**IOT\_PHASE 5**

**(SMART WATER FOUNTAINS)**

**FINAL PROJECT DOCUMENTATION**

**TEAM NAME: proj\_228481\_Team\_1**

**TEAM MEMBERS:**

1. **S .BHUVANASRI**
2. **M.DEEPIKA**
3. **K.KIRUTHIKA**
4. **N.SELVAMEENA**
5. **M.SHOFIYA**

**OVERVIEW :**

This is the overview of our project here you can see the development of our smart water fountain project which was done by us in previous phases. We created a smart water fountains with Raspberry in **wokwi** simulator in this project . We created a simple simulation of water quality checker and then monitoring temperature% humidity values using **ThingSpeak app.**

**AIM OF SMART WATER FOUNTAINS :**

The main aim of a smart water fountain project is to create an efficient and interactive water feature that utilizes technology to conserve water, provide an engaging and visually appealing experience. This can involve sensor networks , data analytics, and automation to ensure sustainable water resources.

**OBJECTIVES OF SMART WATER FOUNTAINS PROJECT :**

* **Water Conservation:** To use water efficiently by incorporating features like sensors to detect demand and control water flow.
* **Energy Efficiency:** Implement energy-efficient components for pump and lighting systems, possibly utilizing solar power.
* **Interactivity:** Create an engaging experience for users by enabling features like touch-sensitive controls or mobile app integration. Water quality.
* **Monitoring:** Include sensors to monitor water quality and filter status, ensuring clean and safe water.
* **Data Collection:** Gather data on fountain usage, maintenance needs, and user preferences for optimization.
* **Sustainability:** Implement materials and practices that minimize environmental impact and support long-term sustainability.
* **Education and Information:** Provide information about the fountain's features, water conservation, and environmental benefits.
* **Cost-effectiveness:** Aim for a design that balances functionality and cost, considering both initial investment and long-term maintenance expenses.

**PROBLEM DEFINITION :**

The project aims to enhance public water fountains by implementing IoT sensors to control water flow and detect malfunctions. The primary objective is to provide real-time information about water fountain status to residents through a public platform. This project includes defining objectives, designing the IoT sensor system, developing the water fountain status platform, and integrating them using IoT technology and Python.

**OBJECTIVE :**

Our goal is to design a smart water fountain that can monitor the water quality and Automatically replace water when polluted(not healthy) or running out. We will use sensors To measure the water quality. Common water quality measurement factors include Temperature, Ph-value, conductance, turbidity and hardness . Considering the pollution at Home can only affect limited factors, we choose temperature, Ph-value and conductance to Be the three properties used for calculating water quality in our water fountain. These data Will be collected, calculated, and reflected to the user in terms of “Good”, “Average” and “Bad”. The water fountain is also designed to self-filter the water every time when water is Pumped through the submersible water pump.

**ALGORITHM STEPS FOR SMARTWATER FOUNTAIN PROJECT :**

* Data collection for smart water fountain.
* Data processing.
* Feature Engineering for smart water fountain.
* Model selection process.
* Training the model process.
* Real-time monitoring for smart water fountain.
* Anomaly Detection for Smart water fountain.
* Alerting and preventive actions for smart water fountain.
* Evaluation and feedback from smart water fountain.
* Deployment and maintenance process for smart water fountain.

This adapted algorithm is designed to help maintain the optimal operation of a smart water fountain by proactively identifying anomalies or potential problems before they lead to significant malfunctions.

**PYTHON CODE FOR RASPBERRY PI INTEGRATION :**

To build an IOT-enabled Smart Water Fountain system , need to deploy sensors, collect data and send it to a platform for analysis.

**Hardware setup :**

1.Flow Rate Sensor: Measure the flow of water in the fountain.

2. Humidity and temperature monitoring : These monitors is used to detect the temperature and humidity level of water quality.

