Task 3

Internet of Things 2

Amany Mohamed Fathy | Artificial Intelligence|2022565968

Hagar Tarek Shaheen | Artificial Intelligence|20221310699

Loreen Mohamed Saeed | Artificial Intelligence|20221377256

Malak Mahmoud Aref | Artificial Intelligence|20221445867

Nada Mohamed Henedy | Artificial Intelligence| 20221377458

Sama Ahmed Mohamed | Artificial Intelligence| 20221377562

Ereeny Wagih Massoud | Artificial Intelligence| 2022513146

Faculty of Computers and Data Sciences,

Alexandria University.

April 2025

1. Introduction

This project aims to implement a cloud-connected IoT system using MQTT protocol via AWS IoT Core to control and monitor devices like servo motors and IR sensors through an ESP32 microcontroller. The system demonstrates real-time remote object detection and actuation, which can be scaled for use in smart homes, industrial automation, or surveillance systems.

2. System Overview

The system integrates several hardware and cloud-based software components:

- **ESP32**: A powerful microcontroller with integrated Wi-Fi and Bluetooth, ideal for IoT applications. It handles sensor readings, connects to AWS IoT via MQTT, and drives the servo motor.
- **AWS IoT Core**: Acts as the MQTT broker. It securely authenticates and communicates with the ESP32 using certificates and policies.
- **Node-RED Dashboard**: Provides a user-friendly interface to control and monitor the system remotely. It uses MQTT input/output nodes to send commands and visualize data.
- **LCD Display (16x2 I2C)**: Provides real-time feedback on system activities such as object detection or servo motor angles, enhancing local monitoring.

3. Hardware & Software Requirements

Hardware:

- ESP32
- IR Sensor
- Servo Motor
- 16x2 LCD

Software:

- Arduino IDE with ESP32 Board Support Package
- Required libraries: `WiFi.h`, `PubSubClient.h`, `Wire.h`, `LiquidCrystal_I2C.h`, `Servo.h`
- AWS IoT Core Console
- Node-RED with MQTT and dashboard nodes installed
- Fritzing or Tinkercad Circuits for visual schematic

4. AWS IoT Core Setup

Step-by-step:

1. **Create a Thing**:
   - Define a name and type for our device.
2. **Generate Certificates**:
   - Download the device certificate, private key, and Amazon root CA.
3. **Define a Policy**:
   - Create a JSON policy allowing access to `iot:Connect`, `iot:Publish`, `iot:Subscribe`, and `iot:Receive` for required topics.
4. **Attach the Certificate and Policy** to the Thing.
5. **Copy the MQTT Endpoint** and use it with `PubSubClient` in the ESP32 code.

5. ESP32 MQTT Code Implementation

Main Functions Explained:

- **WiFi Initialization**: Connect to local Wi-Fi using `WiFi.begin(ssid, password);` and wait until connected.
- **MQTT Client Setup**: Use `PubSubClient` to connect to the AWS MQTT broker over TLS. Load certificates from SPIFFS or hardcoded as string literals.
- **IR Sensor Integration**: Read sensor digital output using `digitalRead(sensorPin)`. When the sensor detects an object, send a JSON message:
- `{ "object": "detected", "timestamp": "<current_time>" }`

  Publish this to topic `sensor/ir_data`.

- **Servo Control via MQTT Subscription**: Subscribe to `actuator/servo_control`. Parse the payload and use `servo.write(angle);` to set the angle.
- **LCD Updates**: Use `lcd.setCursor()` and `lcd.print()` to show current events and motor positions.
- **Reconnection Strategy**: Periodically check and reconnect to MQTT broker in case of disconnection using `client.connected()` and `client.connect()`.

6. Node-RED Dashboard Configuration

Node-RED was used to create a simple dashboard for real-time monitoring and control of the IoT system. The dashboard connects to AWS IoT Core using MQTT over TLS and provides both data visualization and actuator control.

MQTT Integration

Two MQTT nodes were configured:

- Sensor Data Subscriber: Subscribes to the sensor/ir_data topic to receive IR sensor readings published by the ESP32.
- Servo Control Publisher: Publishes control commands to the actuator/servo_control topic to remotely adjust the servo motor position.

The MQTT connection settings (endpoint, certificates, and keys) matched the AWS IoT Core security requirements.

