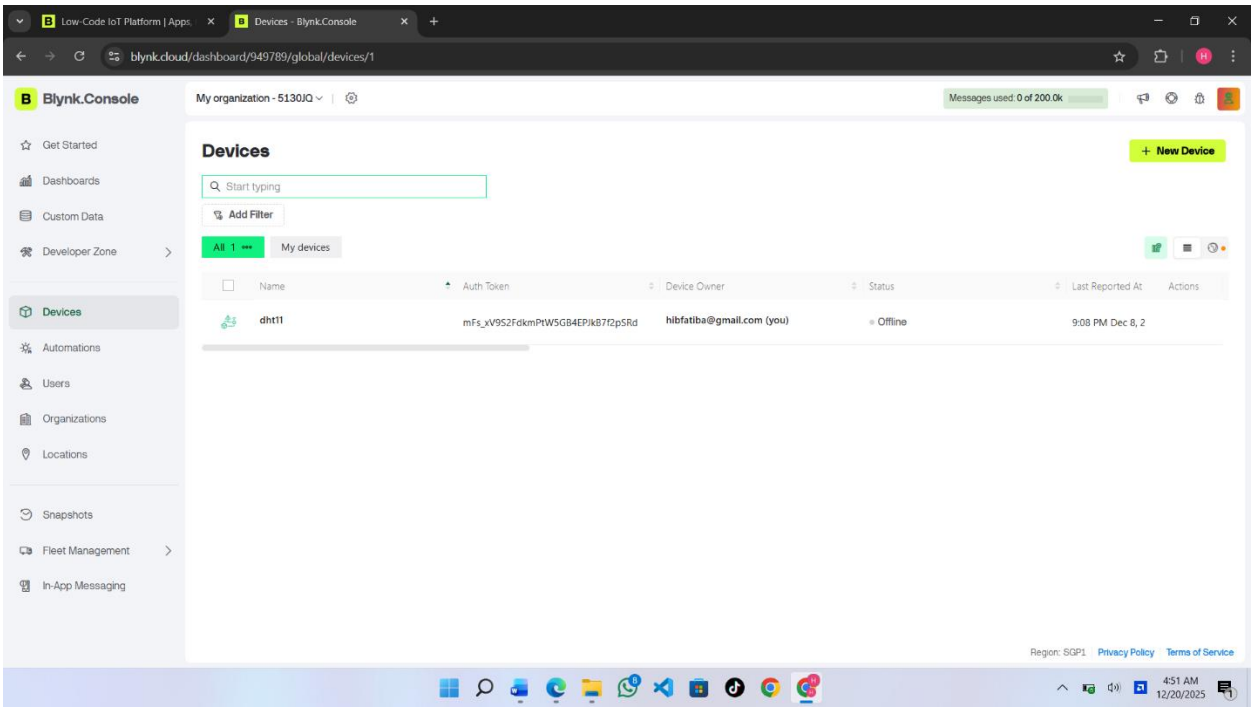
4. Transmitting Data Using Blynk.virtualWrite()

