**Data Flow of IoT Environmental Monitoring System (MQ135 + DHT22)**

1. Sensor Layer (Data Acquisition)

* **MQ135** Sensor - measures air quality (gases like CO₂, NH₃, NOx, smoke, alcohol, benzene).
* **DHT22** Sensor - measures **temperature** and **humidity**.
* Both sensors are connected to an **ESP32S3** microcontroller.

2. Microcontroller Layer (Edge Device)

The **ESP32**:

* Reads analog and digital values from **MQ135** and **DHT22**.
* Formats sensor data into a structured format like JSON.
* Connects to a network via "Wi-Fi".
* Publishes the data to a server using "MQTT Broker".

3. Network/Protocol Layer

* Use Rasberry Pi 4 Model B(Booted Pi OS Lite 64bits): To host MQTT Broker and Run FastAPI API(Backend Server)
* Communication Protocol: Typically MQTT(messaging protocol) for lightweight and real-time.
* Data is transmitted from **ESP32S3** to:
  + An MQTT Broker (e.g., Mosquitto)
  + Send real time sensors' data to API

4. Backend Layer (Data Processing & Storage)

* MQTT Broker / FastAPI Server:
  + Receives sensor data from ESP32S3.
  + Stores it in MongoDB.

5. Frontend Layer (User Interface)

* Website to show sensors' data:
  + Connects to backend via WebSocket.
  + Displays real-time and historical data (temperature, humidity, air quality).

To Control LED

When ON/OFF button clicks from website(Frontend), send the buttton's ON/OFF status to API(Backend server) via websocket and publishes to **ESP32S3** through MQTT Broker and then control the LED ON/OFF.

**Summary Diagram**

**For show sensors' data**

[MQ135 + DHT22 Sensors]

↓

[ESP32S3]

- Reads sensor data

- Converts to JSON

- Sends over Wi-Fi

↓

[Network: MQTT Broker]

↓

[FastAPI Backend Server]

- Validates & stores data

- Triggers alerts

↓

[Website(Frontend)]

- Displays charts & status

**For LED Control**

[Website (Frontend)]

- Click ON/OFF Button

- Send ON/OFF status to API

↓

[FastAPI Backend Server]

- Publish ON/OFF status to ESP32S3 via MQTT Broker

↓

[ESP32S3]

- Control LED