3. Microcontroller (e.g., Raspberry Pi, Arduino): Interface with sensors and send data to the platform.

4. Internet Connectivity (Wi-Fi, GSM, LoRa, etc.): Enable communication between the IoT devices and the platform.

**Software Setup :**

Python Script on IoT Sensors:

```python

Import requests

Import time

From gpiozero import InputDevice # For reading sensors

# Define sensor pins

FLOW\_SENSOR\_PIN = 14

PRESSURE\_SENSOR\_PIN = 15

# Initialize sensors

Flow\_sensor = InputDevice(FLOW\_SENSOR\_PIN)

Pressure\_sensor = InputDevice(PRESSURE\_SENSOR\_PIN)

Def read\_sensors():

Flow\_rate = 0 # Read flow rate from flow sensor (implement logic based on your sensor)

Pressure = 0 # Read pressure from pressure sensor (implement logic based on your sensor)

Return flow\_rate, pressure

Def send\_data\_to\_platform(flow\_rate, pressure):

Data = {

“flow\_rate”: flow\_rate,

“pressure”: pressure,

“timestamp”: time.time()

}

Try:

Response = requests.post(API\_ENDPOINT, json=data)

If response.status\_code == 200:

Print(“Data sent successfully!”)

Else:

Print(“Failed to send data. Status code:”, response.status\_code)

Except Exception as e:

Print(“Error occurred while sending data:”, str€)

Def main():

While True:

Flow\_rate, pressure = read\_sensors()

Send\_data\_to\_platform(flow\_rate, pressure)

Time.sleep(1) # Send data every 1 second

If \_name\_ == “\_main\_”:

Main()

**HARDWARE COMPONENTS:**

Hardware components Used :

1)Raspberry Pi

2)LED

3)Resistor(300 ohm)