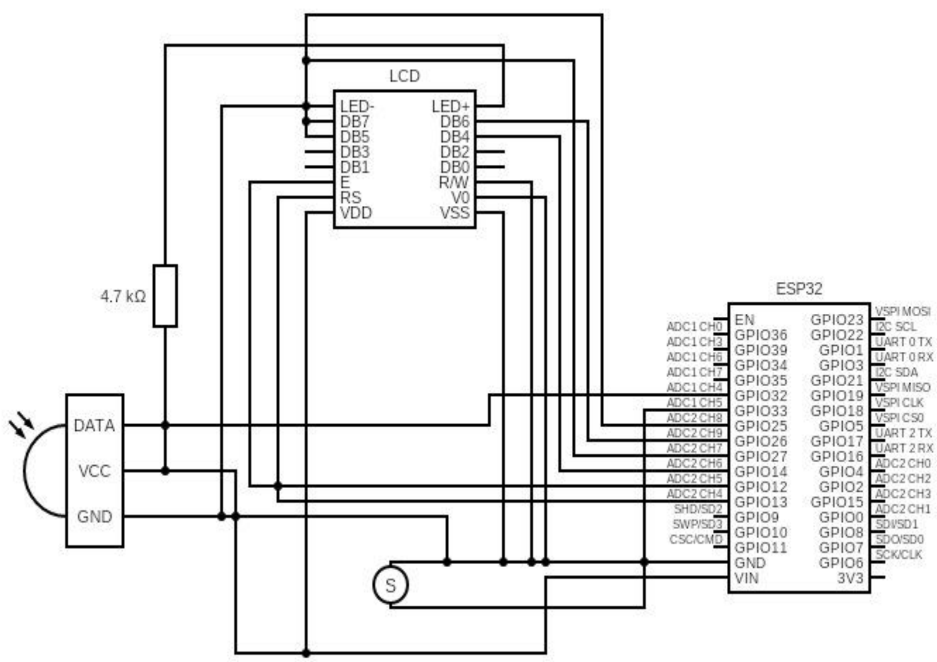
Dashboard Design

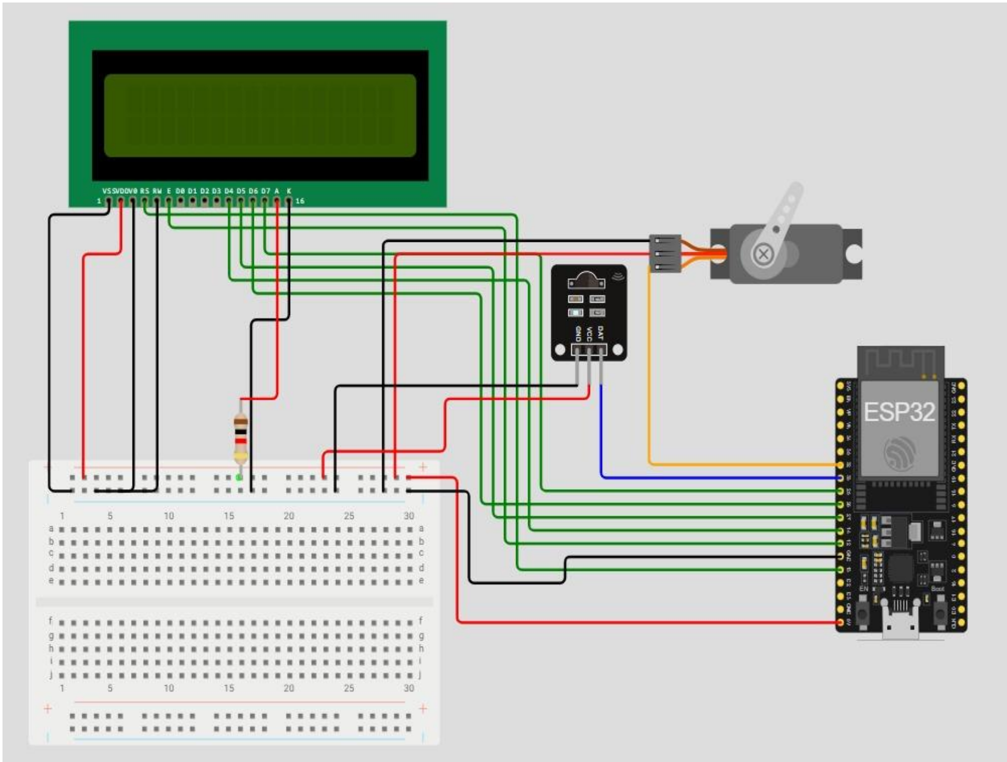
The Node-RED dashboard includes:

- Text Display: Shows the latest IR sensor reading in real time.
- Control Button: Sends a predefined servo angle command (e.g., { "angle": 90 }) when clicked.

This layout allows the user to monitor the system's environment and interact with the servo motor through a web interface.

7. Diagrams





9. Conclusion

- Demonstrated a complete MQTT-based system for sensor-actuator communication.
- Achieved cloud-based monitoring and control using AWS IoT Core.