

# IoT Sensor Data Collection and Visualization. Project Report

Student: Imanbayeva Diana

Group: ST-2304

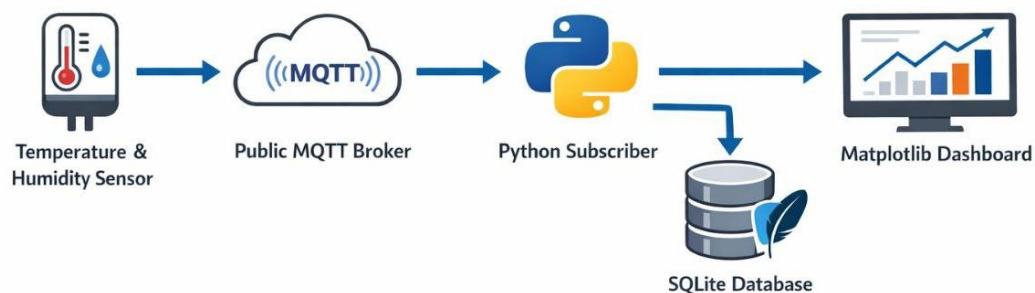
## 1. Introduction

The objective of this project is to demonstrate the workflow of an IoT system, including data generation, transmission, storage, and visualization. A **simulated IoT sensor** was used to generate temperature and humidity readings. This approach allows focusing on the complete data pipeline, database handling, and dashboard creation without requiring physical sensors. The project also incorporates the **MQTT protocol** to simulate real-time IoT data transmission, demonstrating how sensors communicate with a server in a typical IoT environment.

## 2. System Architecture

The project consists of the following components:

- **Simulated IoT Sensor** - generates temperature and humidity readings periodically.
- **MQTT Broker** - a public broker was used to simulate real-time sensor data transmission.
- **Subscriber and Database** - a Python subscriber receives MQTT messages and stores them in a **SQLite database**.
- **Visualization / Dashboard** - Matplotlib is used to plot time-series graphs, providing a real-time-like dashboard.



*Figure 1: System Architecture*

## 3. Data Collection

- Data points were generated every 2 seconds using a Python script in Google Colab.
- Temperature ranged between 20-30 °C, humidity ranged between 40-60 %.
- Sensor readings were serialized in JSON and published via MQTT.
- The subscriber program deserialized the messages and saved them into SQLite.

This approach ensures reproducibility and demonstrates the **full pipeline of IoT data acquisition**, even without physical sensors.

#### 4. Data Storage

The SQLite database contains a table sensor with fields:

Field	Type	Description
time	TEXT	Timestamp
temperature	REAL	Temperature (°C)
humidity	REAL	Humidity (%)

This structure allows efficient storage and retrieval for analysis and visualization.

#### 5. Visualization

- a) **Matplotlib** plots time-series of temperature and humidity.
- b) The dashboard displays changes over time, simulating real-time monitoring.
- c) Graphs provide clear insight into trends and fluctuations of environmental conditions.

```
Collecting paho-mqtt
  Downloading paho_mqtt-2.1.0-py3-none-any.whl.metadata (23 kB)
  Downloading paho_mqtt-2.1.0-py3-none-any.whl (67 kB)
67.2/67.2 kB 4.2 MB/s eta 0:00:00
Installing collected packages: paho-mqtt
Successfully installed paho-mqtt-2.1.0
Database ready
Simulated IoT sensor started
Published: {'time': '17:29:59', 'temperature': 22.31, 'humidity': 41.75}
/tmp/ipython-input-4285550369.py:26: DeprecationWarning: Callback API version 1 is deprecated, update to latest version
  publisher = mqtt.Client()
Published: {'time': '17:30:01', 'temperature': 25.92, 'humidity': 45.42}
Published: {'time': '17:30:03', 'temperature': 28.93, 'humidity': 46.13}
Published: {'time': '17:30:05', 'temperature': 20.98, 'humidity': 54.99}
Published: {'time': '17:30:07', 'temperature': 25.22, 'humidity': 51.49}
Published: {'time': '17:30:09', 'temperature': 21.28, 'humidity': 48.18}
Published: {'time': '17:30:11', 'temperature': 22.29, 'humidity': 57.93}
Published: {'time': '17:30:13', 'temperature': 20.41, 'humidity': 52.19}
Published: {'time': '17:30:15', 'temperature': 24.68, 'humidity': 49.64}
Published: {'time': '17:30:17', 'temperature': 22.75, 'humidity': 41.46}
/tmp/ipython-input-4285550369.py:57: DeprecationWarning: Callback API version 1 is deprecated, update to latest version
  subscriber = mqtt.Client()
Data collection finished
```

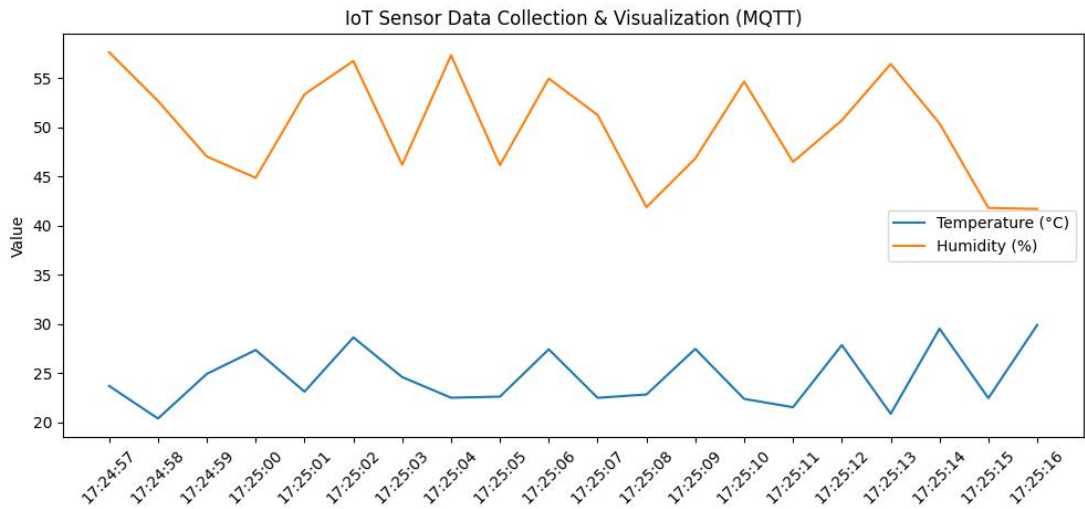


Figure 2: Google Colab Output Dashboard Screenshot

## 6. Technologies Used

Component	Purpose
Python	Scripting and data processing
MQTT (paho-mqtt)	Simulate sensor data transmission
SQLite	Store sensor data
Matplotlib	Visual dashboard creation
Google Colab	Cloud environment to run code

## 7. Conclusion

This project successfully demonstrates a **simplified IoT data pipeline** from data generation to visualization. Using simulated sensors with MQTT allows realistic demonstration without hardware.

The system can be extended to include **real IoT devices, additional sensors, or larger databases**, providing a scalable and flexible IoT environment.