

Hands On Session on

Monitoring Temperature, Humidity and LED State Using ESP32 and Blynk

Raghu Ramaswamy	Chief Architect – Edge Computing Robotics
Shiva G	Senior Technical Lead – IoT & Edge Computing
Pavan Kumar S	Senior Technical Lead – IoT & Edge Computing
Bhavatha Baliga	Engineer – IoT & Edge Computing
Hema Tanmai M	Engineer – IoT and Edge Computing

Table of Contents

1. Introduction	4
1.1. Objective	4
1.2. Key Takeaways by end of the lab session	4
1.3. Architecture	5
2. Hardware Components	7
2.1 Images of different components	7
2.1.1 ESP32	7
2.1.2 DHT11 Sensor for Temperature and Humidity	7
2.1.3 LED	8
2.1. Wiring the components	8
2.1.1. ESP32 to DHT11	9
2.1.2. ESP32 to LED	9
Account Creation and Setup for Blynk	9
2.2. Sign Up and Log in to Blynk	9
2.3. Create a New Template	10
2.4. Add Data streams for Temperature and Humidity	11
2.4.1. Adding Datastream	11
2.5. Create a Device	12
2.6. Adding device	12
2.7. Configure the Dashboard	13
3. Monitor Your Dashboard	16
4. How to Burn Code to ESP32 from Arduino IDE	16
4.1. Requirements	16
4.2. Installing Arduino IDE	17
4.3. Setting Up Arduino IDE for ESP32	17
4.4. Install ESP32 Board	17
4.5. Connecting ESP32 to Your Computer	17
4.6. Troubleshooting	18
4.7. Uploading Your Program	19
4.8. Enabling Events and Notifications	20
4.8.1. Steps to Enable Events and Notifications	20
4.8.2. Set Up Notifications	21
4.8.3. Apply and Save	22

5.	Step-by-Step Guide to Create a Dashboard in Blynk Mobile	23
5.1.	Sign Up and Log in to Blynk Mobile.....	23
5.2.	Find your Template	23
5.3.	Configure the Dashboard.....	23
6.	Monitor Your Dashboard	28
7.	Tasks	29
7.1	Push Heat Index Value to Blynk IoT Cloud.....	29
7.2	Trigger an event upon reaching a threshold of temperature/humidity/heat-index	29

1. Introduction

This documentation provides a comprehensive guide to connecting and configuring an ESP32 microcontroller with a DHT11 temperature and humidity sensor, a push button, and an LED. Additionally, it covers setting up a Blynk dashboard for real-time monitoring and control over the web as well as a mobile app.

1.1. Objective

The objective of this hands-on lab is to equip students with comprehensive skills and knowledge in the entire IoT ecosystem. Students will learn how to acquire real-time data, transmit and store this data, visualize it on various platforms (Web, Mobile ...etc), and implement a bidirectional command and control system, thereby gaining practical experience in developing and managing IoT solutions.

1.2. Key Takeaways by end of the lab session

- Grasp the components and architecture of the IoT ecosystem.
- Use sensors and devices, including the ESP32 microcontroller, to collect real-time environmental data.
- Transmit data to cloud using protocols like MQTT, HTTP, or CoAP.
- Process and store data on cloud platforms.
- Create visual data representations using different tools for various mediums like mobile, web.
- Implement systems for sending commands to IoT devices to control their actions.
- Gain hands-on experience programming the ESP32 microcontroller using development environments such as Arduino IDE or Platform IO.
- Understand and apply security measures, including encryption and authentication.

1.3. Architecture

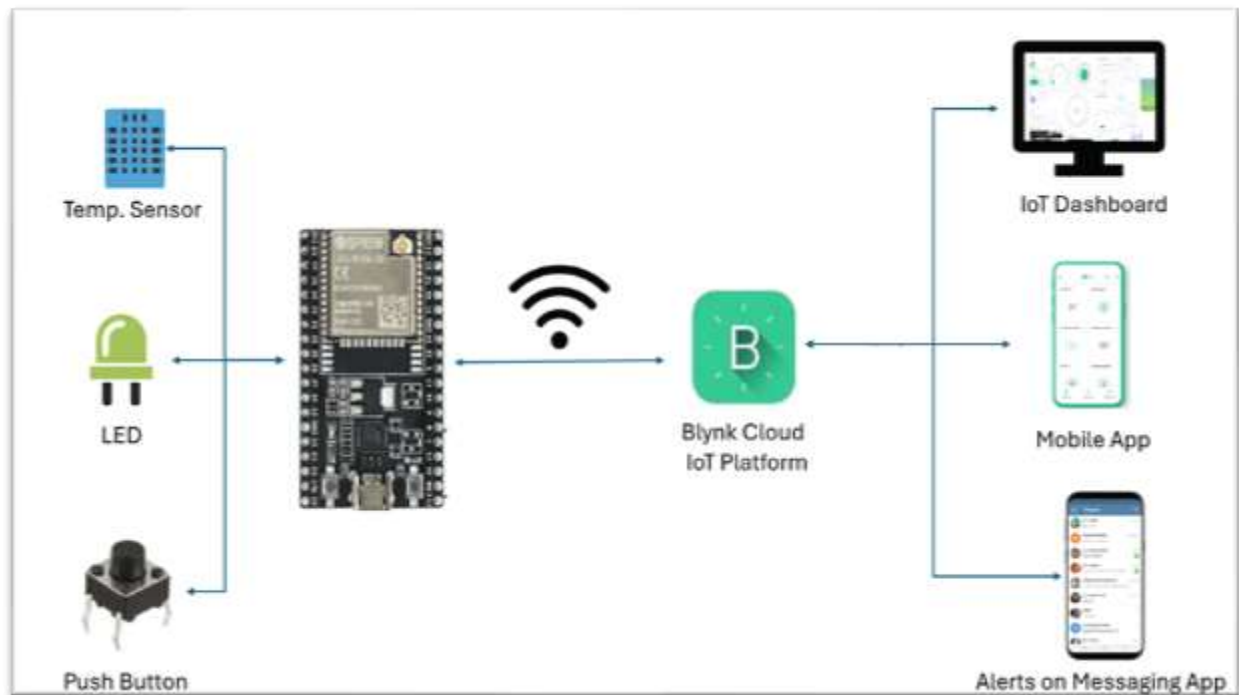


Fig: Architecture Diagram

The ESP32 is a powerful and versatile microcontroller developed by Espressif Systems. It is popular in the IoT (Internet of Things) community due to its robust features, low cost, and ease of use. Here are some key features and details about the ESP32:

1. Dual-Core Processor
2. Memory
3. Wireless Connectivity
4. I/O Ports
5. Peripheral Interfaces
6. Timers and Counters
7. Low Power Modes
8. Security Features
9. Development Environment
10. Community and Ecosystem

Let's delve into the protocol and process of how an ESP32 communicates with the Blynk Cloud using MQTT.

Understanding MQTT: MQTT (Message Queuing Telemetry Transport) is a lightweight messaging protocol that is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited. It's a publish/subscribe model, which makes it efficient for IoT (Internet of Things) devices like the ESP32.

Protocol Between ESP32 and Blynk Cloud

Connection Establishment: The ESP32 connects to a WiFi network. Once connected to WiFi, the ESP32 establishes a connection to the Blynk Cloud using MQTT over TCP/IP.

Authentication: The ESP32 authenticates itself to the Blynk Cloud using an Auth Token, which you receive when you create a project in the Blynk app.

Publishing and Subscribing: The ESP32 publishes data to specific topics that correspond to virtual pins in the Blynk Cloud. The ESP32 subscribes to topics to receive commands or data from the Blynk Cloud.

