

SMART PUBLIC RESTROOM-BUILDING MY PROJECT

INTERNET OF THINGS - PHASE 3 - GROUP 1 – PROJECT

MADHA INSTITUTE OF ENGINEERING AND TECHNOLOGY

COLLEGE CODE: 2112

Reg no: 211221106003

Building a smart public restroom using IoT involves integrating various sensors, devices, and technologies to enhance the user experience, improve efficiency, and ensure cleanliness. Here's a comprehensive guide for building a project on a smart public restroom using IoT

- **Automated Entry and Exit System**

Implement RFID or biometric-based access control for secure entry and exit.

Use sensors to detect entry and exit, triggering appropriate actions and services.

- **Occupancy Monitoring**

Deploy occupancy sensors to monitor the usage of different sections of the restroom.

Display real-time occupancy status outside the restroom to guide users to available facilities.

- **Hygiene Maintenance**

Install automated soap dispensers and water faucets to minimize touch and improve hygiene.

Integrate sensors to monitor hand-sanitizer levels and automate refills.

Use UV disinfection systems or self-cleaning surfaces to maintain a clean environment.

- **Security and Privacy Measures**

Implement robust security protocols to protect user data and ensure the safety of IoT devices.

Adhere to privacy regulations and industry standards to protect user privacy and data security.

Creating a smart public restroom using IoT involves integrating various sensors and devices to monitor and control different aspects of the restroom. Here is a basic Python script to demonstrate how to control and monitor some features of a smart public restroom using the MQTT protocol, which is commonly used in IoT applications.

To run this script, you'll need to install the paho-mqtt library using pip (pip install paho-mqtt).

Python

Copy code

```
import paho.mqtt.client as mqtt
import time

# MQTT settings
mqtt_broker = "broker.example.com" # Update with your broker's IP or hostname
mqtt_port = 1883 # Default MQTT port
mqtt_keepalive = 60 # Keep-alive time for MQTT connection

# Sensors and Actuators
sensor_data = {
    "temperature": 25, # Placeholder temperature value
    "humidity": 60, # Placeholder humidity value
    "occupancy": False # Placeholder occupancy status
}

# Define MQTT callbacks
def on_connect(client, userdata, flags, rc):
```

```
if rc == 0:

    print("Connected to MQTT Broker!")

else:

    print(f"Failed to connect, return code {rc}")


def on_message(client, userdata, message):

    print(f"Received message: {message.payload.decode()} on topic {message.topic}")


# Initialize MQTT client

client = mqtt.Client("SmartRestroom")

client.on_connect = on_connect

client.on_message = on_message

client.connect(mqtt_broker, mqtt_port, mqtt_keepalive)


# Subscribe to relevant topics

client.subscribe("restroom/occupancy")


# Simulate sensor data updates and publish to MQTT broker

while True:

    # Simulate sensor data changes

    sensor_data["temperature"] += 1

    sensor_data["humidity"] -= 1

    sensor_data["occupancy"] = not sensor_data["occupancy"]


    # Publish sensor data to respective topics
```

```
client.publish("restroom/temperature", sensor_data["temperature"])
```

```
client.publish("restroom/humidity", sensor_data["humidity"])
```

```
client.publish("restroom/occupancy", str(sensor_data["occupancy"]))
```

```
time.sleep(5) # Update data every 5 seconds
```

This is a basic simulation of an MQTT client that simulates data coming from various sensors within the restroom. In a real-world scenario, you would need to integrate physical sensors to obtain actual data. Additionally, you would need to create a corresponding MQTT broker and handle the incoming messages in a meaningful way. Make sure to replace the placeholder MQTT broker address with the actual address of your broker.

Team Members:

1) 211221106003

2) 211221106004

3) 211221106005

4) 211221106006

5) 211221106007

6) 211221106003