SMART WATER FOUNTAINS
IoT Phase – 3
Development part - 1

Setup process :

It includes the following steps to setup iot platform for smart water fountains.

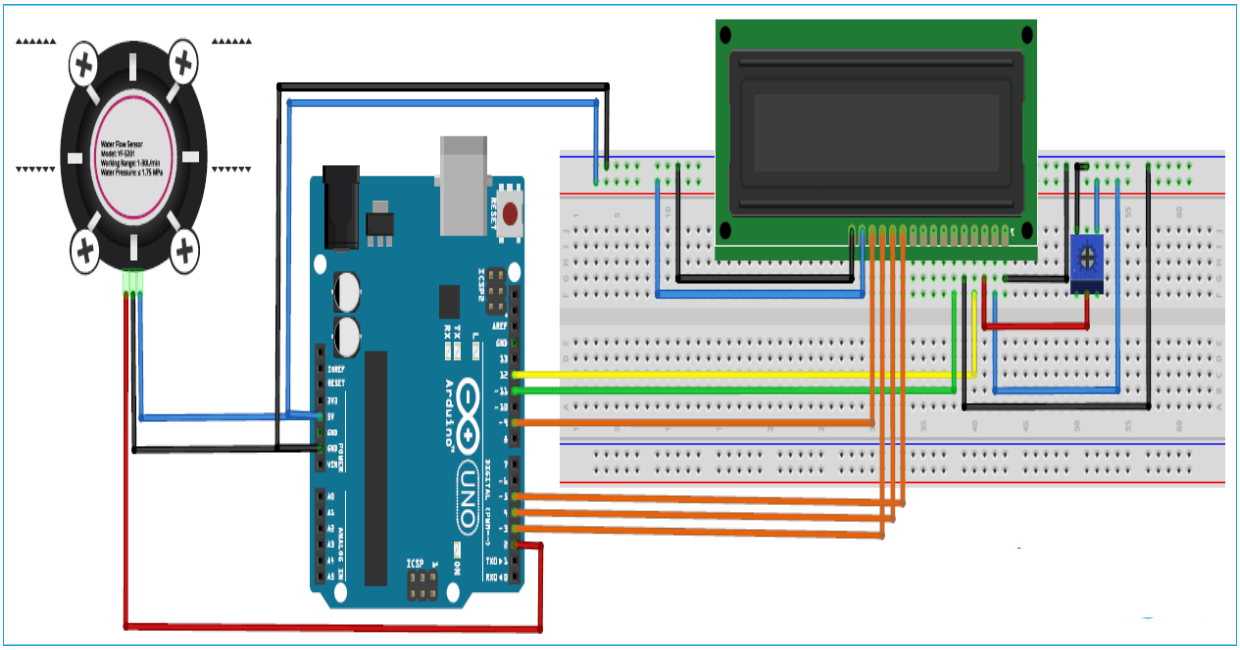
1. **Defining Project Requirements:**
    - Begin by clearly defining the objectives of your Smart Water Fountains project. What specific data are you looking to collect and analyze? What problems are you trying to solve? Understanding your project's goals is essential.
2. **IoT Device Deployment:**
    - Choose the appropriate IoT devices for your project. These devices should be capable of collecting, processing, and transmitting data to a central server or cloud platform.
    - Deploy these IoT devices strategically in the areas where water management is crucial. Ensure they are properly powered and connected to the internet.
3. **Developing Python Script:**
    - Develop a Python script to interface with the sensors and IoT devices. This script should be able to collect data from the sensors, process it, and transmit it to a central database or cloud platform for analysis.
    - Ensure the script is robust, capable of handling data from multiple sensors, and includes error-handling mechanisms.
4. **Data Analysis and Visualization:**
    - Set up a data analysis platform that can receive data from your IoT devices and perform relevant analyses. Tools like Python libraries

(e.g., Pandas, Matplotlib, Seaborn) can be used for data analysis and visualization.

5. **Remote Monitoring and Alerts:**
    - Implement remote monitoring and alerting systems to notify relevant stakeholders when specific water parameters go beyond predefined thresholds. This can be done through emails, SMS, or mobile apps.
6. **Documentation and Assessment:**
    - Create comprehensive documentation of your project, including details about the deployed IoT devices, sensors, the Python script, data analysis, and any findings or insights.

