1. **What are the main differences between HTTP, MQTT, and CoAP?**
   HTTP, MQTT, and CoAP differ primarily in their communication models, transport protocols, and design goals. HTTP is a general-purpose web protocol, while MQTT and CoAP are specifically optimized for the constraints of Internet of Things (IoT) and machine-to-machine (M2M) communication.

- **HTTP when:** Your devices have plenty of power and bandwidth, and you need a high degree of interoperability with existing web-based systems. It's often used for cloud integrations where the client isn't heavily constrained.
- **MQTT when:** You need high reliability, efficient distribution of data to many clients, and a low-bandwidth solution for devices with moderate resource constraints. It is widely used for industrial IoT, fleet tracking, and smart home systems.
- **CoAP when:** You are working with extremely constrained, battery-powered devices in low-power and lossy networks. Its UDP-based, low-overhead design is ideal for applications like smart agriculture or environmental sensing.

2. Which protocol would you choose for:

◦Sending temperature data every second
MQTT
◦Controlling a smart bulb (on/off)
CoAP
◦Uploading a large file
HTTP

3. Explain QoS levels (0, 1, 2) in MQTT and give one use case for each.

QoS 0: At most once (fire-and-forget)

- Description: The message is sent once without any acknowledgment from the broker or receiver. The sender discards the message immediately after sending it. If the network connection is lost, the message may be lost.
- Pros: Fastest delivery and lowest overhead.
- Cons: No guarantee of delivery.
- Use case: Non-critical, high-frequency telemetry data where the latest value is the most important.
    - Example: A temperature sensor in a room that sends a reading every second. If one reading is lost due to a network glitch, the next reading will provide a sufficiently up-to-date value, and the lost message is not critical.

QoS 1: At least once

- Description: The sender ensures the message is delivered at least once by storing a copy until it receives a PUBACK acknowledgment packet from the broker. If the PUBACK isn't received within a set time, the message is retransmitted. This can result in duplicate messages being delivered.
- Pros: Guaranteed delivery with less overhead than QoS 2.
- Cons: Potential for duplicate messages, which the application must handle.
- Use case: Important data or commands where message delivery is critical, but duplicates are manageable.
    - Example: A fleet management system receiving telemetry (e.g., speed, location) from a vehicle. If a message is resent due to a temporary disconnection, the system can simply ignore a duplicate reading, but it's crucial to receive every update to ensure the vehicle is tracked correctly.

QoS 2: Exactly once

- Description: This level guarantees that a message is delivered and processed by the receiver exactly once, with no risk of duplication. It achieves this with a four-step handshake between the publisher and the broker: PUBLISH, PUBREC, PUBREL, and PUBCOMP. This provides the highest level of reliability.
- Pros: The safest and most reliable QoS level.

- Cons: Highest overhead and highest latency due to the multi-step handshake process.
- Use case: Mission-critical operations where message loss or duplication would be catastrophic.
    - Example: A factory automation system sending a command to a robotic arm to perform a precise task. If the command is sent twice, it could cause a dangerous malfunction. The overhead is acceptable because these commands are less frequent but absolutely critical.

4. Why does CoAP use UDP instead of TCP?

CoAP uses UDP instead of TCP primarily to minimize overhead and conserve resources, making it suitable for the very constrained environments found in many Internet of Things (IoT) applications. While TCP offers inherent reliability, its connection-oriented nature and heavier packet headers are unsuitable for low-power, low-bandwidth devices.

5. Why is HTTP still widely used even though MQTT and CoAP are lighter for IoT?

Despite its higher overhead compared to MQTT and CoAP, HTTP (and its secure version, HTTPS) remains widely used in IoT because of its familiarity, interoperability with existing web infrastructure, and robustness for specific use cases, particularly with cloud-connected, resource-rich devices.

1. Familiarity and ease of development
2. Seamless integration with web and cloud
3. The evolution of HTTP