Smart water fountains

Objective:

The specific objectives of the project may vary depending on the goals of the municipality, organization, or individuals responsible for implementing the smart water fountain. Ultimately, the aim is to create a technologically advanced and environmentally responsible water feature that enhances public spaces, conserves resources, and provides an enjoyable experience for all.

Components You’ll Need :

1)Water Pump: To create the fountain effect.

2)Microcontroller: A popular choice is the Arduino, but you can also use Raspberry Pi, ESP8266, or ESP32**.**

3)Relay or Motor Driver: To control the water pump.

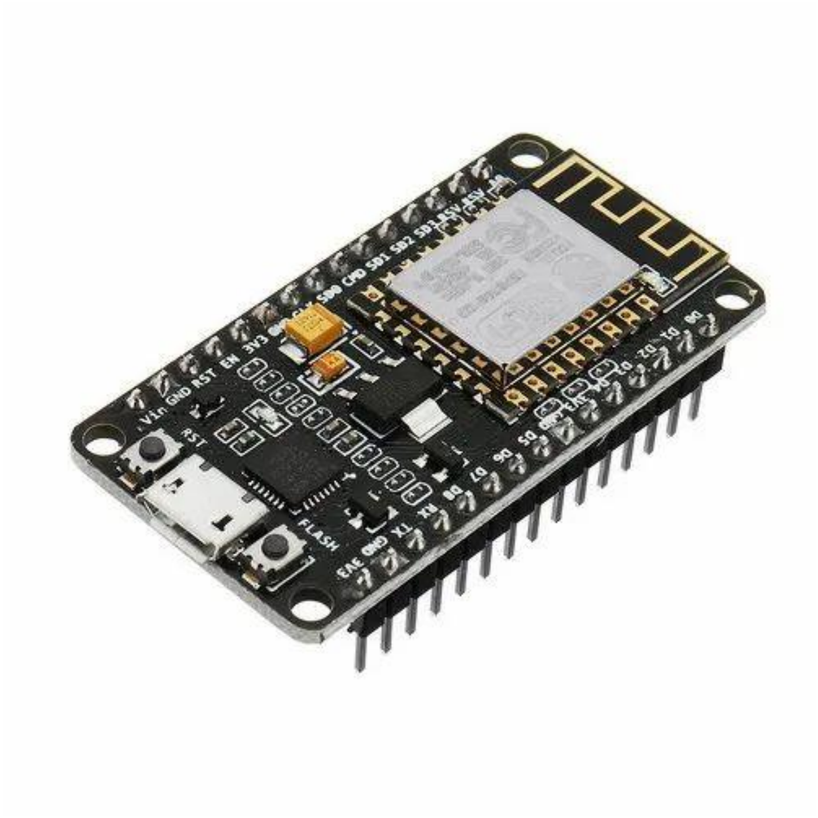
4)Wi-Fi Module: Such as an ESP8266 or ESP32 for IoT connectivity.

5)Power Supply: To provide power to the components.

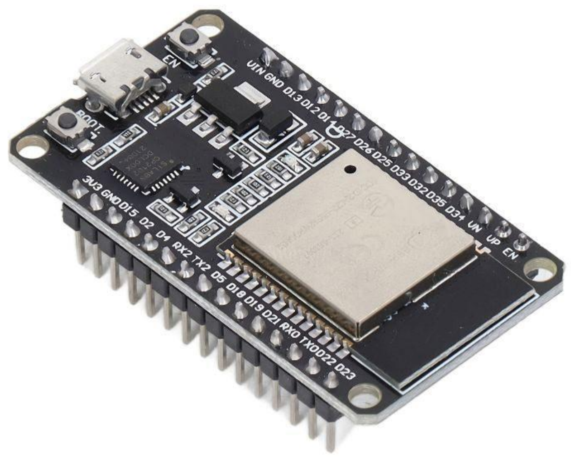
6)Web Interface: To control the fountain remotely.

IOT DEVICES:

ESP8266:



ESP32:


EN
BOOT
VIN GND D13 D12 D14 D27 D26 D25 D33 D32 D35 D34 VN VP EN
3V3 GND D15 D2 D4 RX2 TX2 D5 D18 D19 D21 RX0 TX0 D22 D23

Working principle:

1)Connect the relay module to the Arduino, water pump, water level sensor.

2)Power the components appropriately.

3)Write Arduino code to control the water pump and read the water level sensor.

4)Implement a simple communication protocol for the Arduino to receive commands from the web server.

5)Store data such as water usage and user preferences in a database.

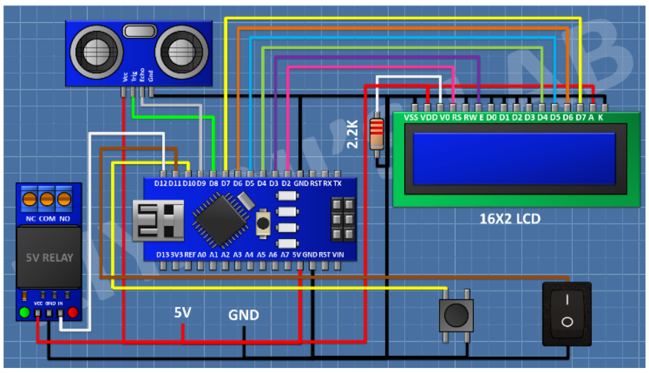
6)Implement security measures to prevent unauthorized access to the system.

7)Thoroughly test the system to ensure it works as expected.

8)Set up monitoring for the system's health.

9)Regularly maintain and update the software and hardware

CIRCUIT DESIGN:


Vcc Trig Echo Gnd
2.2K
VSS VDD V0 RS RW E D0 D1 D2 D3 D4 D5 D6 D7 A K
16X2 LCD
D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 GND RST RX TX
D13 3V3 REF A0 A1 A2 A3 A4 A5 A6 A7 5V GND RST VIN
NC COM NO
5V RELAY
VCC GND IN
5V
GND

Python Code:

```
   import RPi.GPIO as GPIO
import time

# Set the GPIO mode to BCM
GPIO.setmode(GPIO.BCM)

# Define GPIO pins for the water pump and water level
sensor
water_pump_pin = 17
water_level_sensor_pin = 18

# Set up the GPIO pins
GPIO.setup(water_pump_pin, GPIO.OUT)
GPIO.setup(water_level_sensor_pin, GPIO.IN)

# Function to turn on the water pump
def turn_on_pump():
  GPIO.output(water_pump_pin, GPIO.HIGH)

# Function to turn off the water pump
```

```
def turn_off_pump():
    GPIO.output(water_pump_pin, GPIO.LOW)

try:
    while True:
        # Check the water level
        if GPIO.input(water_level_sensor_pin) == GPIO.LOW:
            print("Water level is low. Turning on the pump.")
            turn_on_pump()
        else:
            print("Water level is sufficient. Turning off the pump.")
            turn_off_pump()

        time.sleep(1)  # Check water level every second

except KeyboardInterrupt:
    # Clean up and exit on Ctrl+C
    GPIO.cleanup()
```

Conclusion:

This simple smart fountain project will give you a hands-on experience with IoT and basic hardware integration. As you become more comfortable with these concepts, you can consider adding more features, such as scheduling, remote monitoring, or integration with other IoT devices.