Water monitoring iot based project

Moataz Mahmoud elsayed 19103233

Ahmed salah elsayed 19102372

Omar Ahmed salah 19101365

Fares tarek 19102421

Omar hashim 19102099

Table of contents:

1. Overview

The IoT Water Quality Monitoring System is designed to measure and monitor water turbidity using Arduino, NodeMCU (ESP8266), and MQTT communication. The project involves the use of various components, protocols, and software libraries to achieve a seamless communication and data processing flow.

2. Protocols Used

MQTT (Message Queuing Telemetry Transport)

We have chosen the MQTT protocol to implement connection between application and server and between micro controller and server.

Mqtt broker works as follows:

1. Publishers: Devices or applications that generate data and want to share it with others are known as publishers. They send messages to specific "topics."

2. Subscribers: Devices or applications that are interested in certain types of data subscribe to specific topics. Subscribers receive messages published to the topics they are subscribed to.

3. Broker: The MQTT broker is a server that acts as an intermediary between publishers and subscribers. It receives messages from publishers and then forwards those messages to all the subscribers that have expressed interest in the corresponding topics.

Popular MQTT brokers include:

Mosquitto: An open-source MQTT broker that is lightweight and widely used in various applications.

3. Components of the Project

Arduino Uno:

Functionality: Monitors water turbidity using a turbidity sensor.

Communication: Interacts with the NodeMCU via SoftwareSerial.

Output: Displays turbidity information on an LCD screen.

NodeMCU (ESP8266):

Functionality: Publishes water turbidity reading values to the MQTT broker.

Communication: Communicates with the Arduino via SoftwareSerial.

Network Connectivity: Connects to the Wi-Fi network for MQTT communication.

Turbidity Sensor:

Functionality: Measures water turbidity based on sensor readings.

LCD Display:

Functionality: Displays real-time turbidity information.

4. Libraries Used

Arduino Code

LiquidCrystal_I2C Library:

Purpose: Facilitates communication with the I2C-connected LCD display.

SoftwareSerial Library:

Purpose: Enables software-based serial communication between the Arduino and NodeMCU and vice versa.

NodeMCU Code:

PubSubClient Library

Purpose: Implements MQTT functionality for communication with the MQTT broker.

JavaScript Code

Paho MQTT javascript client library: This library allows you to establish MQTT connections, subscribe to topics and publish messages.

5. Server:

- **Using Microsoft Azure to create a linux virtual machine and set up on it the MQTT broker which will handle the publish/subscribe messages.**
- **The virtual machine will have a public ip which we will use on both Arduino and web application to publish/subscribe informations.**
- **Mosquitto serves as an MQTT broker, which means it is responsible for receiving messages from clients (devices or applications) and distributing them to other interested clients. The broker acts as a centralized message hub that facilitates communication between devices**
- **Certain configuration must be made on the mqtt config file on server to support the web sockets to enable the communication between the broker and the server.**

6. Conclusion

This project serves as an illustrative example of an IoT application for environmental monitoring, showcasing the integration of various technologies to achieve a unified and functional system.