Develop a python script on IOT sensors to send real time water fountains status data to the platform

Certainly, developing an IoT script to send real-time water fountain status data to a platform requires several components, including IoT sensors, a microcontroller, and communication with an IoT platform. Here's a high-level Python script outline to get you started:

- ***Choose Hardware:***

Select an IoT development board (e.g., Raspberry Pi, Arduino, ESP8266, or ESP32).

Connect relevant water fountain sensors (e.g., water level sensor, flow rate sensor, etc.) to the board.

- ***Install Required Libraries:***

Depending on your hardware, install necessary libraries and dependencies for sensor communication.

- ***Set up IoT Platform:***

Sign up for an IoT platform (e.g., AWS IoT, Google Cloud IoT, or IoT platforms like ThingSpeak or Adafruit IO).

Create a device and obtain the necessary credentials (e.g., device ID, API key, or access token).

- ***Write Python Script :***

Import the required libraries for sensor readings, IoT communication, and time management.

```
import time
import Adafruit_GPIO.SPI as SPI
import Adafruit_MCP3008
import requests
import json
```

1. ***Initialize sensors:***

- Initialize and configure your sensors, and define functions to read sensor data. This example uses an ADC and a mock water level sensor:

```
            # Initialize the ADC (MCP3008)
            CLK = 18
            MISO = 23
             MOSI = 24
            CS = 25
           mcp=Adafruit_MCP3008.MCP3008(clk=CLK, cs=CS, miso=MISO, mosi=MOSI)

 # Function to read water level (example)
def read_water_level():
   return mcp.read_adc(0)  # Replace with actual sensor reading logic
```

2. Send Data to IOT Platform :

- Create a loop to continuously read sensor data and send it to the IoT platform:

```
while True:
   water_level = read_water_level()  # Read water level sensor data

   # Create a JSON payload with the data
   data = {
      'water_level': water_level,
      'timestamp': int(time.time())
   }

   # Send data to the IoT platform
   url = 'YOUR_IOT_PLATFORM_ENDPOINT'
   headers = {'Content-Type': 'application/json'}
   response = requests.post(url, data=json.dumps(data), headers=headers)

   print('Data sent:', data)
   time.sleep(60)  # Send data every minute (adjust as needed)
```

Replace

'YOUR_IOT_PLATFORM_ENDPOINT' *with the actual endpoint of your IoT platform.*

3.Run the Script:

- *Execute the script on your IoT device to start sending data to the platform.*
- *Remember to ensure that your IoT platform is correctly configured to receive and process the data. This is a basic example, and you can expand it to include more sensors and error handling, depending on your specific project requirements.*