PHASE 3

DEVELOPMENT PART 1

SMART WATER FOUNTAINS

CHOOSE IOT SENSORS:

Select appropriate sensors based on your requirements (e.g., flow rate sensors, pressure sensors) to monitor the water fountain.

INSTALL SENSORS:

Physically install the chosen sensors at relevant locations in the water fountain system to measure parameters like water flow and pressure accurately.

IOT PLATFORM SELECTION:

Choose an IoT platform to collect, process, and visualize the sensor data. Examples include AWS IoT, Google Cloud IoT, or Microsoft Azure IoT.

ESTABLISH COMMUNICATION:

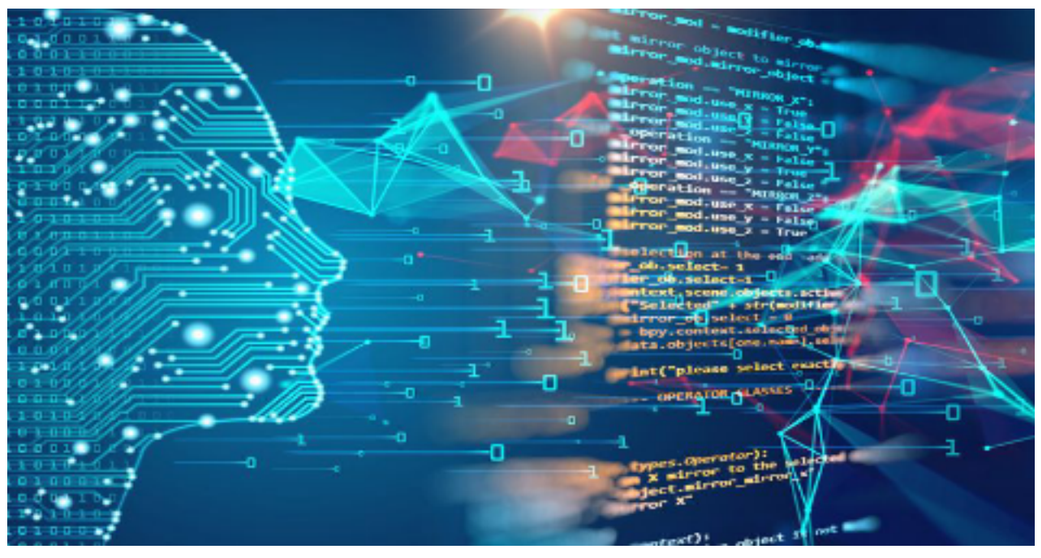
Develop a communication protocol (e.g., MQTT, HTTP) for the IoT sensors to transmit data to the selected IoT platform.

DEVELOP PHYTHON SCRIPT FOR SENSORS:

Write a Python script for each IoT sensor to read data and send it to the IoT platform. Use libraries like paho-mqtt for MQTT communication.

DATA PROCESSING AND ANALYSIS:

On the IoT platform, set up data processing pipelines to analyse incoming sensor data. Implement algorithms for fault detection and real-time monitoring.



INTEGRATION AND DASHBOARD CREATION:

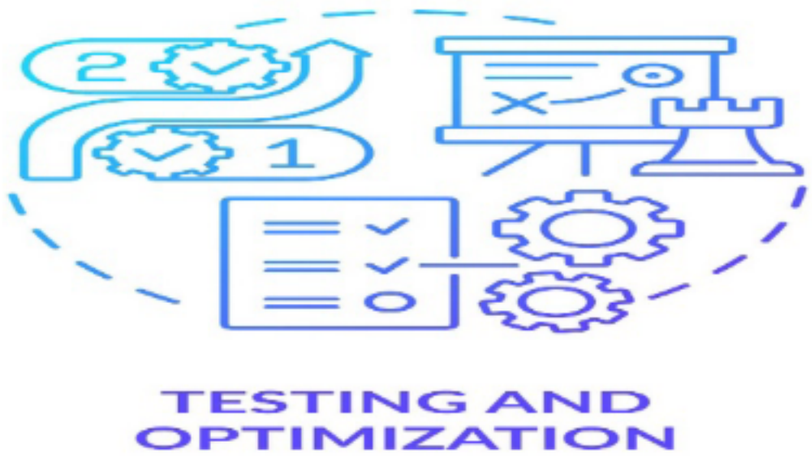
Integrate the sensor data with a dashboard application (web or mobile) to visualize the water fountain status in real-time.