4) HC-SR04 ultrasonic distance sensor

5) Bipolar stepper motor

**USES OF THE COMPONENTS IN SMART WATER FOUNTAIN PROJECT:**

Here we can see the some specific uses of these components in a smart water fountain project:

1. **Raspberry Pi:**

* Controller: Use the Raspberry Pi as the central controller to manage the various components of the smart water fountain.
* IOT Connectivity: Enable remote control and monitoring of the fountain using Wi-Fi or other IOT protocols.
* Data Logging: Collect data on fountain operation and water levels for analysis and maintenance.

2. **LED (Light Emitting Diode):**

* Illumination: Use LEDs to add decorative and dynamic lighting effects to the water fountain.
* Status Indicators: Employ LEDs as indicators for system status, such as power, water level, or mode.

3. **HC-SR04 Ultrasonic Distance Sensor:**

* Water Level Sensing: Utilize the ultrasonic sensor to measure and monitor the water level in the fountain basin.
* Overflow Prevention: Implement an alert or shutdown mechanism to prevent overflows by detecting high water levels.

4. **Bipolar Stepper Motor:**

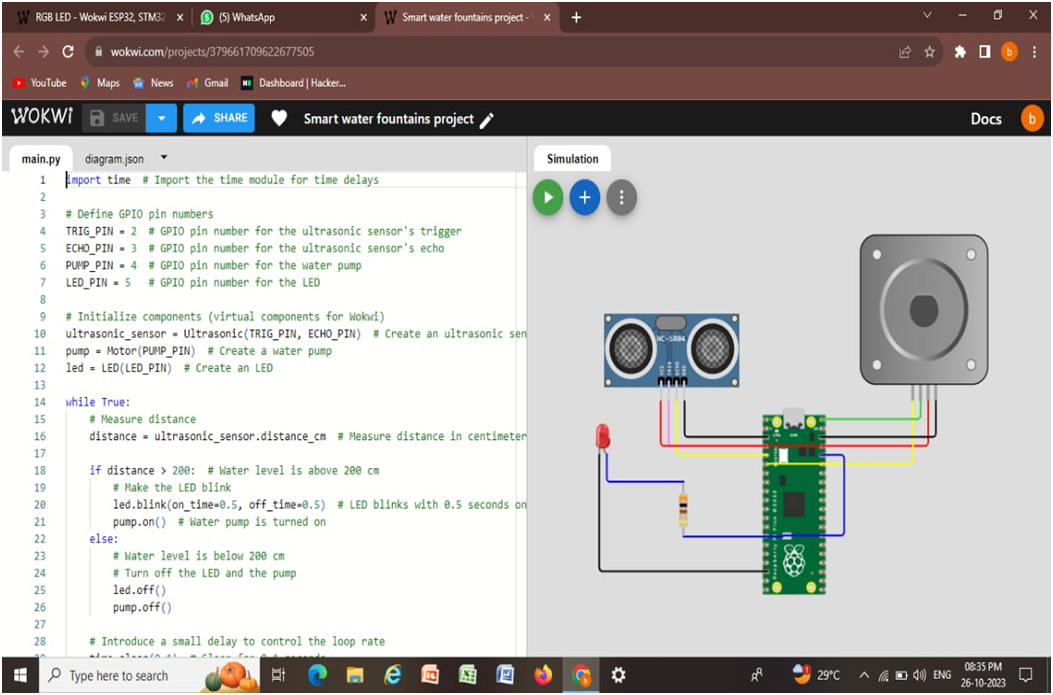
* Water Jet Control: Use the stepper motor to control the direction and angle of water jets, creating dynamic water patterns.
* Fountain Nozzle Adjustment: Adjust the position of fountain nozzles for different water displays and effects.

5. **Resistor (300 Ohm):**

* LED Current Limiting: Connect resistors in series with LEDs to limit the current and protect the LEDs in the fountain lighting system.
