PROJECT TITLE: SMART PUBLIC RESTROOM

TEAM MEMBER: au723721104100: SIVASELVI.K

Project Objectives

Objective: Develop a smart public restroom system to monitor and evaluate toilet condition in real time.

Aim: To improve the hygiene level and personal cleansing experience.

IoT Sensor deployment

Sensor Selection

- Selected IoT sensors capable of monitoring water usage levels and the smell in the toilet.
- Chose sensors suitable for outdoor use and data transmissions.

Sensor Placement

- Ensured optimal sensors placement for accurate data collection.

Data transmission

- Configured sensors to transmit data to a central server using communication protocols (e.g…, Wi-fi, LoRa, cellular networks).
- Established reliable data transmission mechanisms.

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.

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.

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.

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.

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.

User Needs and Experience:

Identify the specific needs of users, including accessibility requirements, preferences for touchless fixtures, and overall comfort. - Consider user feedback and surveys to understand expectations and preferences.

Hygiene and Safety:

Prioritize hygiene and safety features, such as touchless fixtures, UV-C disinfection, and easy-to-clean surfaces. - Ensure compliance with health and safety regulations, especially in post-pandemic environments.

Sustainability Goals:

Choose restrooms with green solutions like waterefficient fixtures, energy□efficient lighting, and sustainable materials. - Evaluate options for renewable energy sources and water recycling systems.

Technology Integration:

Assess the integration of IoT (Internet of Things) technology for real□time monitoring, data analytics, and predictive maintenance. Ensure compatibility with existing building management system.

Maintenance and Durability:

Select fixtures and materials that are durable, easy to maintain, and have a long lifespan. - Consider the availability of spare parts and support services.

Cost Considerations:

Evaluate the initial installation costs and long-term operational expenses. - Calculate potential cost savings from resourceefficient features.

Environmental Impact:

Assess the overall environmental impact of the restroom, including water and energy use, emissions, and waste generation. - Aim for a restroom that minimizes its carbon footprint.

Compliance and Accessibility:

Ensure that the restroom complies with accessibility standards, such as ADA (Americans with Disabilities Act) requirements. - Review local building codes and regulations.

Vendor Reputation:

Research and choose reputable vendors with a track record of delivering quality smart restroom solutions. - Read customer reviews and seek recommendations.

Submission

GitHub Repository: https://github.com/sivakousalya/smart_restroom.git

Replication Instructions:

To replicate this project, follow these steps:

1. Deploy IoT sensors at strategic locations in public areas.

2.Set up a central server for data reception and processing. Implement APIs for data ingestion.

3.Develop Python scripts for data processing, analysis, and restroom cleanliness logic.

4.Create a web-based restroom monitoring platform using web development technologies and Python.

5.Integrate the platform with the central server for real-time data updates and notifications.

6.Test the system with mock data to ensure functionality.

7.Deploy the entire system to your chosen infrastructure (cloud or on-premises).

8.Maintain and monitor the system for continuous operation.

HTML Code (index.html)

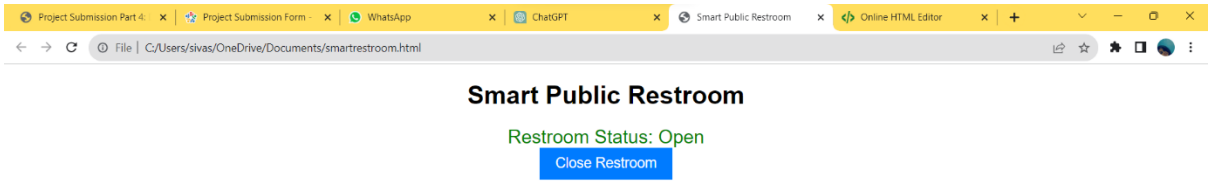
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Smart Public Restroom</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            text-align: center;
        }
        #status {
            font-size: 24px;
            color: green;
        }
        #control-button {
            font-size: 18px;
            padding: 10px 20px;
            background-color: #007BFF;
            color: #fff;
            border: none;
            cursor: pointer;
        }
    </style>
</head>
<body>
    <h1>Smart Public Restroom</h1>
    <div id="status">Restroom Status: Open</div>
    <button id="control-button">Toggle Restroom</button>
```

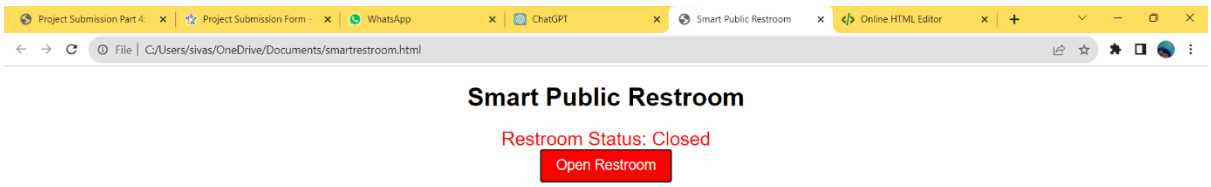
```
    <script>
        let isRestroomOpen = true;