Setting up the following sensors :

Flow Sensor :



Code :

```
flow_frequency = 0  # Measures flow sensor pulses
    vol = 0.0
    l_minute = 0.0
    flowsensor = 2  # Sensor Input
    currentTime = 0
    cloopTime = 0
    from liquidcrystal import LiquidCrystal
    lcd = LiquidCrystal(12, 11, 5, 4, 3, 9)
    def flow():  # Interrupt function
        global flow_frequency
        flow_frequency += 1
    def setup():
        global flowsensor, currentTime, cloopTime
        pinMode(flowsensor, INPUT)
        digitalWrite(flowsensor, HIGH)  # Optional Internal Pull-Up
        Serial.begin(9600)
        lcd.begin(16, 2)
        attachInterrupt(digitalPinToInterrupt(flowsensor), flow, RISING)  # Setup
    Interrupt
```

```
lcd.clear()
 lcd.setCursor(0, 0)
    lcd.print("Water Flow Meter")
    lcd.setCursor(0, 1)
    lcd.print("Circuit Digest")
    currentTime = millis()
    cloopTime = currentTime
def loop():
    global currentTime, cloopTime, flow_frequency, l_minute
    currentTime = millis()
    # Every second, calculate and print litres/hour
    if currentTime >= (cloopTime + 1000):
        cloopTime = currentTime  # Updates cloopTime
        if flow_frequency != 0:
            # Pulse frequency (Hz) = 7.5Q, Q is flow rate in L/min.
            l_minute = (flow_frequency / 7.5)  # (Pulse frequency x 60 min) / 7.5Q =
flowrate in L/hour
            lcd.clear()
            lcd.setCursor(0, 0)
            lcd.print("Rate: ")
           lcd.print(l_minute)
```

Output :

Flow Rate: 3.2 L/min | Total Pulses: 450

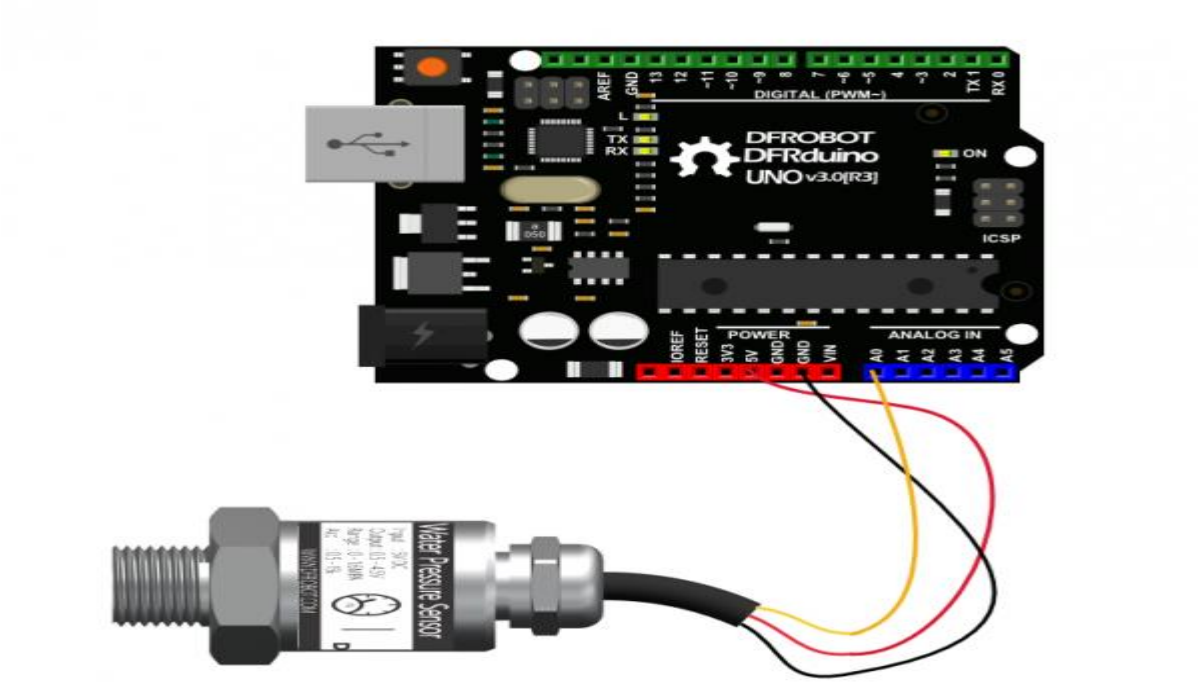
Flow Rate: 3.1 L/min | Total Pulses: 908