* Voltage Division: Use resistors in voltage divider circuits to interface with sensors or devices.

In a smart water fountain project, these components help automate and enhance the functionality and aesthetics of the fountain, providing features such as dynamic lighting, water level monitoring, and control over water patterns.

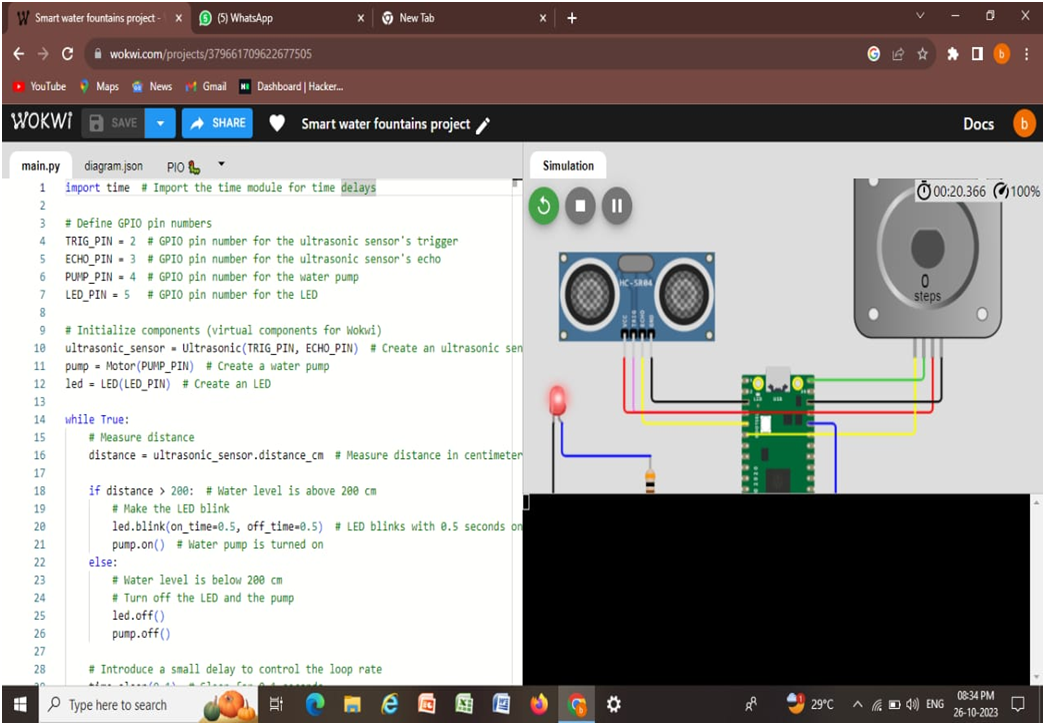
**STIMULATION DESIGN IN WOKWI :**



**Circuit design :**

* This is circuit design of our project here we used LED(red), resistor (300 ohm),ultrasonic distance sensor, raspberry pi and bipolar stepper motor .
* These components are connected according to the pin number of raspberry pi which mentioned in the code.

**OUTPUT :**



**THE PROGRAM EXECUTION CODE:**

Python program

Import time # Import the time module for time delays

# Define GPIO pin numbers

TRIG\_PIN = 2 # GPIO pin number for the ultrasonic sensor’s trigger

ECHO\_PIN = 3 # GPIO pin number for the ultrasonic sensor’s echo

PUMP\_PIN = 4 # GPIO pin number for the water pump

LED\_PIN = 5 # GPIO pin number for the LED

# Initialize components (virtual components for Wokwi)

Ultrasonic\_sensor = Ultrasonic(TRIG\_PIN, ECHO\_PIN) # Create an ultrasonic sensor

Pump = Motor(PUMP\_PIN) # Create a water pump

Led = LED(LED\_PIN) # Create an LED

While True:

# Measure distance

Distance = ultrasonic\_sensor.distance\_cm # Measure distance in centimeters

If distance > 200: # Water level is above 200 cm