Message Format: Messages sent to the Blynk Cloud must follow a specific format that Blynk understands. Similarly, messages received from the Blynk Cloud will be in a format that the ESP32 must parse.

2. Hardware Components

S.No	Component	Count
1.	ESP32 Microcontroller	1
2.	DH11 Temperature and Humidity Sensor	1
3.	LED	1
4.	Jumper Wires	5

2.1 Images of different components

2.1.1 ESP32

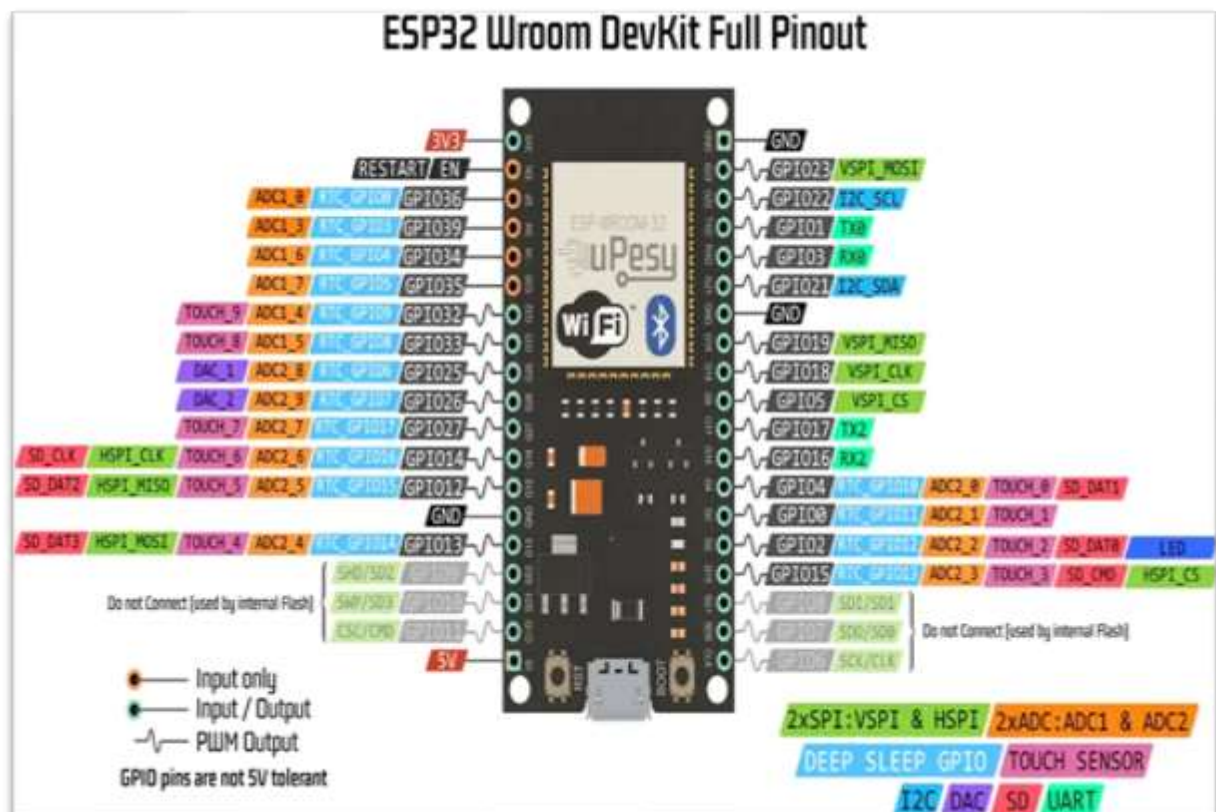


Fig: ESP32 Pin Diagram

2.1.2 DHT11 Sensor for Temperature and Humidity



Fig: DHT11 Sensor

2.1.3 LED



Fig: LED

2.2 Wiring the components

Below is the circuit diagram & explanation for each connection.

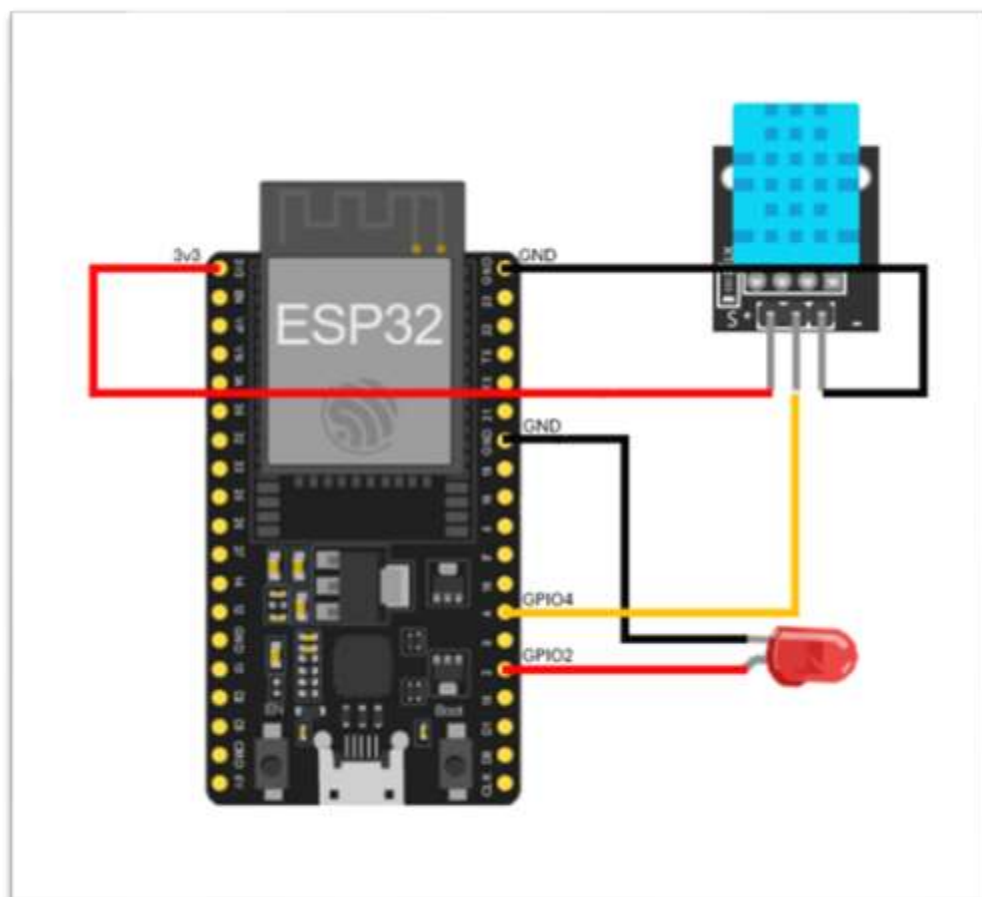


Fig: Circuit Diagram

2.2.1 ESP32 to DHT11

- Connect the VCC pin of DHT11 to 3.3V on the ESP32.
- Connect the GND pin of DHT11 to GND on the ESP32.
- Connect the Data pin of DHT11 to GPIO 4 on the ESP32.

2.2.2 ESP32 to LED

- Connect the anode (long leg) of the LED to GPIO 2 on the ESP32.
- Connect the cathode (short leg) of the LED to GND on the ESP32.

