IOT BASED SMART WATER MANAGEMENT

Developing a complete IoT-based smart water management is a complex project that involves hardware, software and network component.

Smart Water Sensors:

Install water sensors in key areas, like near water sources, pipes, and potential leak points. These sensors will detect changes in water flow or leaks.

Hub or Controller:

You'll need a central hub or controller that connects to the sensors. This hub communicates with the sensors and often connects to your Wi-Fi network.

Network Connection:

Ensure you have a stable Wi-Fi or other network connection for the hub/controller to communicate with the sensors and cloud-based services.

Power Supply:

Depending on the devices, they may require batteries or a reliable power source. Ensure all sensors and the hub have a consistent power supply.

Dashboard or App:

Download and configure the accompanying app or use a web-based dashboard provided by the manufacturer. This is where you'll monitor water usage and receive alerts.

Alerts and Automation:

Set up alerts and automation rules for when certain conditions are met, like excessive water usage or detected leaks. These can be configured through the app or dashboard.

Regular Maintenance:

Periodically check the sensors for battery levels, connections, and ensure they are functioning correctly.

Data Analysis:

Over time, analyze the data collected by the smart water management system to identify patterns, optimize water usage, and detect any anomalies.

PYTHON SCRIPT:

```
Import random

Import time

# MQTT Broker Settings

Broker_address = “mqtt.eclipse.org”

Port = 1883
```

```
# Topics for Water Sensors
Water_level_topic = “water/level”
Water_quality_topic = “water/quality”

# Simulated Sensor Data
Def generate_water_level():
    Return random.uniform(0.0, 100.0)

Def generate_water_quality():
    Return random.uniform(0.0, 10.0)

# MQTT Callbacks
Def on_connect(client, userdata, flags, rc):
    Print(“Connected with result code “ + str(rc))

Def on_publish(client, userdata, mid):
    Print(“Message Published”)

# Create MQTT Client
Client = mqtt.Client()
Client.on_connect = on_connect
Client.on_publish = on_publish

# Connect to MQTT Broker
Client.connect(broker_address, port, 60)
```

```
# Main Loop
While True:
    Water_level = generate_water_level()
    Water_quality = generate_water_quality()

    # Publish sensor data to respective topics
    Client.publish(water_level_topic, water_level)
    Client.publish(water_quality_topic, water_quality)

    Print(f”Water Level: {water_level}, Water Quality: {water_quality}”)

    Time.sleep(5)  # Simulate data update every 5 seconds

# Disconnect from MQTT broker
Client.disconnect()
```

Remote Monitoring:

It enables real-time monitoring, analysis, and efficient decision-making to optimize water usage and address potential issues promptly.

Power and Connectivity:

Smart water management systems use sensors and IoT technology to monitor water usage, quality, and infrastructure, enabling real-time data collection and analysis. This enhances efficiency, reduces waste, and helps in proactive maintenance of water networks.

Testing and Callibration:

It involves validating and fine-tuning IoT sensors and devices to ensure accurate data collection and analysis related to water usage, quality, and distribution.

This script simulates two water sensors, one for water level and the other for water quality. It generates random data and publishes it to MQTT topics. You should replace the broker_address and port with the information of your MQTT broker. Additionally, in a

real-world scenario, you would use actual sensor data from hardware.To make this script more practical, you would need to set up an MQTT broker, connect real water sensors to a microcontroller, and modify the script to read data from these sensors.