# Make the LED blink

Led.blink(on\_time=0.5, off\_time=0.5) # LED blinks with 0.5 seconds on and off time

Pump.on() # Water pump is turned on

Else:

# Water level is below 200 cm

# Turn off the LED and the pump

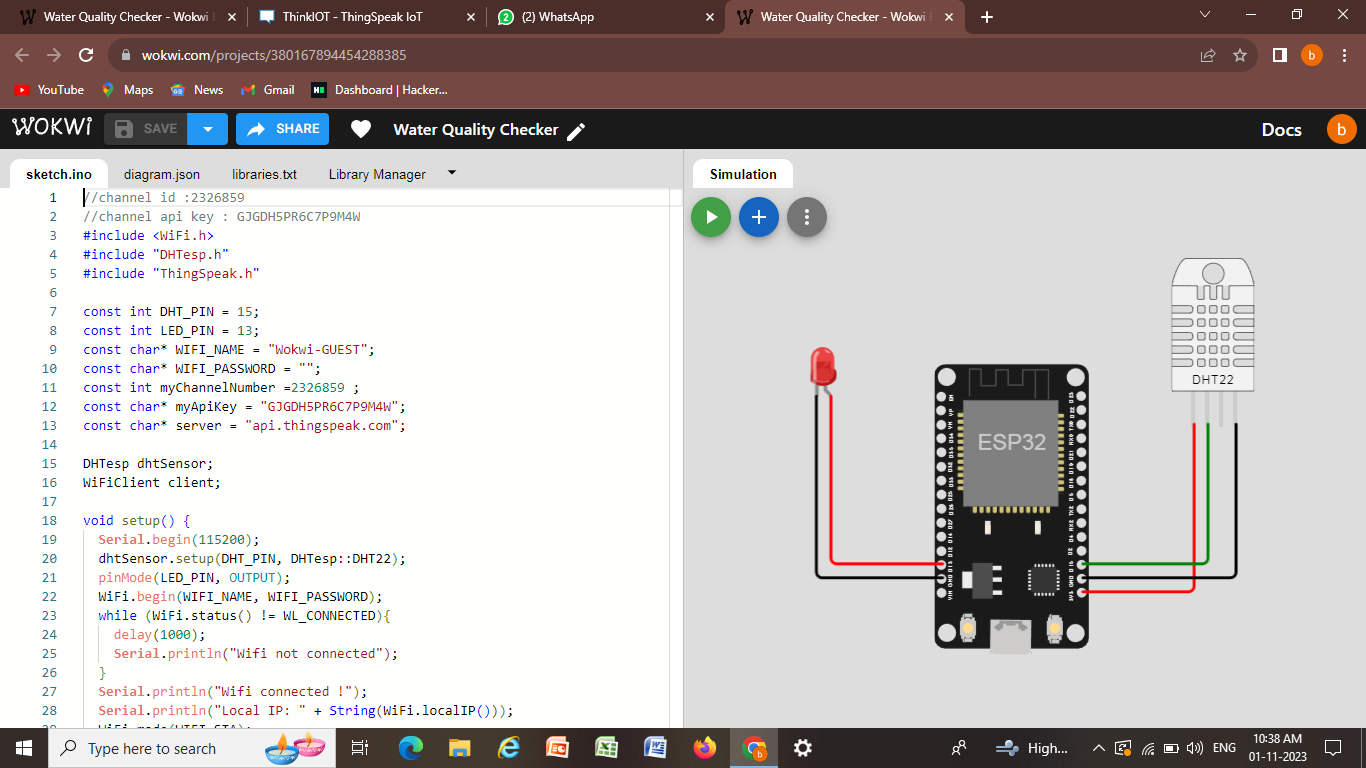
Led.off()

Pump.off()

# Introduce a small delay to control the loop rate

Time.sleep(0.1) # Sleep for 0.1 seconds

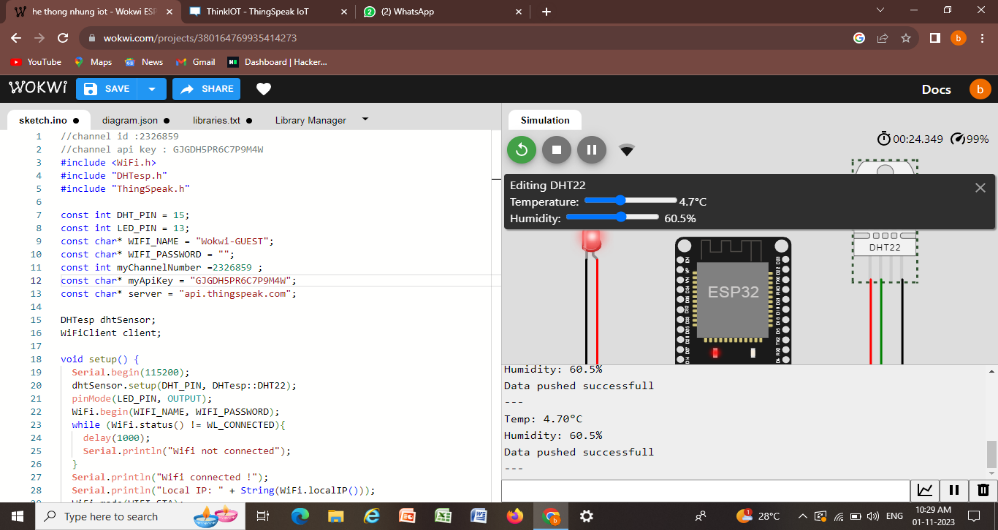
**WATER QUALITY CHECKER SIMULATION DESIGN :**



**Circuit Design :**

* Here we connected the VCC, GND and SDA pin of the DHT22 to the respectively 3V3, GND1 and D2 on the ESP32.
* We connect the Anode and cathode pin of the LED to the respectively D13 and GND2 on the ESP32.
* For this water quality checker we downloaded library like the wifi , ThingSpeak, and DHT sensor library for ESPx.

**SIMULATION OUTPUT :**



**PROGRAM EXECUTION CODE :**

//channel id :2326859

//channel api key : GJGDH5PR6C7P9M4W

#include <WiFi.h>

#include "DHTesp.h"

#include "ThingSpeak.h"

const int DHT\_PIN = 15;

const int LED\_PIN = 13;

const char\* WIFI\_NAME = "Wokwi-GUEST";

const char\* WIFI\_PASSWORD = "";

const int myChannelNumber =2326859 ;

const char\* myApiKey = "GJGDH5PR6C7P9M4W";

const char\* server = "api.thingspeak.com";

DHTesp dhtSensor;

WiFiClient client;

void setup() {

Serial.begin(115200);

dhtSensor.setup(DHT\_PIN, DHTesp::DHT22);

pinMode(LED\_PIN, OUTPUT);

WiFi.begin(WIFI\_NAME, WIFI\_PASSWORD);

while (WiFi.status() != WL\_CONNECTED){

delay(1000);

Serial.println("Wifi not connected");