Flow Rate: 2.9 L/min | Total Pulses: 1356

Flow Rate: 3.0 L/min | Total Pulses: 1784

Flow Rate: 3.3 L/min | Total Pulses: 2220

Pressure sensor :



Code :

```
import time
import board
import busio
import adafruit_ssd1306

SCREEN_WIDTH = 128
SCREEN_HEIGHT = 64
OLED_RESET = -1

i2c = busio.I2C(board.SCL, board.SDA)
display = adafruit_ssd1306.SSD1306_I2C(SCREEN_WIDTH,
SCREEN_HEIGHT, i2c, addr=0x3C)

OffSet = 0.254
avgV = 0.0
V = 0
P = 0

def setup():
    print("/** Water pressure sensor demo **/")
    print("/** Water pressure sensor demo **/")
    display.fill(0)
    display.show()
    display.text(" Caliberation", 25, 0, 1)
    display.text("P:0.0KPa", 10, 20, 2)
    display.text("Pressure:", 10, 45, 2)
    display.show()

def loop():
    V = analogio.AnalogIn(board.A0).value * 5.00 / 65535
    P = (V - OffSet) * 250
    print("Voltage:", round(V, 3), "V")
    print(" Pressure:", round(P, 1), "KPa")
    display.fill(0)
    display.show()
    display.text(" Caliberation", 25, 0, 1)
    display.text("P:" + str(round(P, 1)) + "KPa", 10, 20, 2)
    display.text("Pressure:", 10, 45, 2)
    display.show()
```

```
while True:
    setup()
    loop()
    time.sleep(1)
```

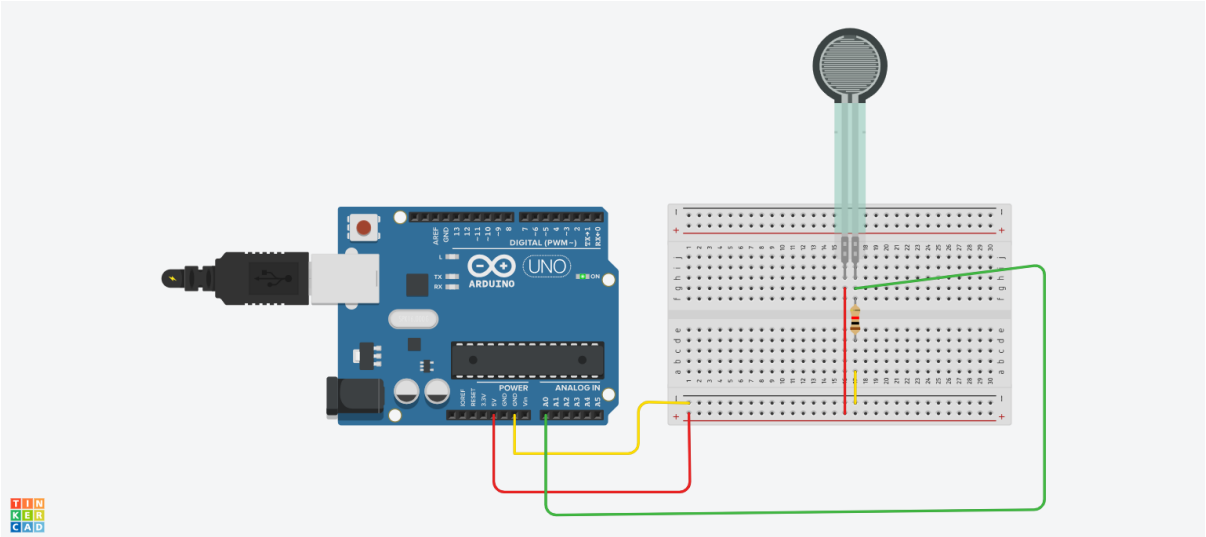
output :

```
Current Pressure: 45.2 psi

Status: Normal

Timestamp: 2023-10-18 15:30:00
```

Practical Implementation :

For practical implementation , Force sensor is used for fluid monitoring.

Force sensor has pressure sensitive function as well as water resistant function.



Force sensor has pressure sensitive function as well as water-resistant function