USER NOTIFICATIONS AND ALERTS:

Implement a notification system to alert administrators or users in case of malfunctions or unusual events detected by the sensors.

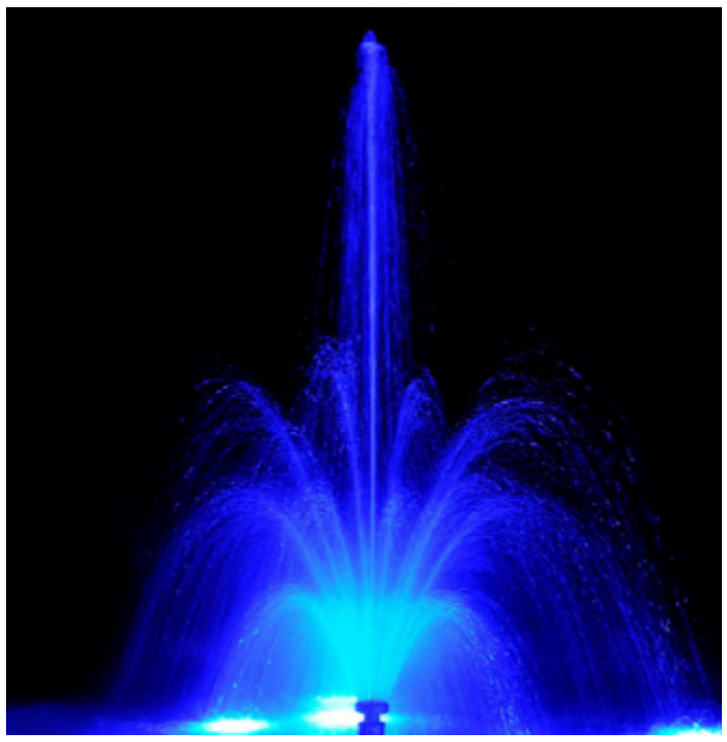
TEST AND OPTIMIZE:

Conduct thorough testing to ensure the system's reliability and efficiency. Optimize the code and system for better performance.



DEPLOYMENT:

Deploy the complete IoT-enabled Smart Water Fountain system in public water fountains.



PHYTHON SCRIPT:

```
import paho.mqtt.client as mqtt

import json

import time

from random import uniform


# Sensor ID and topic to publish data
```

```
sensor_id = "sensor1"

mqtt_topic = "water_fountain_data"


# MQTT broker details

broker_address = "mqtt.example.com"

broker_port = 1883


client = mqtt.Client(sensor_id)


def on_connect(client, userdata, flags, rc):

    print("Connected with result code " + str(rc))


def on_publish(client, userdata, mid):

    print("Message published.")


client.on_connect = on_connect

client.on_publish = on_publish


client.connect(broker_address, broker_port, 60)


while True:

    # Simulated data (replace with actual sensor readings)

    water_flow = round(uniform(0.5, 5.0), 2)  # Simulate water flow rate

water_pressure = round(uniform(20, 60), 2)  # Simulate water pressure


# Create a JSON payload with sensor data

payload = {

    "sensor_id": sensor_id,

    "water_flow": water_flow,
```

```
        "water_pressure": water_pressure
    }
# Publish data to the MQTT broker
    client.publish(mqtt_topic, json.dumps(payload))

    # Delay between readings
    time.sleep(5)
```

This Python script simulates data from the sensors and publishes it to the specified MQTT topic.

Adapt and modify the script according to the actual sensors you are using and the specified IoT platform you have chosen.