}

Serial.println("Wifi connected !");

Serial.println("Local IP: " + String(WiFi.localIP()));

WiFi.mode(WIFI\_STA);

ThingSpeak.begin(client);

}

void loop() {

TempAndHumidity data = dhtSensor.getTempAndHumidity();

ThingSpeak.setField(1,data.temperature);

ThingSpeak.setField(2,data.humidity);

if (data.temperature > 35 || data.temperature < 12 || data.humidity > 70 || data.humidity < 40) {

digitalWrite(LED\_PIN, HIGH);

}else{

digitalWrite(LED\_PIN, LOW);

}

int x = ThingSpeak.writeFields(myChannelNumber,myApiKey);

Serial.println("Temp: " + String(data.temperature, 2) + "°C");

Serial.println("Humidity: " + String(data.humidity, 1) + "%");

if(x == 200){

Serial.println("Data pushed successfull");

}else{

Serial.println("Push error" + String(x));

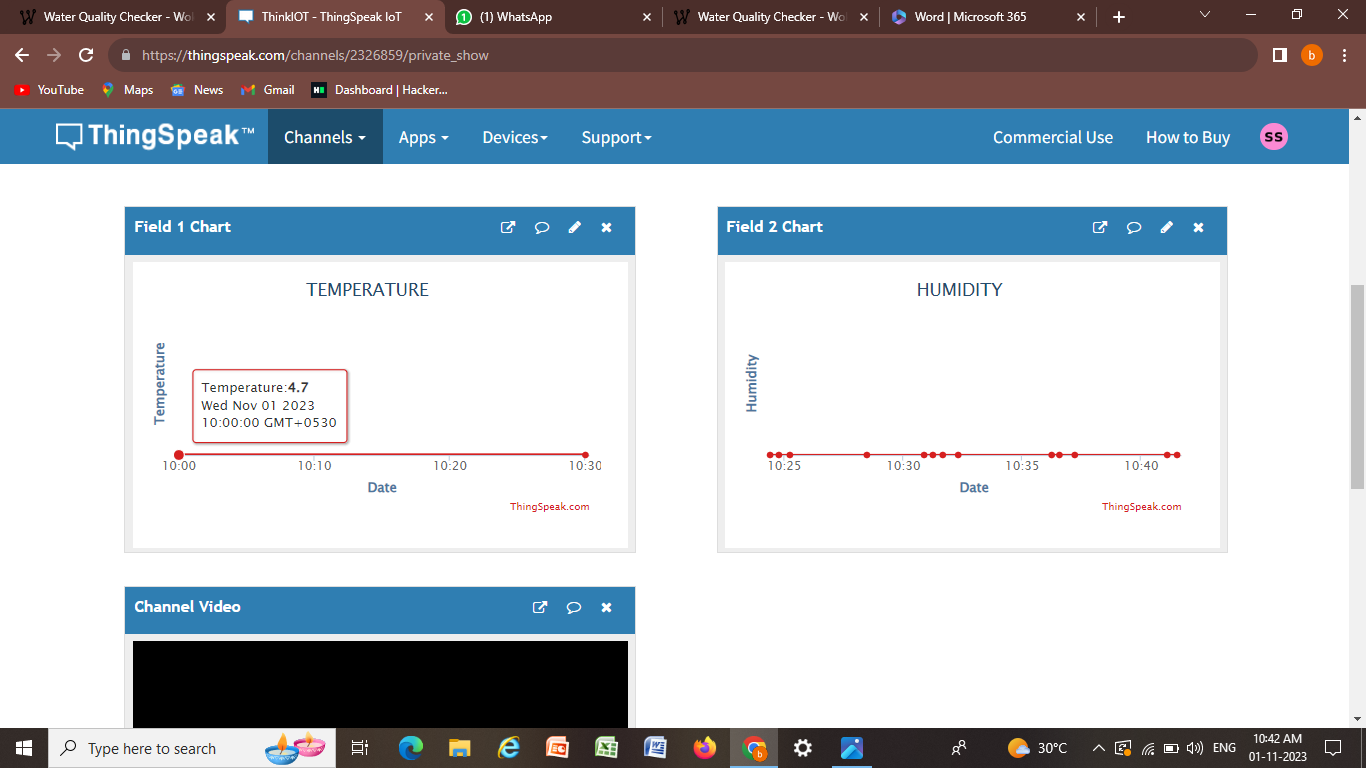
}

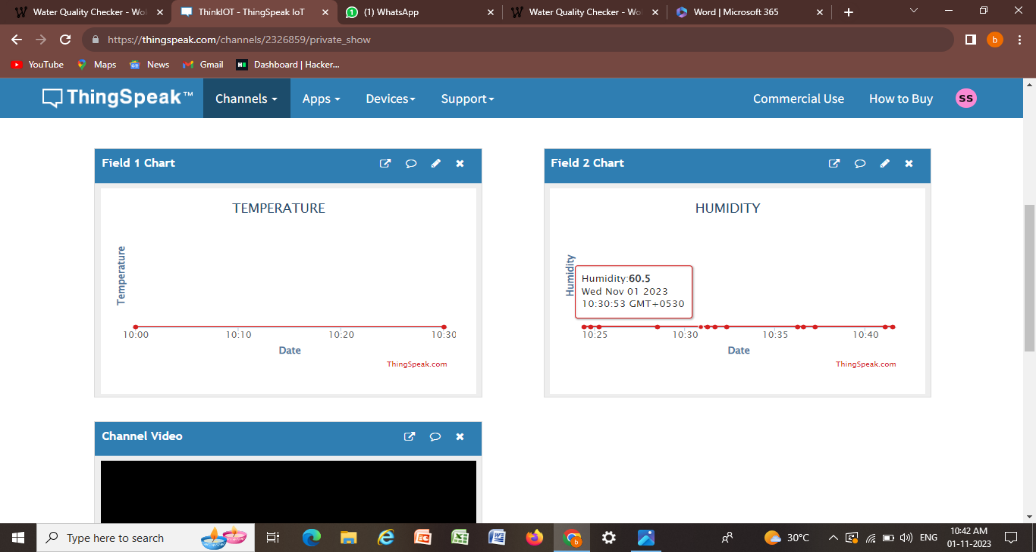
Serial.println("---");

delay(10000);}

**THINGSPEAK OUTPUT :**

Here we can get the data of temperature and humidity level of water by using the wokwi simulation.





**WEB CODE:**

<!DOCTYPE html>

<html>

<head>

<title>Water Fountain Status</title>

<link rel=”stylesheet” type=”text/css” href=”style.css”>

</head>

<body>

<header>

<h1>Water Fountain Status</h1>

</header>

<section class=”status”>

<h2>Current Status</h2>

<p id=”flowRate”>Flow Rate: <span id=”flowValue”>0</span> GPM</p>

</section>

<section class=”alerts”>

<h2>Alerts</h2>

<ul id=”alertList”>

<!—Alerts will be displayed here à

</ul>

</section>

