PROJECT TITLE: SMART PUBLIC RESTROOM

TEAM MEMBER:

au723721104100:Sivaselvi.K

PROCEDURE:

Building an IoT-enabled Smart Public Restrooms system involves several components and steps.

1. *Define system requirements:* Identify the specific features and objectives of your Smart Public Restrooms system. Determine the desired functionality, such as occupancy monitoring, cleanliness tracking, automatic alerts, and data analytics.

2. *Choose sensor technologies:* Select appropriate sensors based on the requirements defined. For example, use occupancy sensors to monitor restroom usage and cleanliness sensors to assess hygiene levels.

3. *Select IoT platform:* Choose an IoT platform that can collect, manage, and analyze data from the sensors. Common platforms include AWS IoT, Azure IoT, or Google Cloud IoT. Consider factors like scalability, real-time processing capabilities, and integration with other systems.

4. *Design the system architecture:* Create a system architecture that includes the sensors, microcontrollers or IoT devices, communication protocols, and the IoT platform. Determine how the sensors will be connected to the microcontrollers and how the microcontrollers will transmit data to the IoT platform.

5. *Develop firmware/software:* Write the firmware or software for the microcontrollers or IoT devices to collect sensor data and communicate with the IoT platform. Use programming languages like C++, Python, or MicroPython depending on the chosen hardware.

6. *Connectivity and data transfer:* Configure the microcontrollers or IoT devices to establish connectivity with the internet. Set up the appropriate communication protocols like MQTT or HTTP to transmit sensor data securely to the IoT platform.

7. *Build the dashboard and analytics:* Develop a user-friendly dashboard that displays real-time occupancy, cleanliness, and other relevant data. Implement analytics and reporting features for gaining insights from the collected data.

8. *Integrate notifications and alerts:* Configure the system to generate alerts or notifications for specific events, such as detecting low cleanliness levels or when a restroom exceeds its capacity.

9. *Ensure security and privacy:* Implement appropriate security measures to protect data transmission and storage. Consider encryption, access controls, and secure authentication mechanisms. Address privacy concerns and comply with applicable regulations.

10. *Deploy and test:* Install the IoT devices in public restrooms and validate the system's functionality. Perform testing to ensure proper data collection, transmission, and analytics.

11. *Monitor and maintain:* Regularly monitor the system, ensuring that sensors remain operational, connectivity is stable, and data is being accurately collected and analyzed. Perform maintenance and address any issues promptly.

12. *Continuously improve:* Gather feedback from users and stakeholders to identify areas for improvement. Enhance the system's capabilities based on user experiences and emerging technologies.

Remember, building an IoT-enabled Smart Public Restrooms system is a complex endeavor that requires careful planning, development, and ongoing maintenance. Consider consulting with experts in IoT and building management systems to ensure a successful implementation.

IoT sensors in public restrooms for collecting data:

1. *Identify the sensor types:* Determine the types of sensors needed for your data collection. In this case, you mentioned occupancy sensors and cleanliness sensors. Ensure you select compatible sensors that can communicate with your IoT platform.

2. *Choose a microcontroller or IoT device:* Select a microcontroller or IoT device that can interface with the chosen sensors and connect to the internet. Some popular options include Arduino, Raspberry Pi, or ESP32/ESP8266-based devices.

3. *Connect and wire the sensors:* Connect the sensors to your microcontroller or IoT device according to the manufacturer's instructions. This may involve connecting power, ground, and data lines for each sensor.

4. *Install the sensors in the restroom:* Physically install the sensors in appropriate locations within the public restrooms. For occupancy sensors, you might consider mounting them near entry/exit doors or within toilet stalls. For

cleanliness sensors, they could be positioned to detect cleanliness indicators like odors or presence of cleaning agents.

5. *Develop firmware/software:* Write the firmware or software to collect data from the sensors and transmit it to your data collection platform. You can use programming languages like C++, Python, or MicroPython to develop the code according to the microcontroller or IoT device you've chosen.

6. *Set up data communication:* Establish connectivity between the microcontroller/IoT device and your data collection platform. This usually involves configuring the device to connect to Wi-Fi or a cellular network. Additionally, set up the required protocols such as MQTT or HTTP to send data from the device to your platform.

7. *Test and calibrate sensors:* Perform testing to ensure the sensors are working correctly within the public restrooms. Calibrate the sensors if necessary to ensure accurate data collection.

8. *Deploy and monitor:* Install the microcontroller/IoT device with the connected sensors in each targeted public restroom. Monitor and maintain the devices to ensure continuous data collection.

IoT sensors in public restrooms to collect data:

```
import paho.mqtt.client as mqtt

import time

broker_address = "mqtt.example.com"

broker_port = 1883

username = "your_username"

password = "your_password”
```

```
occupancy = 0  # 0 - No occupancy, 1 - Occupied
cleanliness = 0  # 0 - Dirty, 1 - Clean
def send_sensor_data():
    client = mqtt.Client()
    client.username_pw_set(username, password)
    client.connect(broker_address, broker_port)
    client.publish("restroom/occupancy", str(occupancy))
    print("Sent Occupancy:", occupancy)
   client.publish("restroom/cleanliness", str(cleanliness))
    print("Sent Cleanliness:", cleanliness)
    client.disconnect()
    while true
    occupancy = read_occupancy_data()
    cleanliness = read_cleanliness_data()
    send_sensor_data()
    time.sleep(5)
```