Aim 2: Basic of Network (CoAP, MQTT) and Cloud (ThingSpeak).

Theory:

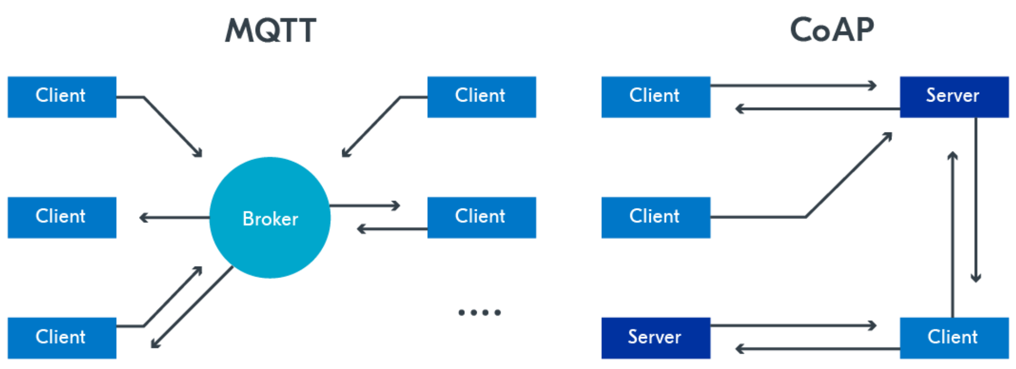
1. **Constrained Application Protocol (CoAP)**

CoAP (Constrained Application Protocol) is a specialized web transfer protocol designed for constrained devices and constrained networks, often found in IoT environments. It's designed to be lightweight and efficient, working over UDP (User Datagram Protocol) to minimize overhead. CoAP follows a RESTful architecture, similar to HTTP, using methods like GET, POST, PUT, and DELETE to interact with resources identified by URIs (Uniform Resource Identifiers).

Key Features:

- **Lightweight:** Minimal message overhead, suitable for devices with limited resources (CPU, memory, power).
- **UDP-based:** Reduces connection establishment overhead compared to TCP.
- **RESTful:** Uses familiar web concepts, making integration with web technologies easier.
- **Request/Response Model:** Clients send requests to servers, which then send back responses.
- **Observe Option:** Allows clients to subscribe to resource changes on a server and receive notifications automatically.
- **Asynchronous Communication:** Supports non-blocking communication.
- **Multicast Support:** Enables sending messages to multiple devices simultaneously.
- **Optional Reliability:** Offers confirmable (CON) messages with acknowledgments for reliable delivery when needed.
- **Security:** Can be secured using DTLS (Datagram Transport Layer Security) over UDP.

Diagram:



Explanation:

- A CoAP client (often a sensor or a constrained device) sends a request to a CoAP server (which could be a gateway or another more powerful device).
- The request specifies a method (e.g., GET to retrieve the current temperature) and a resource URI (/temp).
- The server processes the request and sends back a response with a status code (e.g., 205 Content for success) and the requested data (e.g., 25 degrees Celsius).
- Using the "Observe" option, the client can subscribe to changes in the /temp resource, and the server will send notifications whenever the temperature value changes.

2. Message Queuing Telemetry Transport (MQTT)

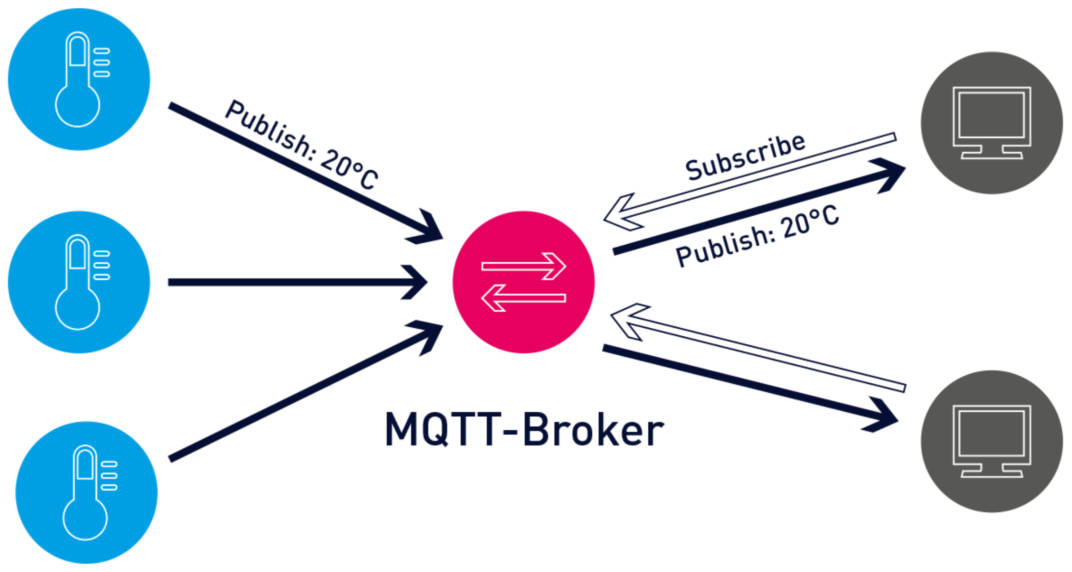
Theory:

MQTT (Message Queuing Telemetry Transport) is a lightweight, publish/subscribe messaging protocol ideal for IoT due to its efficiency and ability to handle unreliable networks. It operates over TCP/IP (or other transports) and uses a central broker to manage communication between clients. Clients can publish messages to specific topics, and other clients can subscribe to those topics to receive the messages.