<script src=”script.js”></script>

</body>

</html>

Body {

Font-family: Arial, sans-serif;

}

Header {

Background-color: #3498db;

Color: #fff;

Text-align: center;

Padding: 20px;

}

Section {

Margin: 20px;

Padding: 10px;

Border: 1px solid #ccc;

Border-radius: 5px;

}

H2 {

Color: #3498db;

}

Ul {

List-style: none;

}

// Simulate real-time data (replace with actual data source)

Function generateRandomData() {

Return {

flowRate: (Math.random() \* 10).toFixed(2), // Random flow rate between 0 and 10 GPM

alerts: Math.random() < 0.2 ? [“Malfunction detected”] : [],

};

}

Function updateData() {

Const data = generateRandomData();

// Update flow rate

ConstflowValue = document.getElementById(“flowValue”);

flowValue.textContent = data.flowRate;

// Update alerts

ConstalertList = document.getElementById(“alertList”);

alert List inner HTML = “”;

data.alerts.forEach((alert) => {

const li = document.createElement(“li”);

li.textContent = alert;

alertList.appendChild(li);

});

}

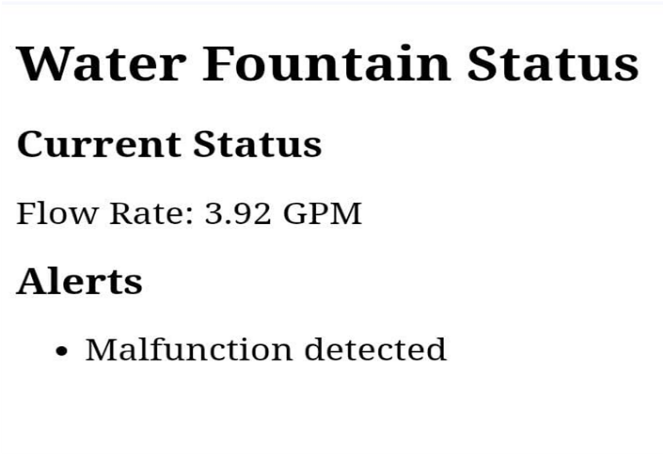
// Update data every 5 seconds (adjust as needed)

setInterval(updateData, 5000);

// Initial data update

updateData();

**OUTPUT :**



**SMART WATER FOUNTAIN SIMULATION LINK IN WOKWI :**

<https://wokwi.com/projects/379661709622677505>

**WATER QUALITY CHECKER SIMULATION LINK IN WOKWI :**

<https://wokwi.com/projects/380167894454288385>

**THINGSPEAK LINK :**

<https://thingspeak.com/channels/2326859/private_show>