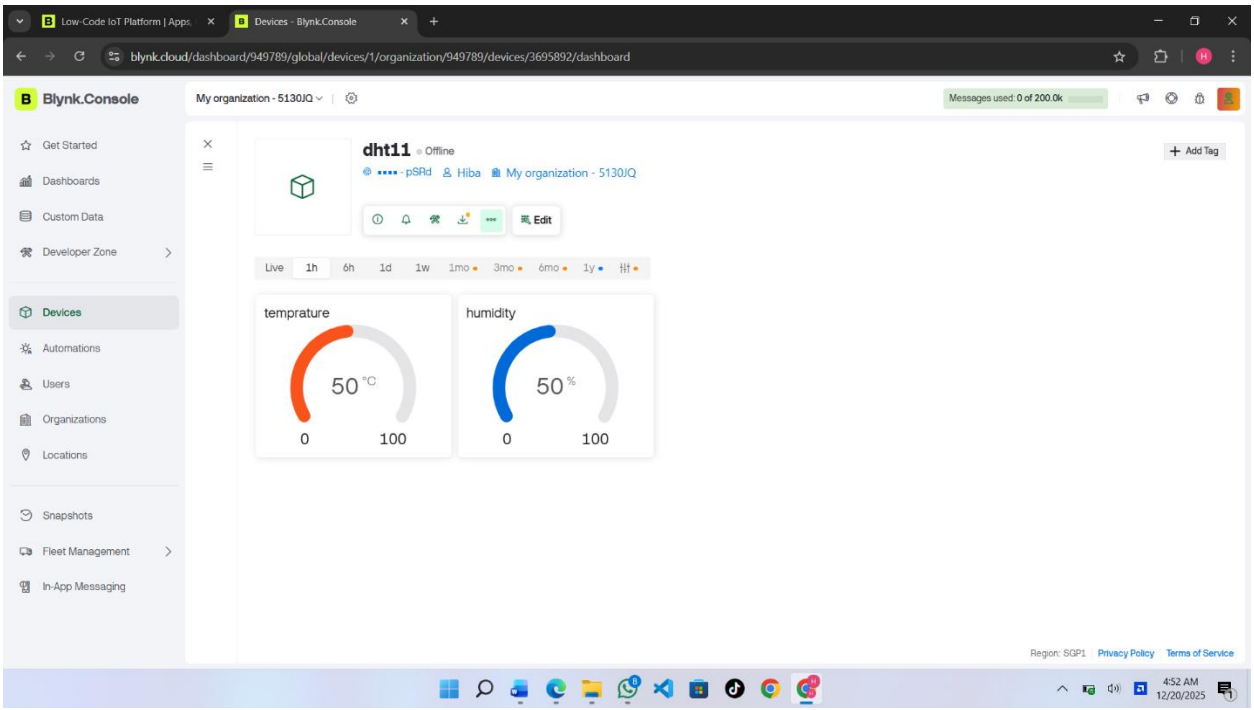
Once the ESP32 reads temperature and humidity values from the sensor, the data is sent to the Blynk Cloud using the Blynk.virtualWrite() function. These values are then displayed in real time on the Blynk dashboard widgets associated with the specified virtual pins.

