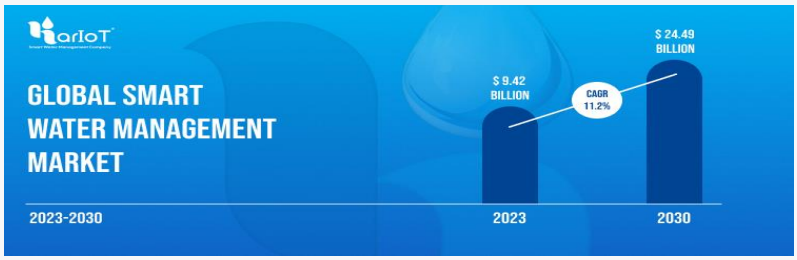
SMARY WATER MANAGEMENT BASED ON IOT

Requirements:

- **Identify the specific goals of your smart water management system (e.g., reducing water wastage, optimizing irrigation, monitoring water quality).**
- **Determine the types of sensors needed (flow sensors, moisture sensors, water quality sensors, etc.).**
- **Decide on the actuators for controlling water flow (solenoid valves, pumps).**



HARDWARE:

- **Sensors: Choose appropriate sensors based on your requirements. For example, flow sensors can measure the rate of water flow, while moisture sensors can measure soil moisture levels.**
- **Actuators: Select actuators like solenoid valves or pumps for controlling water flow based on sensor data.**
- **Microcontrollers: Use microcontrollers (Raspberry Pi, Arduino) to interface with sensors and actuators. Raspberry Pi is suitable for more complex applications, while Arduino is great for simpler tasks.**

9

PYTHON SCRIPT:

```
import RPi.GPIO as GPIO

import time

import requests

FLOW_SENSOR_PIN = 18

SOLENOID_VALVE_PIN = 17

API_ENDPOINT = "https://example.com/api/water-usage"

total_water_usage = 0  # in liters

last_measurement_time = time.time()  # timestamp of last measurement

def calculate_flow_rate(channel):

    global total_water_usage, last_measurement_time

    pulse_count = 0

    flow_rate = 0

    try:
```

```
        pulse_count += 1
        elapsed_time = time.time() - last_measurement_time
        if elapsed_time >= 1:  # Calculate flow rate every 1 second
            flow_rate = pulse_count / (elapsed_time / 60)  # in liters per minute
            total_water_usage += pulse_count / 450.0  # 1 pulse = 2.25 milliliters
            last_measurement_time = time.time()
            pulse_count = 0
    except Exception as e:
        print("Error in calculate_flow_rate:", str(e))
    return flow_rate
def control_water_flow(flow_rate):
    try:
        if flow_rate > 10:  # Example threshold flow rate (adjust based on your needs)
            GPIO.output(SOLENOID_VALVE_PIN, GPIO.HIGH)  # Turn on the solenoid valve
        else:
            GPIO.output(SOLENOID_VALVE_PIN, GPIO.LOW)  # Turn off the solenoid valve
    except Exception as e:
        print("Error in control_water_flow:", str(e))
# Main function
def main():
```

```
    try:
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(FLOW_SENSOR_PIN, GPIO.IN,
pull_up_down=GPIO.PUD_UP)  # Input pin for water flow sensor
        GPIO.setup(SOLENOID_VALVE_PIN, GPIO.OUT)  # Output pin
for solenoid valve


        GPIO.add_event_detect(FLOW_SENSOR_PIN, GPIO.FALLING,
callback=calculate_flow_rate, bouncetime=20)  # Setup event
detection


        while True:
            flow_rate = calculate_flow_rate(FLOW_SENSOR_PIN)
            print("Flow Rate: {:.2f} L/min".format(flow_rate))
            print("Total Water Usage: {:.2f}
liters".format(total_water_usage))


            # Send water usage data to the API endpoint
            payload = {
                "flow_rate": flow_rate,
                "total_usage": total_water_usage
            }
            response = requests.post(API_ENDPOINT, json=payload)
            if response.status_code == 200:
                print("Data sent successfully!")
```

```
            else:
                print("Failed to send data!"
            control_water_flow(flow_rate)  # Control water flow based
on usage
            time.sleep(60)  # Wait for 1 minute before the next
measurement
    except KeyboardInterrupt:
        print("Monitoring stopped by the user.")
    finally:
        GPIO.cleanup()
# Run the main function if the script is executed
if __name__ == "__main__":
    main()
```

PROJECT SUBMITTED BY:

NAME : DEEPAN.S

REGISTER NO: 713921106009

TOPIC: SMART WATER MANAGEMENT BASED ON IOT

.

NM ID:au713921106009

Email:srdeepan2110@gmail.com

College code :7139

MAIL ID: sreenu1713257@gmail.com

COLLEGE CODE:7139