College code : 4212

Register number : 421221106011

**SMARTWATER FOUNTAIN USING IOT**

**PHASE:3**

Replace WATER\_SENSOR\_PIN with the GPIO pin number connected to your water level sensor.

Modify the WATER\_THRESHOLD variable based on your sensor readings to determine the water level at which the fountain should activate. “YOUR\_WATER\_API\_ENDPOINT” with the actual endpoint where you want to send the water level data (could be a cloud service or a local server).

This phase involves in designing of the steps that defining in each phase of the Previous documentation this involves developing a Python script on the IoT Devices as per the project requirement.

# Import necessary libraries

Import RPi.GPIO as GPIO

Import time

Import requests

# Set up GPIO pins and other configurations

GPIO.setmode(GPIO.BCM)

GPIO.setup(WATER\_SENSOR\_PIN, GPIO.IN)

WATER\_THRESHOLD = 100 # Adjust this threshold based on sensor readings

WATER\_API\_ENDPOINT = “YOUR\_WATER\_API\_ENDPOINT” # Replace this with your API endpoint

# Main function to monitor water level and send data to IoT platform

Def main():

While True:

# Read water level sensor data

Water\_level=GPIO.input(WATER\_SENSOR\_PIN)

# Check if water level is below the threshold

If water\_level< WATER\_THRESHOLD:

# Water level is low, activate the fountain

# Code to control the fountain (e.g., using a relay)

# Send data to IoT platform

Data = {“water\_level”: water\_level}

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

# Wait for a specific interval before checking again

Time.sleep(5) # Adjust this interval based on your requirements

# Run the main function

If \_\_name\_\_ == “\_\_main\_\_”:

Try:

Main()

Except KeyboardInterrupt:

# Handle keyboard interrupt (Ctrl+C)

Print(“Exiting the program”)

Finally:

# Clean up GPIO pins

GPIO.cleanup()