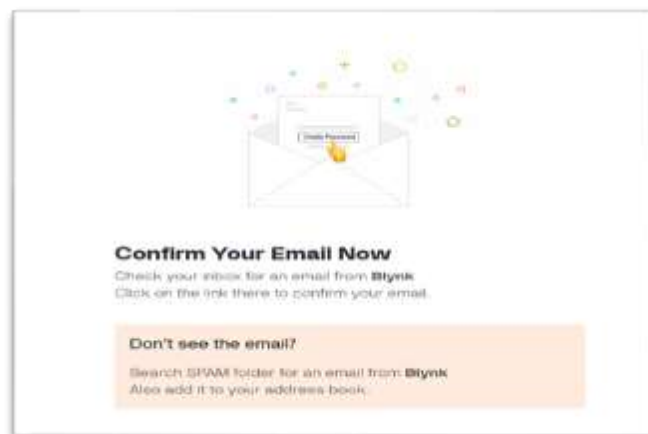
Now power up the ESP32 board by connecting it to power source using USB provided.

3. Account Creation and Setup for Blynk

Step by step guide to setup an account in Blynk and creation of dashboard

3.1 Sign Up and Log in to Blynk

- Go to <https://blynk.io>.
- If you don't have an account, sign up for a new one.
- After getting the below screen, check your email and create your password for the account.



- Log in to your Blynk account.

3.2 Create a New Template

In Blynk, templates are predefined configurations that include a set of widgets and settings, allowing you to quickly set up and manage IoT projects by applying consistent layouts and functionalities across multiple devices. They simplify the process of creating, deploying, and updating projects with a uniform design.

Adding templates:

- Navigate to the **Templates** section.
- Click on **+ New Template**.



- Fill in the details for your new template:
 - **Name:** e.g., " Monitor ESP32 with sensors"
 - **Hardware Type:** Select the appropriate hardware (e.g., ESP32).
 - **Connection Type:** Select the appropriate connection type (e.g., WiFi, Ethernet, etc.).

A screenshot of the 'Create New Template' form. It has a title 'Create New Template'. Below the title are three input fields: 'NAME' with the value 'Monitor ESP32 with sensors' and a character count '26 / 50'; 'HARDWARE' with a dropdown menu showing 'ESP32'; and 'CONNECTION TYPE' with a dropdown menu showing 'WiFi'. Below these is a 'DESCRIPTION' text area with the placeholder text 'Description' and a character count '0 / 128'. At the bottom right are two buttons: 'Cancel' and 'Done'.

3.3 Add Data streams for Temperature and Humidity

In Blynk, datastreams are channels for sending and receiving data between your devices and the Blynk platform, enabling real-time monitoring and control of various parameters. They facilitate the communication of sensor readings, control signals, and other data types.

3.3.1 Adding Datastream

- Inside the template, go to the **Datastreams** tab.
- Click on **+ New Datastream**.
 - **Datastream Type:** Virtual Pin.
 - **Name:** Temperature
 - **Virtual Pin:** V6 (or any other available pin).
 - **Data Type:** Double.
 - **Min/Max Values:** Set appropriate ranges, e.g., 0 to 100.
 - **Unit:** °C.
 - **Enable the Historic data toggle button.**



ADVANCED SETTINGS

- ☐ Save raw data UPGRADE
- ☐ Invalidate in secs then set ?
- ☐ Sync with latest server value every time device connects to the cloud ?
- ☒ Show in Custom Charts
- ☒ Show in Reports

Cancel Save

- Repeat the process for the humidity parameter:
 - **Name:** Humidity
 - **Virtual Pin:** V5 (or any other available pin).
 - **Data Type:** Double.

- **Min/Max Values:** Set appropriate ranges based on your sensor's output.
 - **Enable the Historic data toggle button.**
- Repeat the process for both the LED parameter:
 - **Name:** LED State
 - **Virtual Pin:** V2 (or any other available pin).
 - **Data Type:** Integer.
 - **Min/Max Values:** Set appropriate ranges based on your sensor's output(0,1).
 - **Enable the Historic data toggle button.**



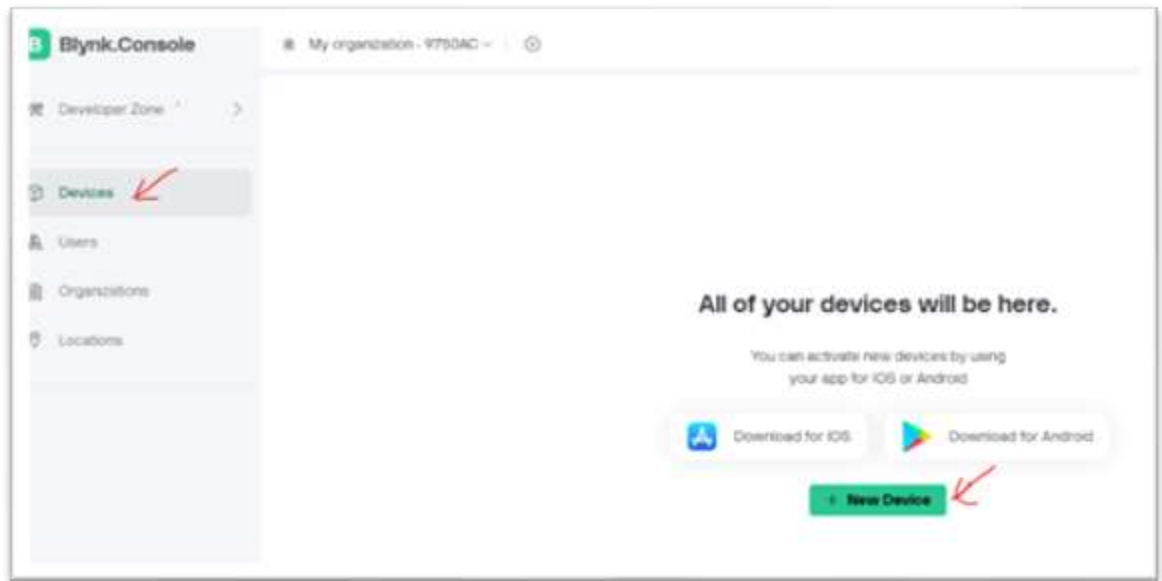
- Save the template after adding the pins.

3.4 Create a Device

In Blynk, devices represent the individual hardware units (such as microcontrollers or sensors) connected to the Blynk platform, enabling you to control and monitor them remotely through the Blynk app. Each device can be associated with a template for standardized configuration and management.

3.5 Adding device

- Navigate to the **Devices** section.
- Click on **+ New Device**.



- Select the template you just created [Monitor ESP32 with sensors].
- Name your device and assign it to a location or group if necessary.
- Copy the details popup after creating the device.

```
#define BLYNK_TEMPLATE_ID " "
```

```
#define BLYNK_TEMPLATE_NAME " "
```

```
#define BLYNK_AUTH_TOKEN
```

```
" "
```

