Phase 3: Development part 1

***Smart water fountain**:-*

Here are some technology requirements:

=>Water Pump:
Need a water pump to circulate and deliver water to the fountain's spout.

=>Water Reservoir:
container to hold the water, which can be refilled or connected to a water source.

=>Sensors:
Use water level sensors to monitor the water level in the reservoir. This can help you maintain the right water level and prevent pump damage.

=>Microcontroller:
To control the fountain's operations need a microcontroller like Arduino or Raspberry Pi.

=>Actuators:
These are components that control the water flow and fountain features. For instance, servos or solenoid valves can control the height and direction of water streams.

=>Power Supply:
Ensure a stable power supply for the water pump, microcontroller, and sensors. Battery or mains power may be suitable.

=>Connectivity:
To make the fountain smart, include Wi-Fi or Bluetooth modules to enable remote control and monitoring.

=>User Interface:
Can add a user interface, such as a touch screen or mobile app, for users to control the fountain.

=>Programming:
Need to write code to control the components, read sensor data, and create the fountain's behavior.

=>Enclosure:
Consider an enclosure to protect the electronic components from water and environmental elements.

=>Safety :
Implement safety features like automatic shutoff in case of low water levels or component malfunctions.

=>Data Storage and Analytics:
If it's a large-scale installation, want to collect data on water consumption and usage patterns for analysis.

=>Remote Monitoring and Control:
Consider cloud connectivity to monitor and control the fountain remotely.

2023-10-17

Python script:

```
import RPi.GPIO as GPIO
import time
import paho.mqtt.client as mqtt

# GPIO setup
GPIO.setmode(GPIO.BCM)
pump_pin = 17
GPIO.setup(pump_pin, GPIO.OUT)

# MQTT setup
broker_address = "mqtt_broker_address"
client = mqtt.Client("SmartWaterFountain")

def on_connect(client, userdata, flags, rc):
   if rc == 0:
      print("Connected to MQTT Broker")

client.subscribe("smart_water_fountain/
control")

def on_message(client, userdata, message):
   payload =
message.payload.decode("utf-8")
   if payload == "on":
      GPIO.output(pump_pin, GPIO.HIGH)
      print("Water pump is ON")
   elif payload == "off":
      GPIO.output(pump_pin, GPIO.LOW)
      print("Water pump is OFF")

client.on_connect = on_connect
client.on_message = on_message

# Connect to the MQTT broker
client.connect(broker_address)

try:
   while True:
      client.loop_start()
      time.sleep(1)

except KeyboardInterrupt:
   print("Exiting...")
   GPIO.cleanup()
   client.loop_stop()
   client.disconnect()
```

Output:

```
Connected to MQTT Broker
Water pump is ON
Water pump is OFF
...
Exiting...
```

In this script:

=>> SET UP THE GPIO PIN TO CONTROL THE WATER PUMP.

=>>CONFIGURE AN MQTT CLIENT TO CONNECT TO MQTT BROKER (REPLACE "YOUR_MQTT_BROKER_ADDRESS" WITH YOUR ACTUAL BROKER ADDRESS).

=>>THE `ON_CONNECT` AND `ON_MESSAGE` FUNCTIONS DEFINE WHAT HAPPENS WHEN THE CLIENT CONNECTS TO THE BROKER AND RECEIVES MESSAGES.

=>>INSIDE THE MAIN LOOP, THE SCRIPT LISTENS FOR MQTT MESSAGES ON THE "WATER_FOUNTAIN/CONTROL" TOPIC. WHEN IT RECEIVES "ON," IT TURNS ON THE PUMP, AND WHEN IT RECEIVES "OFF," IT TURNS OFF THE PUMP.

=>>THE SCRIPT CONTINUOUSLY LOOPS WHILE HANDLING MQTT MESSAGES UNTIL YOU MANUALLY STOP IT WITH A KEYBOARDINTERRUPT (CTRL+C).

Output explanation:

=>>THE PROGRAM IS INTENDED TO CONTROL A WATER PUMP USING THE RASPBERRY PI GPIO PINS BASED ON MESSAGES RECEIVED FROM AN MQTT BROKER. WHEN THE "ON" MESSAGE IS RECEIVED, THE WATER PUMP IS TURNED ON, AND WHEN THE "OFF" MESSAGE IS RECEIVED, THE WATER PUMP IS TURNED OFF.

=>>THE PROGRAM FIRST ESTABLISHES A CONNECTION TO THE MQTT BROKER, AND WHEN A MESSAGE IS RECEIVED ON THE "SMART_WATER_FOUNTAIN/CONTROL" TOPIC, IT EITHER TURNS THE WATER PUMP ON OR OFF AND PRINTS THE CORRESPONDING MESSAGE.

=>>THE PROGRAM RUNS IN AN INFINITE LOOP UNTIL IT IS INTERRUPTED BY A KEYBOARD INTERRUPT (E.G., PRESSING CTRL+C), AT WHICH POINT IT CLEANS UP THE GPIO SETUP AND DISCONNECTS FROM THE MQTT BROKER.