**National Textile University, Faisalabad**

A logo with blue and red squares

Description automatically generated

**Department of Computer Science**

|  |  |
| --- | --- |
| **Name:** | Hasham Ul Haq |
| **Class:** | BSAI-6th |
| **Registration No:** | 22-NTU-CS-1361 |
| **Assignment:** | 2 |
| **Course Name:** | IOT |
| **Submitted To:** | Dr Nasir Mehmood |
| **Submission Date:** | 02-06-2025 |

# Section A

1. **What is Thing Speak and why is it used in IoT applications?**

ThingSpeak is a cloud-based platform designed for real-time visualization of IoT sensor data. It enables efficient monitoring of devices by assigning separate channels to each project

1. **How does MQTT differ from HTTP in terms of communication for IoT devices?**

MQTT is a lightweight protocol based on the publish-subscribe model, making it ideal for live data exchange. In contrast, HTTP relies on request-response and is less suitable for continuous data communication.

1. **Why is multi-core tasking important in an ESP32-based IoT system?** ESP32 has two cores, which allows it to handle multiple tasks at the same time like reading sensor data while uploading it without delays or interruptions.
2. **What is a mutex/lock, and why is it needed when updating an OLED display concurrently?**

A mutex (mutual exclusion) is used to avoid conflicts when multiple tasks try to access the same resource. It prevents the OLED display from getting corrupted data when updated by different processes.

1. **Name two features of Firebase that make it suitable for real-time sensor data logging.**
   * Real-time database that syncs instantly across devices
   * Easy REST API integration with microcontrollers like ESP32
2. **What is InfluxDB, and why is it preferred over traditional databases for IoT data?**

InfluxDB is a time-series database designed for storing timestamped data. It’s ideal for IoT because it efficiently handles frequent, time-based sensor readings.

1. **Define time-series data and give one example from this assignment.** Time-series data consists of values recorded at regular time intervals. Example: Logging temperature and humidity from a DHT22 sensor to InfluxDB every 5 seconds.
2. **What is the purpose of NTP synchronization in IoT systems?** NTP (Network Time Protocol) syncs the system time on devices like ESP32 to ensure accurate timestamps for data logging and time-sensitive operations.

# Section B

**9. ESP32 to ThingSpeak Flow Design:**

• **Step-by-step flow:**

* Connect ESP32 to Wi-Fi. o Read data from DHT22 sensor.
* Format data and include the ThingSpeak **API key**.
* Send an HTTP GET request with data to ThingSpeak’s URL. **code:**

python CopyEdit connect\_to\_wifi() read\_temp\_humid()

url =

"https://api.thingspeak.com/update?api\_key=XXXX&field1=temp&field2=h umid" send\_http\_request(url)

# Arduino-C vs MicroPython

|  |  |  |
| --- | --- | --- |
| **Feature** | **Arduino (C++)** | **MicroPython** |
| Ease of Use | Harder syntax | Easier, Pythonic syntax |
| Performance | Faster, low-level control | Slightly slower |
| Hardware Control | Deep hardware access | Some limitations |
| Ecosystem Support | Huge community, many  libraries | Growing, but smaller ecosystem |
| Learning Curve | Steep for beginners | Friendly for AI/CS students |

**11. Firebase Data Flow Design:**

• **Flow:**

* ESP32 syncs time via **NTP**.
* Reads sensor data (temp/humidity).
* Formats data into **JSON**:

{ "temp": 25, "humidity": 60, "timestamp": "2025-06-01T12:00:00Z" } oSends to Firebase using HTTP with secure **authentication**.

* Firebase stores and displays in real-time.

**12. Environmental Classification**

# (Python + InfluxDB)

* **Steps:**
  1. Fetch data from InfluxDB using influxdb-client-python.
  2. Preprocess (clean nulls, normalize).
  3. Train model (e.g., Decision Tree or KNN) on temp/humidity.
  4. Predict environment class (e.g., "Hot", "Comfortable").
  5. Write result back to InfluxDB for visualization.
* **Snippet:**

from sklearn.tree import DecisionTreeClassifier

model.fit(X\_train, y\_train) predictions = model.predict(new\_data) write\_to\_influxdb(predictions)

**Section C:**

**13. End-to-End IoT Architecture Comparison**

1. **Gateway-based vs Cloud-based Model**

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Gateway Model (Edge)** | **Cloud Model** |
| Processing | Done locally on device/gateway | Done on cloud servers |
| Latency | Very low | Higher due to network delays |
| Deployment | More complex | Easier (just send to cloud) |
| Scalability | Limited by hardware | Easily scalable |

1. **IoT System Layers and Components:**

|  |  |
| --- | --- |
| **Layer** | **Example Components** |
| Sensing | DHT22, Soil Moisture Sensor |
| Communication | Wi-Fi, MQTT, HTTP |
| Storage | Firebase, InfluxDB |
| Visualization | ThingSpeak, Grafana |
| DecisionMaking | ML model on ESP32 or cloud-based alerts |
|  |  |

# Diagram

Sensor

ESP32/Gateway

Local Processing

(

Edge

)

Cloud/Firebase

Grafana/ThingSpeak

# Bonus Q14

**AI-Based Environmental Prediction (Edge Computing)**

* AI models can be trained on past temperature/humidity data.
* The trained model (e.g., Decision Tree) can be stored on ESP32.
* When new data arrives, prediction is done **on-device**.
* This reduces delay and keeps the system fast, even offline.
* Edge inference saves bandwidth and improves real-time actions.
* Cloud can still be used for logging and retraining models.
* Apps include smart farming (e.g., irrigation alerts), HVAC systems, or home automation.