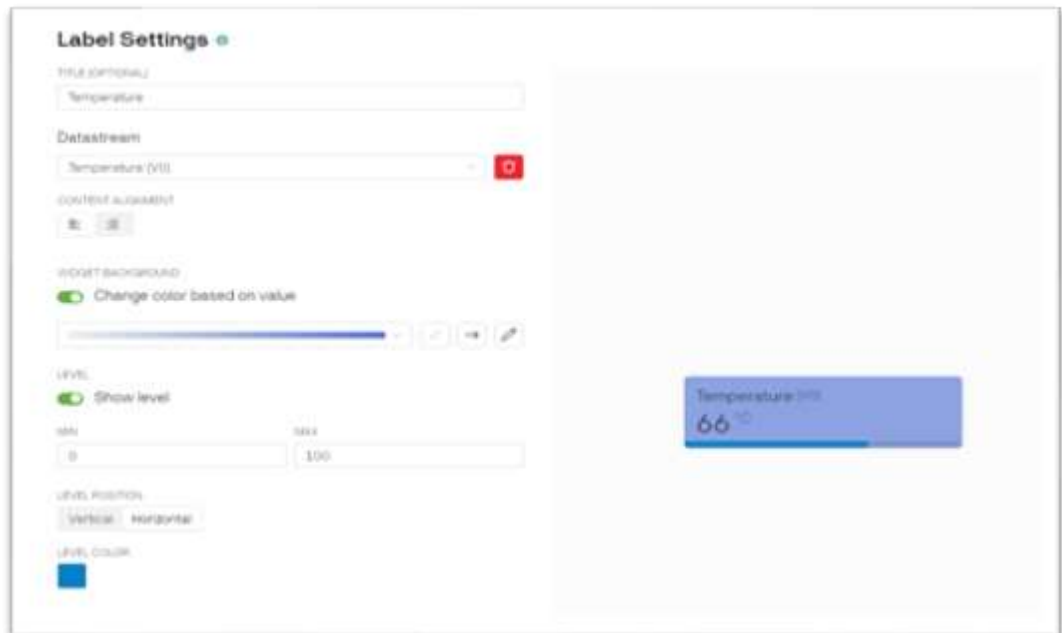
NOTE: The above parameters to be replaced in the code used for ESP32 microcontroller.

3.6 Configure the Dashboard

- Navigate to the **Dashboards** section.
- Select your newly created device.

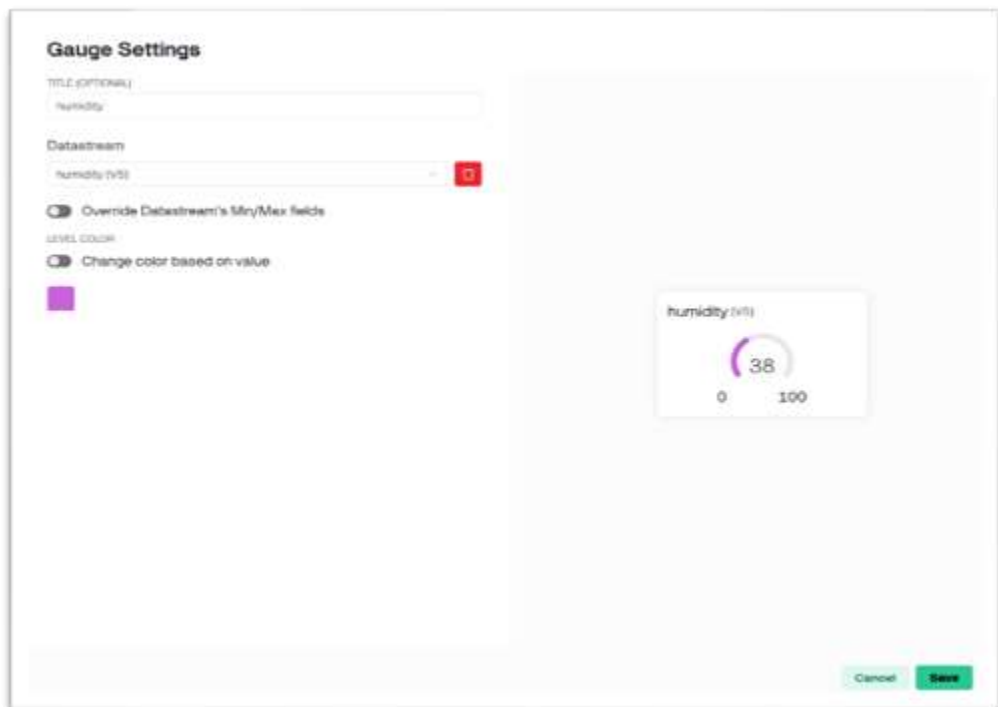


- Click on **Edit Dashboard** to customize it.
- Add widgets to your dashboard:
 - **Temperature Widget:**
 - Go to settings → datastream.
 - Drag a **Chart** and **Label** widget to the dashboard or double click to add the same.
 - Link it to the Temperature datastream (V6).
 - Configure display settings (e.g., units, range, color).



- **Humidity Widget:**
 - Drag a **Gauge** and **Label** widget to the dashboard.
 - Link it to the Humidity data stream (V5) similar to Temp widget.

- Configure display settings.



Gauge Settings

TITLE (OPTIONAL)
humidity

Datstream
humidity (V2)

☒ Override Datstream's Min/Max fields

LEVEL COLOR
☒ Change color based on value

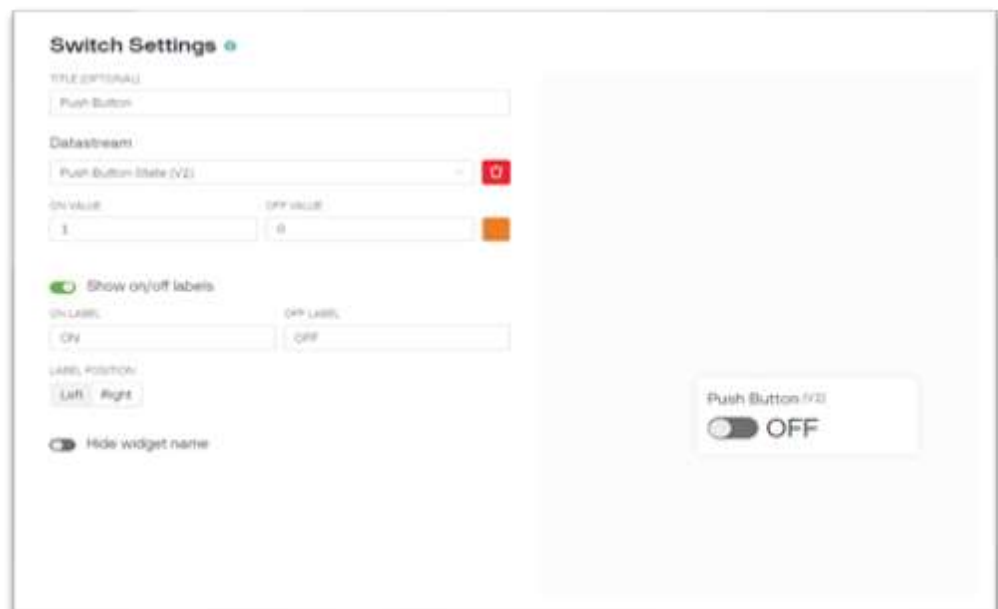
38

0 100

Cancel Save

- **LED Widget:**

- Drag a Switch widget to the dashboard.
- Link it to the LED data stream (V2).
- Configure display settings.
- Click on Save and Apply after adding the widgets.