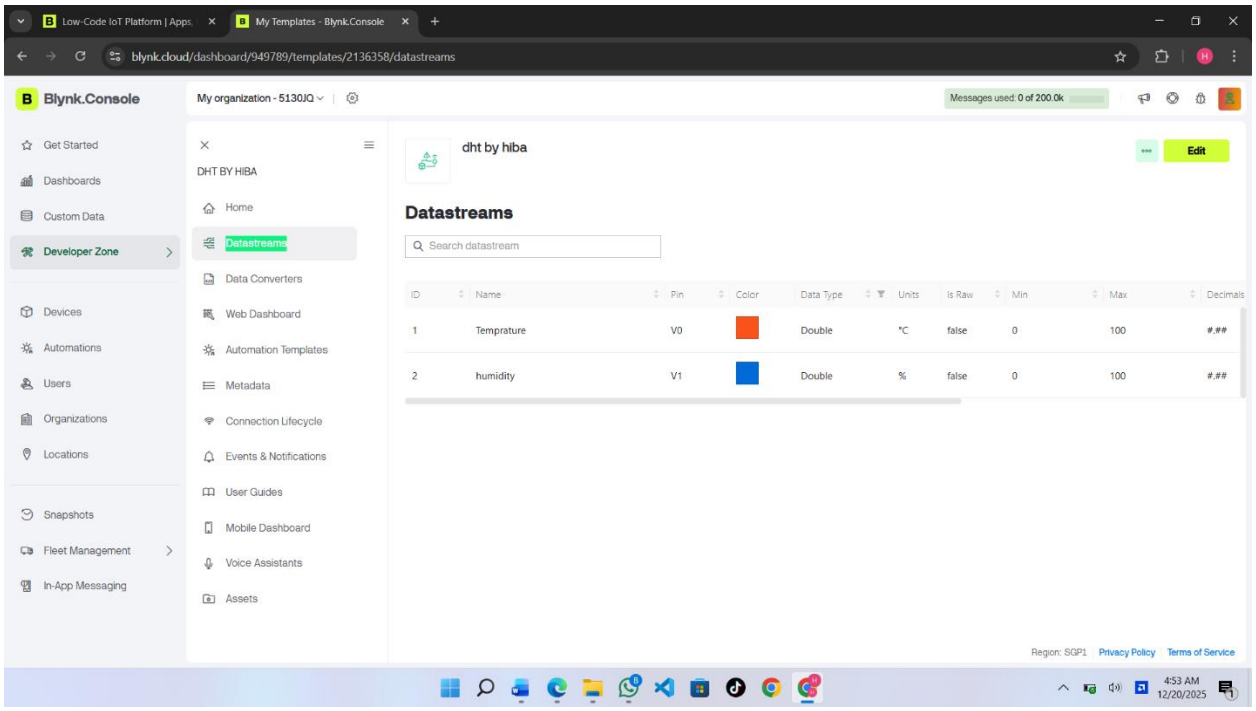
5. Common Issues and Their Solutions

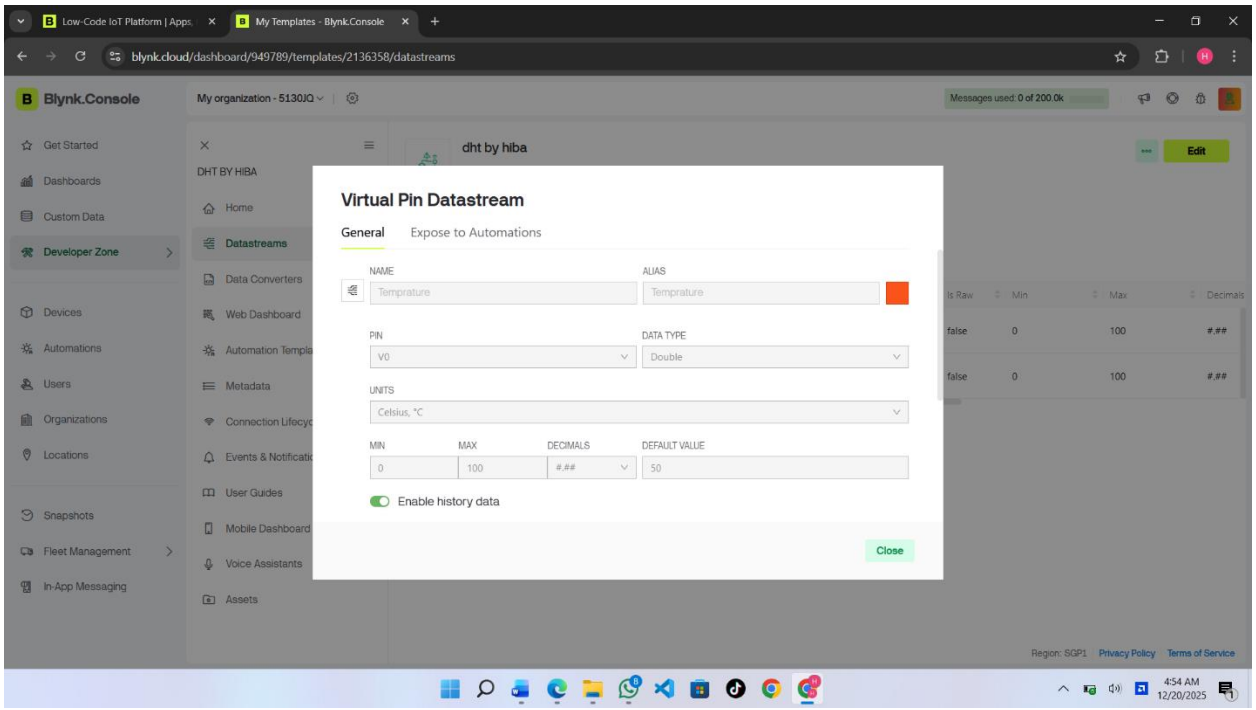
- **ESP32 fails to connect to Blynk Cloud:** Verify Wi-Fi credentials, Template ID, and Auth Token.
- **Sensor values not visible in the app:** Confirm that the virtual pin numbers match the configured datastreams.
- **Incorrect sensor data:** Check sensor wiring, confirm the correct sensor type, and include appropriate timing delays.
- **Frequent disconnections:** Avoid using delay() in the loop and implement BlynkTimer for better stability.

