Smart water management

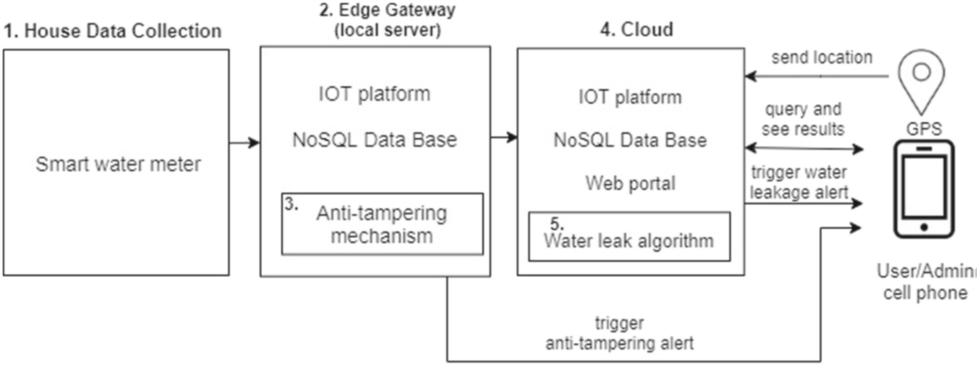
IOT \_PHASE 3

REG NO:610821106012

NAME: Chandana.S

**Configure IoT sensors to measure water consumption in public places.**

* The smart measurement system is based on the development of an architecture for IoT that covers 5 important aspects.
* The below figure shows the five main components of the system, which allow the collection, storage, analysis and visualization of water consumption.
* In the “House Data Collection” component, each time period *t*1 (can be 1 min), the value of water consumption is obtained through a smart meter, which is sent to the “Edge Gateway” component for storage.
* Within this component there is an installed “Anti-Tampering” security mechanism that alerts the user and administrator in case of manipulation of the device.
* Then, each time period *t*2 (*t*2 *> t*1, it can be 1 h), the accumulated consumption is sent to the “Cloud” server so that this value is stored together with the user’s location, which is obtained through the cell phone’s GPS, and both are analyzed by the leak detection algorithm “Water leak Algorithm,” which alerts to the user and administrator if there is a possible water leak. Also, within the “Cloud” there is a web portal that allows the user to visualize, in real time, the history of their water consumption.

**Python script on IoT sensors to send real-time water consumption data to the data-sharing platform.**

import paho.mqtt.client as mqtt

import json

import time

from random import uniform

# Define MQTT parameters

broker\_address = "your\_broker\_address"

port = 1883

topic = "water\_consumption"

# Function to simulate water consumption data

def generate\_water\_data():

return {"timestamp": int(time.time()), "flow\_rate": round(uniform(0.5, 5.0), 2)}

# Callback when the client connects to the broker

def on\_connect(client, userdata, flags, rc):

print("Connected with result code "+str(rc))

client.subscribe(topic)

# Callback when a message is published to the topic

def on\_publish(client, userdata, mid):

print("Message Published")

# Main script

client = mqtt.Client()

client.on\_connect = on\_connect

client.on\_publish = on\_publish

# Connect to the broker

client.connect(broker\_address, port, 60)

try:

while True:

water\_data = generate\_water\_data()

payload = json.dumps(water\_data)

# Publish the data to the topic

client.publish(topic, payload)

time.sleep(10) # Adjust the interval based on your requirements

except KeyboardInterrupt:

print("Script terminated by user.")

client.disconnect()