Switch Settings

TITLE (OPTIONAL)
Push Button

Datstream
Push Button State (V2)

ON VALUE
1

OFF VALUE
0

☒ Show on/off labels

ON LABEL
ON

OFF LABEL
OFF

LABEL POSITION
Left Right

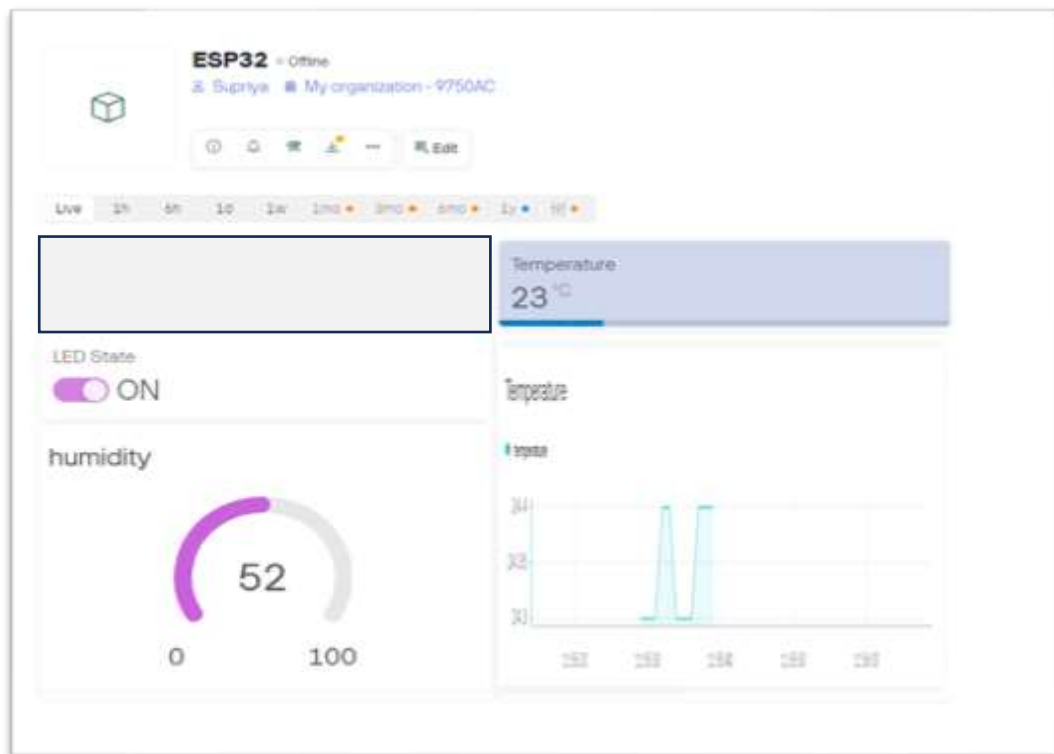
☒ Hide widget name

Push Button (V2)

OFF

4 Monitor Your Dashboard

- Return to the Blynk.
- Navigate to your device's dashboard.
- Select Live option.
- You should now see real-time data for temperature and humidity.
- This is how your dashboard will look like now.



5 How to Burn Code to ESP32 from Arduino IDE

The ESP32 is a powerful microcontroller with built-in Wi-Fi and Bluetooth capabilities, making it ideal for a wide range of projects. This manual provides a step-by-step guide on how to burn code to the ESP32 using the Arduino IDE, a popular platform for developing and uploading code to various microcontroller boards.

5.1 Requirements

Before starting, ensure you have the following items:

- **ESP32 Development Board**
- **USB Cable (Micro-USB or USB-C, depending on your ESP32 model)**
- **Computer with Internet Access**
- **Arduino IDE installed.**

5.2 Installing Arduino IDE

- **Download Arduino IDE:** Visit the Arduino website and download the latest version of the Arduino IDE compatible with your operating system.
- **Install Arduino IDE:** Follow the installation instructions specific to your operating system (Windows, macOS, Linux).

5.3 Setting Up Arduino IDE for ESP32

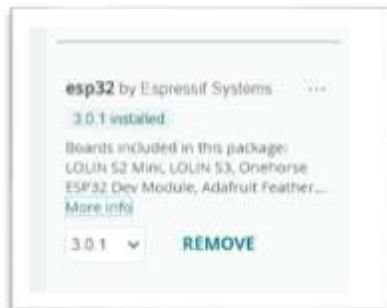
- Open Arduino IDE.
- Add ESP32 Board Manager URL by navigating to Tools → Board Manager.
- Go to File -> Preferences.
- In the "Additional Board Manager URLs" field, add the following URL:

https://dl.espressif.com/dl/package_esp32_index.json

- Click Ok.

5.4 Install ESP32 Board

- Go to Tools -> Board -> Boards Manager.
- Search for "ESP32" in the search bar as shown in the image below.

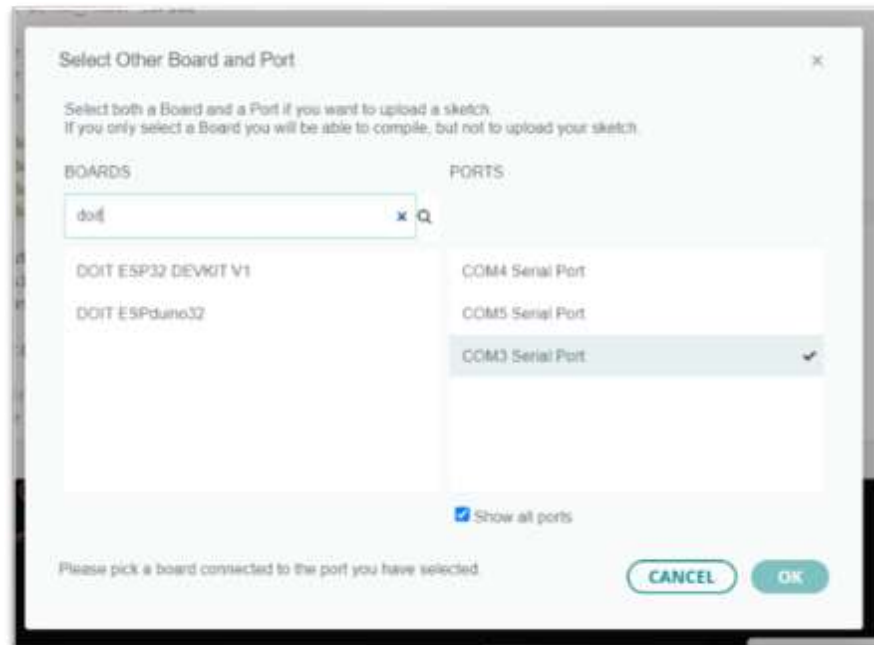


- Select "esp32 by Espressif Systems" and click Install.
- Wait for the installation to complete.

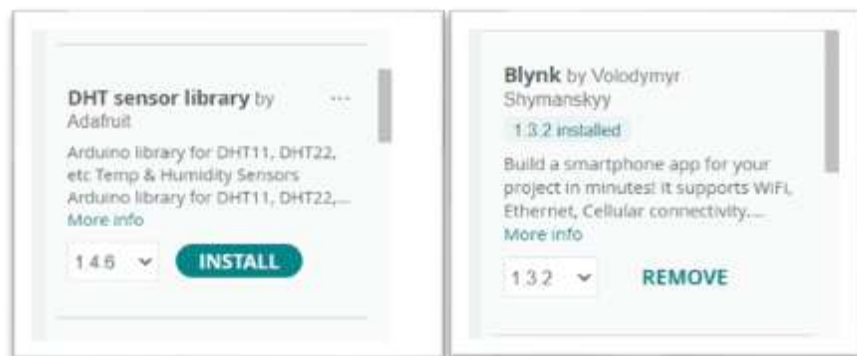
5.5 Connecting ESP32 to Your Computer

- **Connect the ESP32:**
 - Use the USB cable given to connect your ESP32 development board to your computer.
- **Select the ESP32 board and port:**

- Go to Tools -> Board, and select your ESP32 model (e.g., "ESP32 Dev Module").
- Go to Tools -> Port and select the COM port that corresponds to your ESP32 board. This might be labelled as "COM3", "COM4", etc., on Windows, or "/dev/ttyUSB0" or similar on macOS/Linux as shown below.



- If Port option is greyed out or not found, please follow the steps mentioned in Troubleshooting to install the required driver.
- **Adding required dependencies:**
 - Once board is added go to Sketch→ Include Library → Manage Libraries & install the below mentioned dependencies.