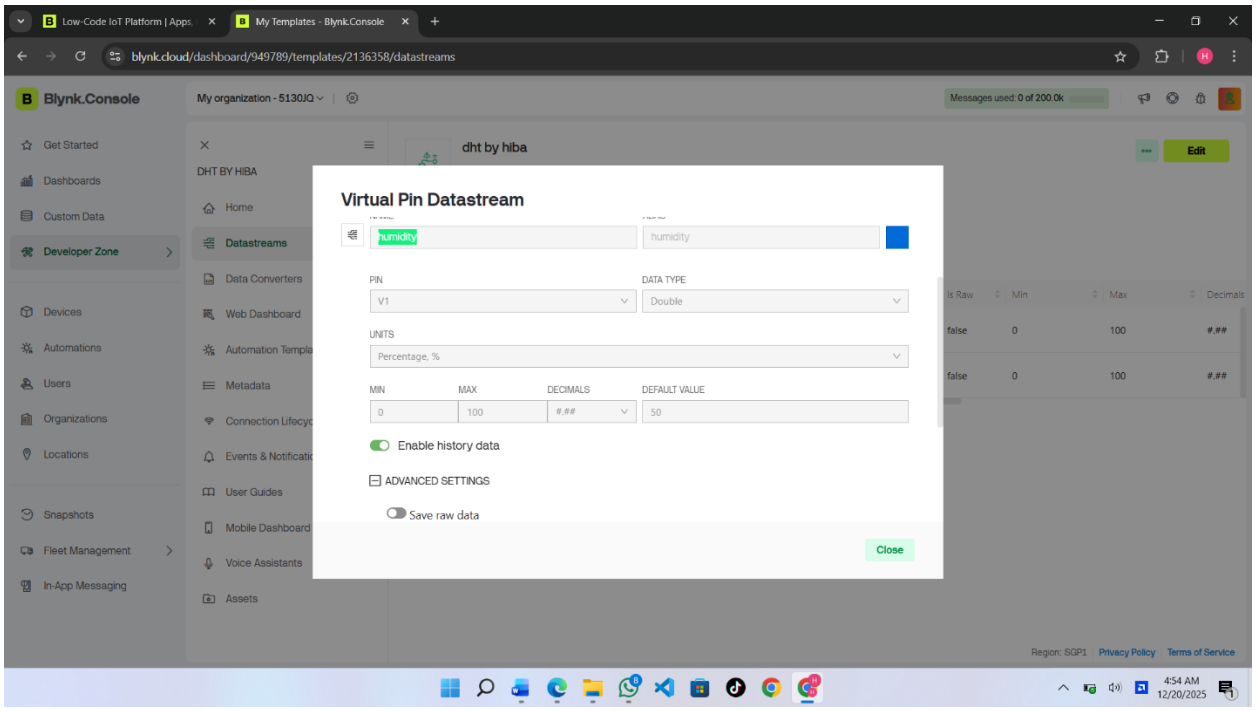

5:49
dht11
50°C
50%
0
100
0
100


Low-Code IoT Platform | Apps
Devices - Blynk.Console
blynk.cloud/dashboard/949789/global/devices/1
Blynk.Console
My organization - 5130JQ
Messages used: 0 of 200.0k
Get Started
Dashboards
Custom Data
Developer Zone
Devices
Automations
Users
Organizations
Locations
Snapshots
Fleet Management
In-App Messaging
Devices
New Device
Start typing
Add Filter
All 1
My devices
Name
Auth Token
Device Owner
Status
Last Reported At
Actions
dht11
mFs_xV9S2FdkmPtW5GB4EPJkB7f2pSRd
hibfatiba@gmail.com (you)
Offline
9:08 PM Dec 8, 2
Region: SGP1
Privacy Policy
Terms of Service
4:51 AM
12/20/2025


