NAAN MUDHALVAN PROJECT FILE FOR PHASE 3:

PROJECT NAME: SMART WATER MANAGEMENT

Hardware components of Smart Water Management:

Smart water management systems typically consist of various hardware components, including:

1.Sensors
2.Communication infrastructure
3.Data loggers
4.Control valves
5.Pumps
6.Actuators
7.Metering equipment
8.Weather station
9.Remote terminal units
10.Power supply
11.User interface
12.Security system

Software used in Smart Water Management:

Smart water management relies on various software solutions to effectively monitor, control, and optimize water resources. Some of the key software applications used in smart water management include:

1.SCADA (Supervisory Control
and Data Acquisition) Systems
2.IOT platforms
3.Data analytics and machine learning
4.Geogarphical information system
5.Water quality monitoring software
6.Leak detection software
7.Cloud based solution
8.Security and access control
9.Asset management software
10.Reporting and visualisation tools

Python script for Smart Water Management:

You'll need to install the 'paho-mqtt' library for MQTT communication.

```
import paho.mqtt.client as mqtt
import json
import random
import time
```

```
# Define MQTT broker and topic
broker_address = "your_broker_address"
topic = "water_consumption_data"

def generate_water_data():
    return {
        "sensor_id": "sensor_1",
        "timestamp": int(time.time()),
        "water_consumption": random.uniform(1, 5)  # Simulated water consumption in liters
    }

def on_connect(client, userdata, flags, rc):
    print("Connected with result code " + str(rc))
    client.subscribe(topic)

def on_publish(client, userdata, mid):
    print("Data published")

client = mqtt.Client()
client.on_connect = on_connect
client.on_publish = on_publish

client.connect(broker_address, 1883, 60)

try:
    while True:
        water_data = generate_water_data()
        payload = json.dumps(water_data)

        client.publish(topic, payload)

        print(f"Sent data: {payload}")

        time.sleep(5)  # Send data every 5 seconds (adjust as needed)

except KeyboardInterrupt:
    client.disconnect()
    print("Disconnected")
```

End of the file

By,
Krishnaraj S