5.6 Troubleshooting

- **Port Not Found:** Ensure the USB cable is properly connected and that the correct port is selected in the Arduino IDE.

- And if still not connected please follow the following steps to install the CP210x driver:
 - https://www.silabs.com/documents/public/software/CP210x_Universal_Windows_Driver.zip
- **Driver Issues:** Install the necessary drivers for your ESP32 board if it's not automatically recognized.
- **Error Messages:** Common errors and solutions can be found on the ESP32 troubleshooting forums and the Arduino IDE support pages.

5.7 Uploading Your Program

- **Open the Github URL mentioned below & go through the readme.md for more information.**
 - https://github.com/shivag12/esp32_tinker_lab_v1/tree/main
- **Download the code from here.**
 - https://github.com/shivag12/esp32_tinker_lab_v1/blob/main/esp32_sketch.ino
- **Now add it in the Sketch Tab.**
 - Go to File -> New Sketch
 - And add your code here & save.
- **Now add the configuration parameters you get it from BLYNK application which is described in detail in the above section.**
 - BLYNK_TEMPLATE_ID
 - BLYNK_DEVICE_NAME
 - BLYNK_AUTH_TOKEN
 - ssid
 - pass

NOTE: *ssid & pass will be your wifi/mobile hotspot username & password respectively.*

- **Verify the Code:**
 - Click the Checkmark button (Verify) to compile the code. Ensure there are no errors.
- **Upload the Code:**
 - Click the Arrow button (Upload) to upload the code to the ESP32.
 - Wait for the message "Done uploading" in the status bar.
- **Confirm the Program:**
 - If everything is correct, the built-in LED on the ESP32 should start blinking.

5.8 Enabling Events and Notifications

To make your Blynk project more interactive and responsive, you can enable events and notifications. This allows you to get alerts for specific conditions or actions taken by your devices.

5.8.1 Steps to Enable Events and Notifications

- **Log in to Blynk Console:**
 - Open your Blynk account in a web browser.
- **Select Your Project:**
 - Navigate to the project where you want to enable events and notifications.
- **Go to the Events Tab:**
 - Find the "Events" tab in your project dashboard.
- **Create a New Event:**
 - Click on "Create New Event".
 - Define the event name, description, and the conditions that trigger the event (e.g., sensor value exceeds a threshold) as shows below.

tempalert

General
Notifications
Expose to Automations

EVENT NAME
EVENT CODE

tempalert
tempalert

TYPE

Info
Warning
Critical
Content

DESCRIPTION (OPTIONAL)

Event description (optional)

0 / 300

Limit

Every 1 message will trigger the event

Event will be sent to user only once per 1 second

☒ Show event in Notifications section of mobile app

☒ Send event to Timeline

Other

Apply tax when this event is recorded
Choose Tax

Cancel
Save

5.8.2 Set Up Notifications

- Choose how you want to receive notifications (email, push notifications, etc.).
- Customize the message and notification settings as shown below.

tempalert

General **Notifications** Expose to Automations

☒ Enable notifications

Default recipients

E-MAIL TO
Device Owner X

PUSH NOTIFICATIONS TO
Device Owner X

SMS TO
Device Owner X

☒ Deliver push notifications as alerts
When turned on, push notifications will use critical alert sounds. End-users will need to turn this setting on in their app settings. They can also change a sound.

Notifications Management
When turned ON, end-users will access advanced notification management for this event

☐ Enable notifications management

Cancel Save

5.8.3 Apply and Save

- Save the event and notification settings.
- Ensure your ESP32 code includes the necessary commands to trigger these events using the Blynk event function as shown below.

Below is an example of simple condition were the temperature is greater than 25 degrees.

```
Blynk.logEvent("tempalert","Temperature is high");
```

Note: Event definition should be created in the blynk IoT cloud before configuring it.

By following these steps, your Blynk project will be able to notify you about important events, enhancing the overall functionality and user experience.

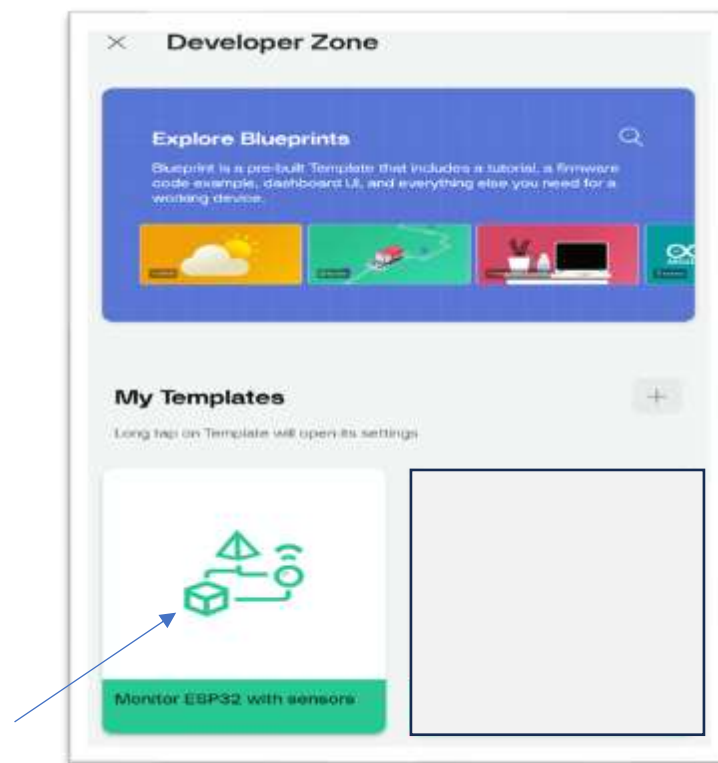
6 Step-by-Step Guide to Create a Dashboard in Blynk Mobile

6.1 Sign Up and Log in to Blynk Mobile

- Go to [Blynk](#).
- Log in to your Blynk account with same id you used it before.

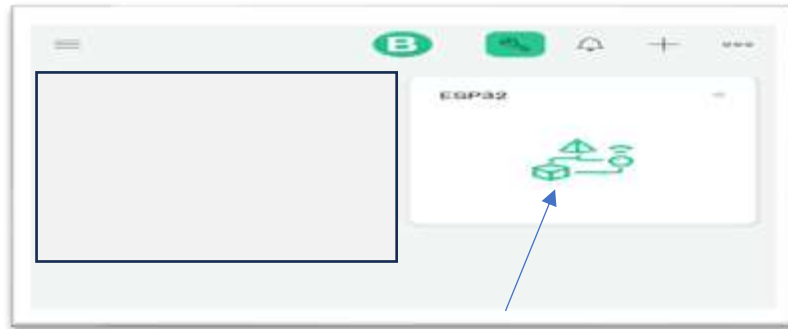
6.2 Find your Template

- Navigate to the **Templates** section.
- Find the same template as configured above Monitor ESP32 with sensors [TMPL3-P-V73DL].