Low-Code IoT Platform | Apps
Devices - Blynk.Console
blynk.cloud/dashboard/949789/global/devices/1/organization/949789/devices/3695892/dashboard
Blynk.Console
My organization - 5130JQ
Messages used: 0 of 200.0k
Get Started
Dashboards
Custom Data
Developer Zone
Devices
Automations
Users
Organizations
Locations
Snapshots
Fleet Management
In-App Messaging
dht11
Offline
pSRd
Hiba
My organization - 5130JQ
Add Tag
Edit
Live
1h
6h
1d
1w
1mo
3mo
6mo
1y
temprature
50 °C
0
100
humidity
50 %
0
100
Region: SGP1
Privacy Policy
Terms of Service
4:52 AM
12/20/2025


Low-Code IoT Platform | Apps
My Templates - Blynk.Console
blynk.cloud/dashboard/949789/templates/2136358/datastreams
Blynk.Console
My organization - 5130JQ
Messages used: 0 of 200.0k
Get Started
Dashboards
Custom Data
Developer Zone
Devices
Automations
Users
Organizations
Locations
Snapshots
Fleet Management
In-App Messaging
DHT BY HIBA
Home
Datastreams
Data Converters
Web Dashboard
Automation Templates
Metadata
Connection Lifecycle
Events & Notifications
User Guides
Mobile Dashboard
Voice Assistants
Assets
dht by hiba
Edit
Datastreams
Search datastream
ID
Name
Pin
Color
Data Type
Units
Is Raw
Min
Max
Decimals
1
Temprature
V0
Double
°C
false
0
100
#.##
2
humidity
V1
Double
%
false
0
100
#.##
Region: SGP1
Privacy Policy
Terms of Service
4:53 AM
12/20/2025


Low-Code IoT Platform | Apps
My Templates - Blynk.Console
blynk.cloud/dashboard/949789/templates/2136358/datastreams
Blynk.Console
My organization - 5130JQ
Messages used: 0 of 200.0k
DHT BY HIBA
dht by hiba
Edit
Virtual Pin Datastream
General
Expose to Automations
NAME
Temprature
ALIAS
Temprature
PIN
V0
DATA TYPE
Double
UNITS
Celsius, °C
MIN
0
MAX
100
DECIMALS
#.##
DEFAULT VALUE
50
Enable history data
Close
Region: SGP1
Privacy Policy
Terms of Service
4:54 AM
12/20/2025


Low-Code IoT Platform | Apps
My Templates - Blynk.Console
blynk.cloud/dashboard/949789/templates/2136358/datastreams
Blynk.Console
My organization - 5130JQ
Messages used: 0 of 200.0k
Get Started
Dashboards
Custom Data
Developer Zone
Devices
Automations
Users
Organizations
Locations
Snapshots
Fleet Management
In-App Messaging
DHT BY HIBA
Home
Datastreams
Data Converters
Web Dashboard
Automation Templa
Metadata
Connection Lifecyc
Events & Notificatic
User Guides
Mobile Dashboard
Voice Assistants
Assets
dht by hiba
Edit
Virtual Pin Datastream
humidity
humidity
PIN
V1
DATA TYPE
Double
UNITS
Percentage, %
MIN
0
MAX
100
DECIMALS
#.##
DEFAULT VALUE
50
Enable history data
ADVANCED SETTINGS
Save raw data
Close
Is Raw
Min
Max
Decimals
false
0
100
#.##
false
0
100
#.##
Region: SGP1
Privacy Policy
Terms of Service
4:54 AM
12/20/2025