Smart water fountains

# Components for Smart Fountain

1.water fountain

2.iot sensor

3.microcontroller

4.wifi Bluetooth

5.mobile app (or)Web interface

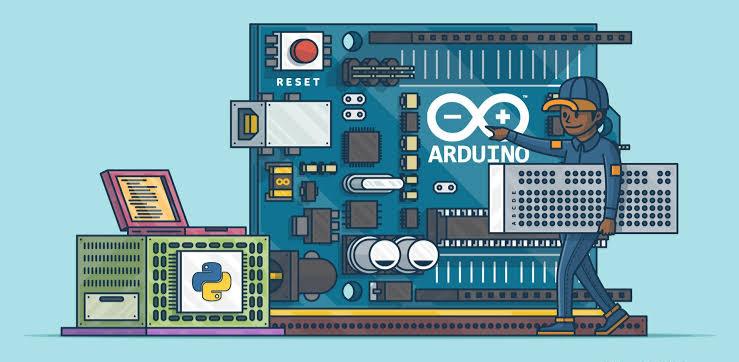
6.power supply

7.python programme

Working

Creating a smart water fountain using IoT typically involves several components, including:

1. Water Fountain: The physical water fountain itself is the core component. It can be a traditional fountain with IoT enhancements or a custom-built design.
2. IoT Sensors: Various sensors can be used to monitor and control aspects of the fountain, such as water level sensors, water quality sensors, and temperature sensors.
3. Microcontroller or Single-Board Computer: You'll need a microcontroller (e.g., Arduino, Raspberry Pi) to interface with the sensors and control the fountain's functions.
4. Connectivity Module: To make the fountain IoT-enabled, you need a communication module (e.g., Wi-Fi, Bluetooth, LoRa) to connect the microcontroller to the internet.
5. IoT Platform: An IoT platform or cloud service is necessary to store and manage the data collected from the fountain. Popular platforms include AWS IoT, Azure IoT, and Google Cloud IoT.
6. Mobile App or Web Interface: Create a user interface for controlling the fountain remotely, allowing users to turn it on/off or change settings.
7. Power Supply: Depending on the fountain's location, you might need a power source or a battery with charging capabilities.
8. Security: Implement security measures to protect the IoT system from unauthorized access and potential cyber threats.
9. Data Analytics: You can integrate data analytics tools to gain insights from the collected data, such as water consumption patterns.
10. Notification System: Set up notifications, such as alerts when the water level is low or if there are maintenance issues.
11. Maintenance and Support: Consider ongoing maintenance to ensure the fountain's proper functioning and support for users.



**# Python code for a smart water fountain using IoT**

water\_level\_pin = 17

pump\_pin = 18

GPIO.setmode(GPIO.BCM)

GPIO.setup(water\_level\_pin, GPIO.IN)

GPIO.setup(pump\_pin, GPIO.OUT)

mqtt\_broker = "mqtt.example.com"

mqtt\_port = 1883

mqtt\_topic = "water\_fountain"

client = mqtt.Client("WaterFountainClient")

def on\_connect(client, userdata, flags, rc):

print("Connected to MQTT broker with result code " + str(rc))

client.subscribe(mqtt\_topic)

def on\_message(client, userdata, message):

payload = message.payload.decode()

if payload == "on":

turn\_on\_fountain()

elif payload == "off":

turn\_off\_fountain()

def turn\_on\_fountain():

GPIO.output(pump\_pin, GPIO.HIGH)

def turn\_off\_fountain():

GPIO.output(pump\_pin, GPIO.LOW)

client.on\_connect = on\_connect

client.on\_message = on\_message

client.connect(mqtt\_broker, mqtt\_port, 60)

client.loop\_start()

while True:

water\_level = GPIO.input(water\_level\_pin)

client.publish(mqtt\_topic, "Water level: " + str(water\_level))

time.sleep(10)

except KeyboardInterrupt:

GPIO.cleanup()