**BENEFITS OF SMART WATER FOUNTAIN PROJECT :**

* **Water Savings:** IoT fountains use water more efficiently, saving resources.
* **Cost Reduction:** They lower water bills and maintenance expenses.
* **Maintenance Alerts**: They notify when repairs are needed.
* **Remote Monitoring:** You can check their status from anywhere.
* **Improved User Experience:** Touchless operation and customization.
* **Data Insights:** Learn usage patterns for better decisions.
* **Sustainability :** Contribute to environmental goals.
* **Health and Hygiene:** Reduce disease transmission risks.
* **Community Features:** Offer Wi-Fi or emergency alerts.
* **Convenience:** Provide clean, cool water on demand.

**CONCLUSION :**

IoT-based smart water fountain project signifies the culmination of its objectives and outcomes. It involves a comprehensive evaluation ofwhether the project successfully met its intended goals, whether they were related to water conservation, cost reduction, or enhancing user experiences. User feedback plays a pivotal role, as it offers insights into the practicality and user-friendliness of the smart fountain system. Additionally, data collected from IoT sensors aids in the analysis of water usage patterns and the performance of the system. A cost-benefit analysis weighs the project's financial implications, considering savings in water and maintenance costs against the initial investment. The environmental impact is assessed, determining how the project contributes to sustainability goals and resource conservation.