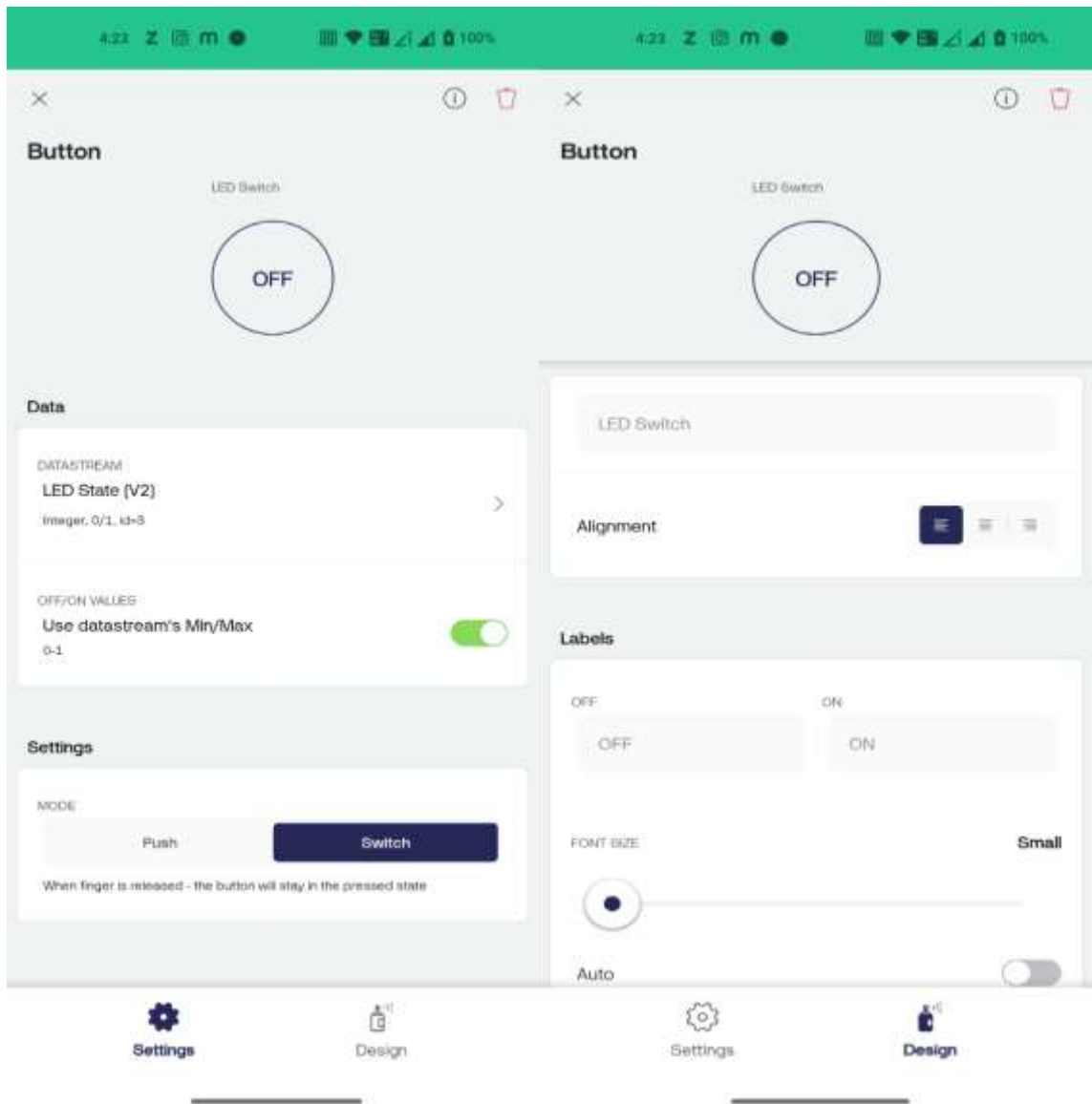


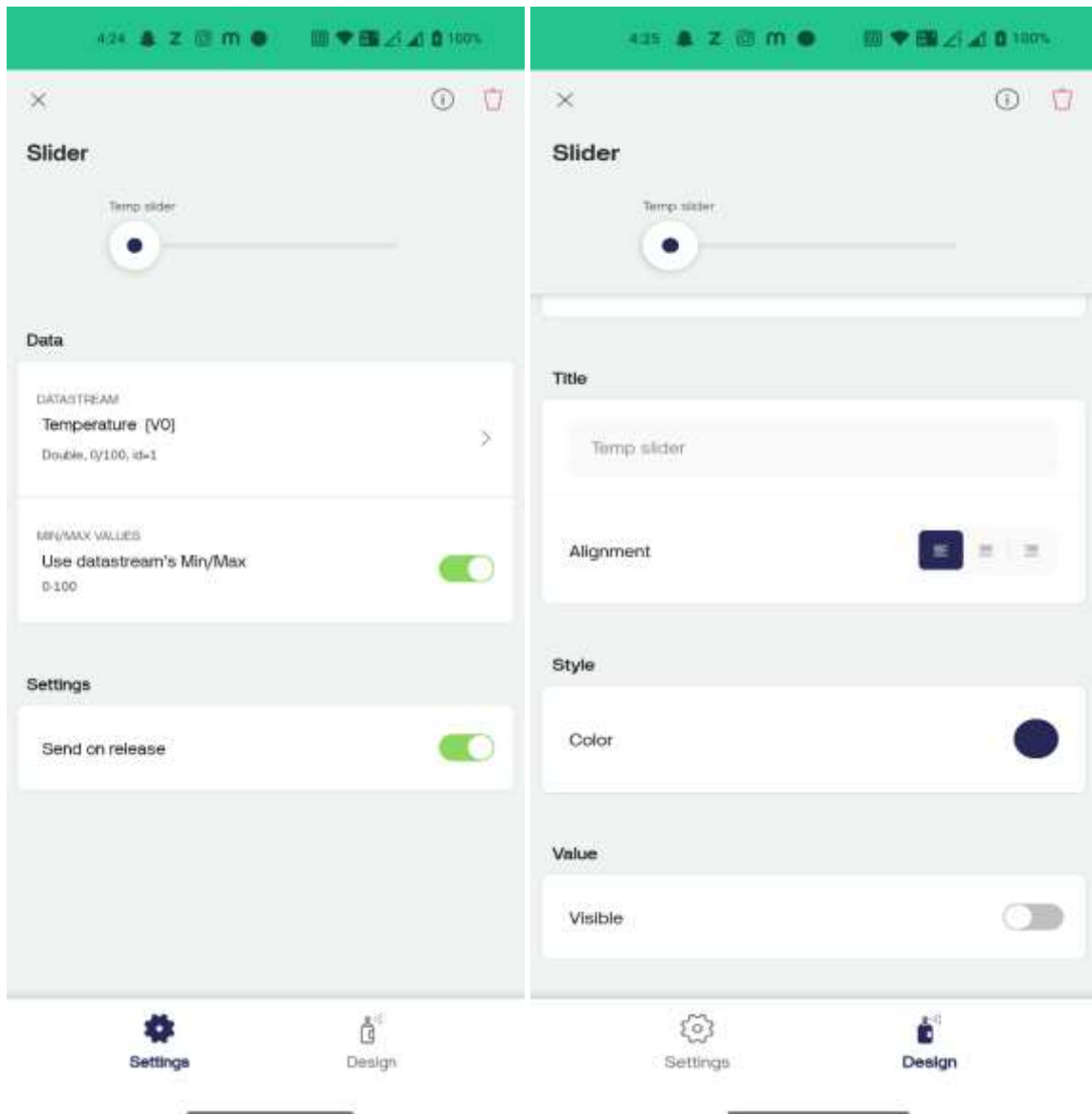
6.3 Configure the Dashboard

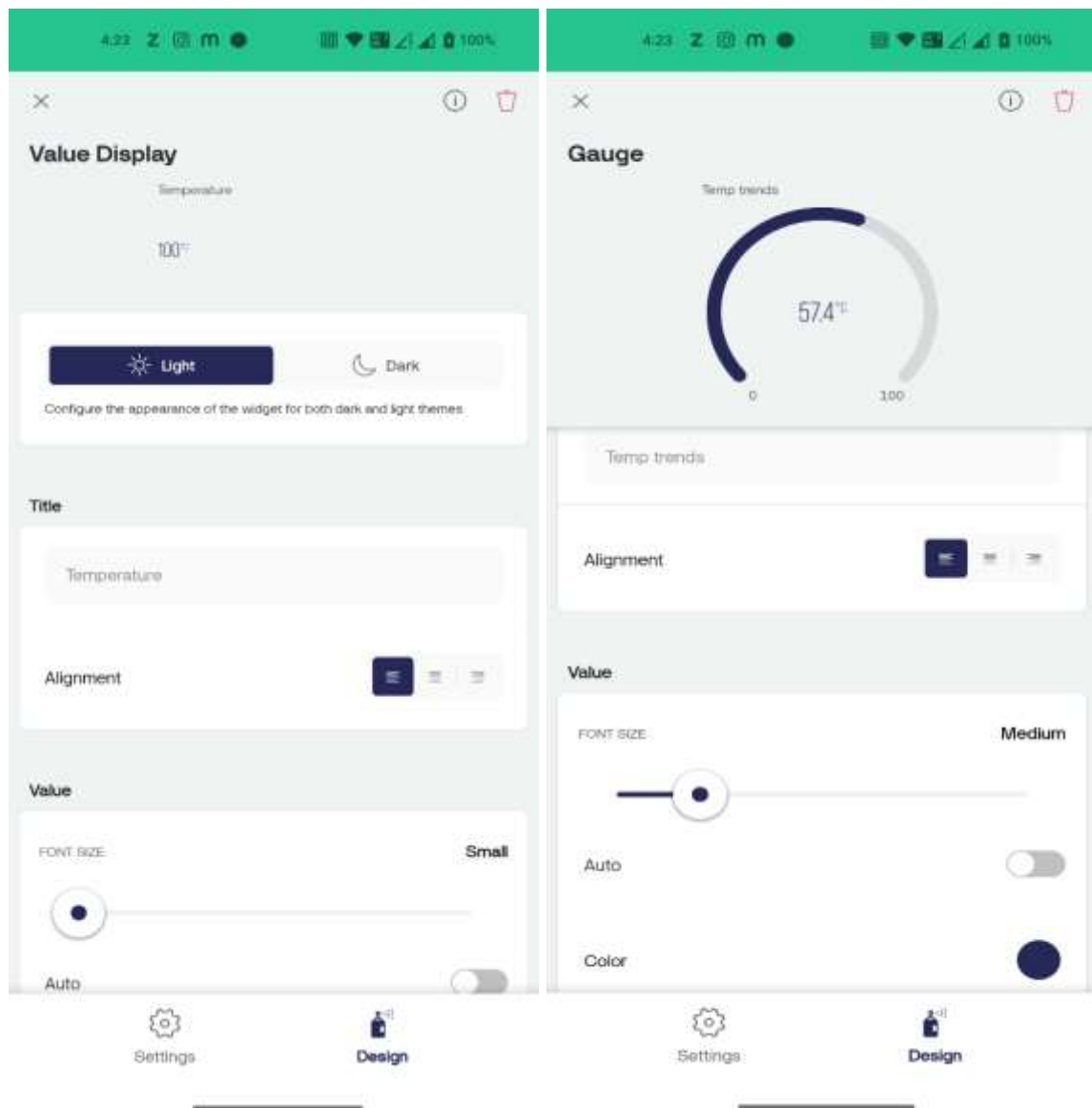
- Go to devices.
- Select your created device [ESP32].



- Add widgets to your dashboard:
 - **Temperature Widget:**
 - Drag a **Slider**, **Gauge** and **Label** widget to the dashboard.
 - Link it to the Temperature datastream (V0).
 - Configure display settings (e.g., units, range, color).
 - **Humidity Widget:**
 - Drag a **Slider**, **Gauge** and **Label** widget to the dashboard.
 - Link it to the Humidity datastream (V1).
 - Configure display settings.
 - **LED Widget:**
 - Drag a **Switch** widget to the dashboard.
 - Link it to the LED datastream (V2).
 - Configure display settings.



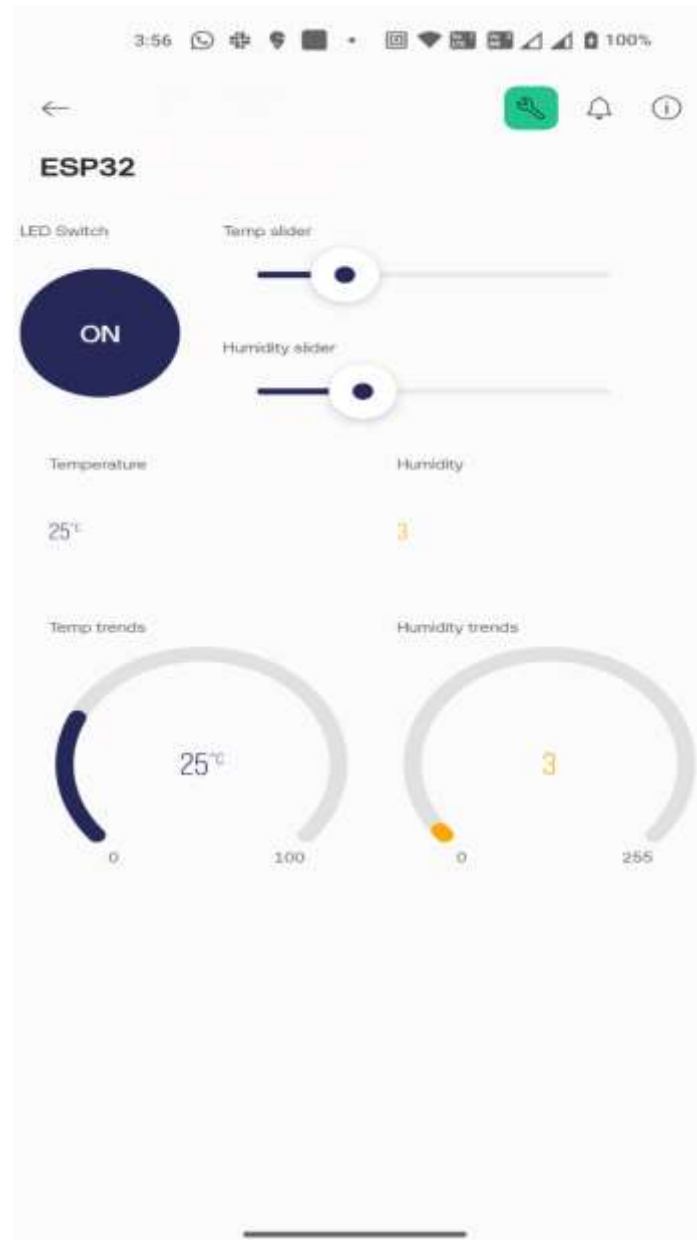




- Click on Save and Apply after adding the widgets.

7 Monitor Your Dashboard

- Return to the Blynk.
- Navigate to your device's dashboard.
- You should now see real-time data for temperature and humidity.



8 Tasks

8.1 Push Heat Index Value to Blynk IoT Cloud

Objective:

Students will enhance an existing ESP32 project to calculate and send the Heat Index value to the Blynk IoT Cloud & display the same over dashboard. This will be similar to the process used to send temperature and humidity values.

8.2 Trigger an event upon reaching a threshold of temperature/humidity/heat-index

Objective:

Design and implement using Blynk Cloud to monitor temperature, humidity, and heat index. Set up triggers to send notifications when these parameters exceed predefined thresholds.