Key Features:

- **Publish/Subscribe Model:** Decouples message senders (publishers) from receivers (subscribers).
- **Lightweight:** Small message size and low bandwidth usage.
- **Scalable:** Can handle a large number of clients and messages.
- **Reliable:** Offers different Quality of Service (QoS) levels (0: at most once, 1: at least once, 2: exactly once) to ensure message delivery based on application needs.
- **Persistent Sessions:** Allows clients to reconnect and resume their subscriptions and pending messages.
- **Retained Messages:** The broker can store the last message published on a topic and deliver it to new subscribers.
- **Last Will and Testament (LWT):** Clients can define a message that the broker will publish on a specific topic if the client disconnects unexpectedly.
- **Security:** Supports TLS/SSL for encrypted communication and username/password authentication.

Diagram:



Explanation:

- An MQTT publisher (like a temperature sensor) connects to the MQTT broker and publishes a message containing the temperature value to a specific topic (e.g., "sensor/temp").
- The MQTT broker receives this message and forwards it to all clients that have subscribed to the "sensor/temp" topic.
- An MQTT subscriber (like a dashboard application) connects to the same broker and subscribes to the "sensor/temp" topic. It then receives the temperature updates published by the sensor.
- Publishers and subscribers don't need to know each other's existence; they only interact through the broker and the topic.

3. Cloud (ThingSpeak)

Theory:

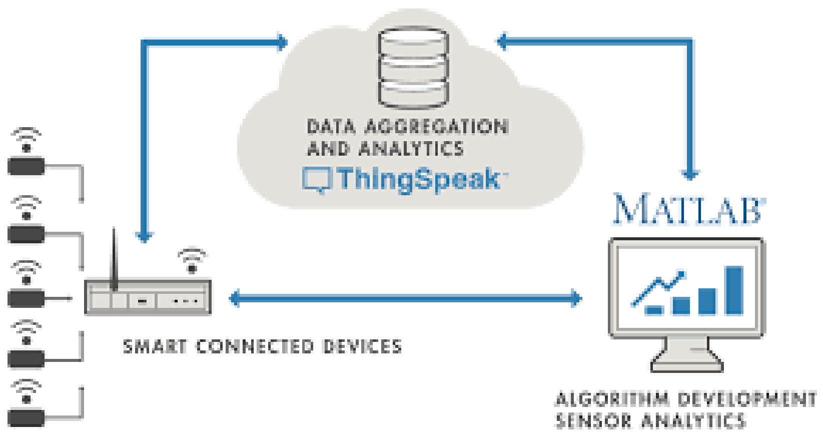
ThingSpeak is an open-source IoT analytics platform service provided by MathWorks. It allows you to collect, visualize, and analyze live data streams in the cloud. It provides channels where you can send data from your IoT devices using protocols like HTTP (REST API) and MQTT. ThingSpeak offers built-in data visualization tools and the ability to use MATLAB for advanced analytics.

Key Features:

- **Data Collection:** Accepts data from various IoT devices and platforms via HTTP REST API and MQTT.
- **Channels:** Organizes data into channels, each containing multiple data fields, location information, and a status field.

- **Data Visualization:** Provides built-in charts and graphs for visualizing real-time and historical data.
- **MATLAB Analytics:** Enables users to write and execute MATLAB code to perform data analysis, signal processing, and machine learning on the collected data.
- **Triggers and Actions:** Allows you to create event-based triggers that can send alerts (e.g., email, SMS) or interact with other web services based on data conditions.
- **APIs:** Offers well-documented REST and MQTT APIs for easy integration with IoT devices and other applications.
- **Public and Private Channels:** Supports both public channels for open data sharing and private channels for secure data storage and analysis.
- **Integrations:** Integrates with other IoT platforms and services like The Things Network.

Diagram:



Explanation:

- An IoT device (like your Raspberry Pi 5 with sensors) connects to the internet via a network.
- It then sends sensor data to the ThingSpeak cloud using either HTTP POST requests to the REST API or by publishing MQTT messages to specific ThingSpeak topics.
- ThingSpeak stores this data in the designated channel.
- Users can then access the ThingSpeak web interface or use the APIs to visualize the data through charts and graphs.
- They can also leverage MATLAB integration for more advanced analysis and create triggers to automate actions based on the data.

These are the fundamental concepts of CoAP, MQTT, and ThingSpeak as they relate to IoT. Understanding these basics will help you in designing and implementing your IoT practical aim using the Raspberry Pi 5. Remember to consider the specific requirements of your project when choosing which protocols and platforms to use.