        document.addEventListener('DOMContentLoaded', function () {
            const statusElement = document.getElementById("status");
            const controlButton = document.getElementById("control-button");

            // Initialize the status based on the initial value of
                `isRestroomOpen`
            updateStatus();

            // Add a click event listener to the control button
            controlButton.addEventListener('click', function () {
                isRestroomOpen = !isRestroomOpen;
                updateStatus();
            });

            function updateStatus() {
                if (isRestroomOpen) {
                    statusElement.textContent = "Restroom Status: Open";
                    statusElement.style.color = "green";
                    controlButton.textContent = "Close Restroom";
                    controlButton.style.backgroundColor = "#007BFF";
                } else {
                    statusElement.textContent = "Restroom Status: Closed";
                    statusElement.style.color = "red";
                    controlButton.textContent = "Open Restroom";
                    controlButton.style.backgroundColor = "red";
                }
            }
        });
    </script>
</body>
</html>
```

CSS Code: (style.css)

```css
body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
}
header {
    background-color: #4CAF50;
    color: white;
    padding: 15px;
    text-align: center;
}
.restroom-info {
    /* Styles for displaying restroom availability and cleanliness data */
}

```

RESULT:





Design app interference

- Create two labels (on off light and status).
- Add a button . Block-Based Coding:
- Use MIT App Inventor's block-based coding to program the app:
- When Screen1.Initialize:
- Initialize any necessary variables .
- When Button.Click:
- Toggle the state of the room free or valve (Open/Closed) .

• Update the status label accordingly .

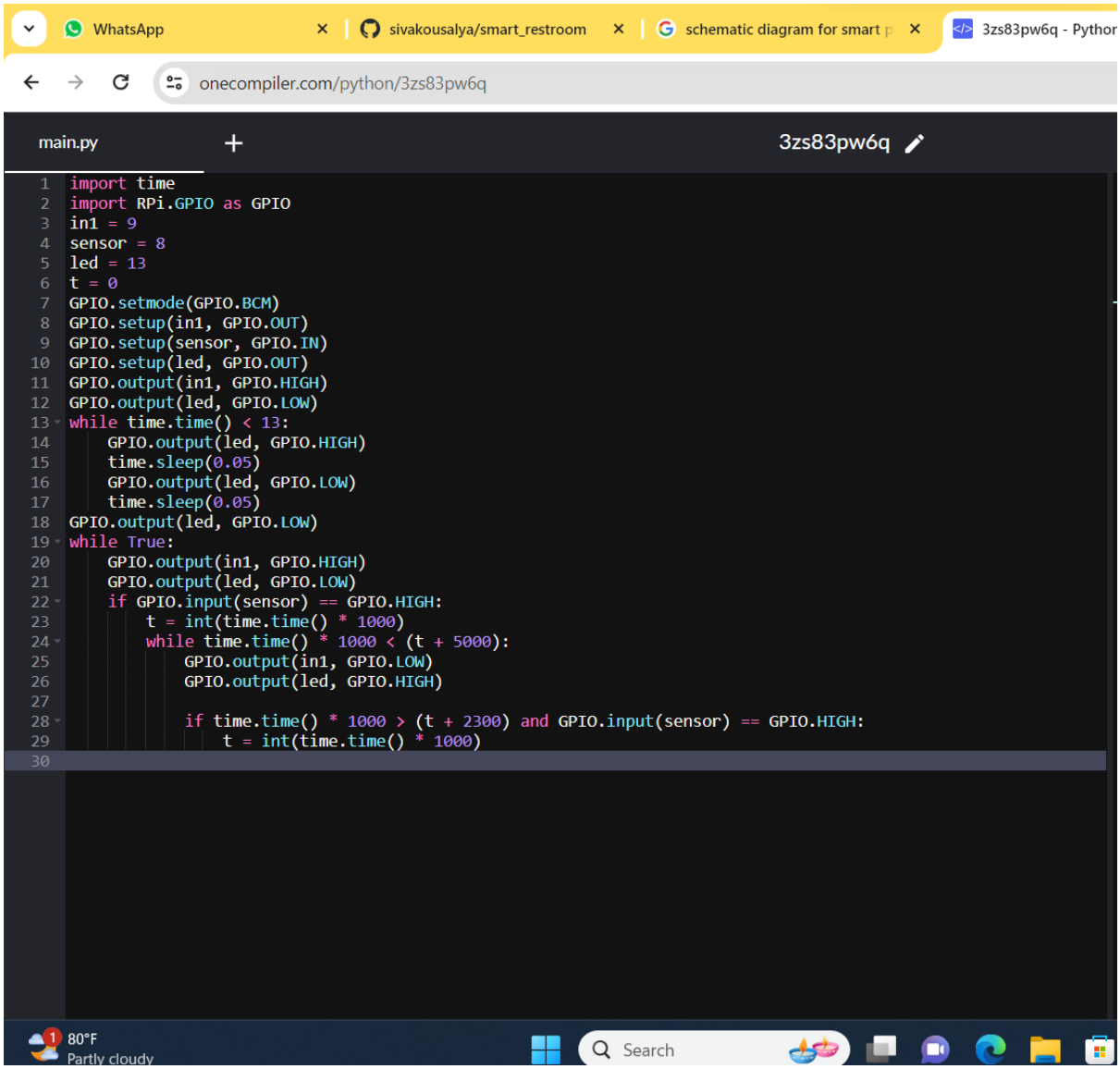
• Use a Timer component to periodically check the vacency(simulated for this example).

• Inside the Timer event handler, you can generate vacancy values as follows:

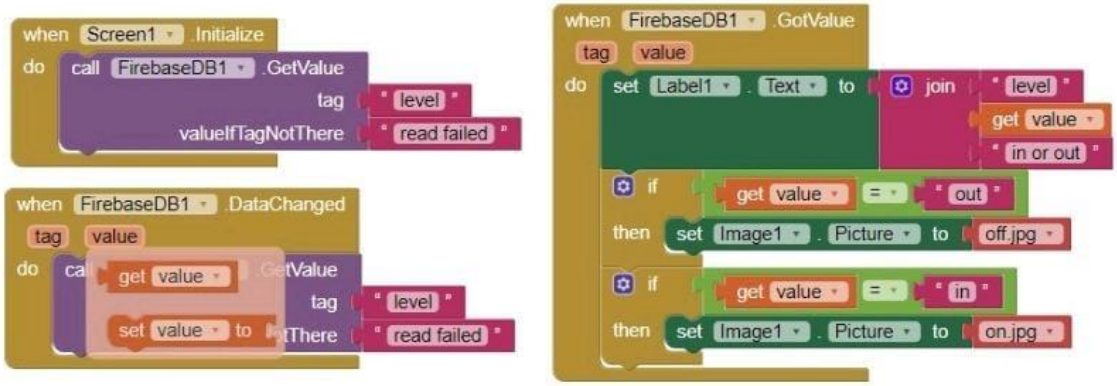
• Set a variable to a random number within a range representing the restroom.

• Display the vacency on the appropriate label.

INPUT PROGRAM:

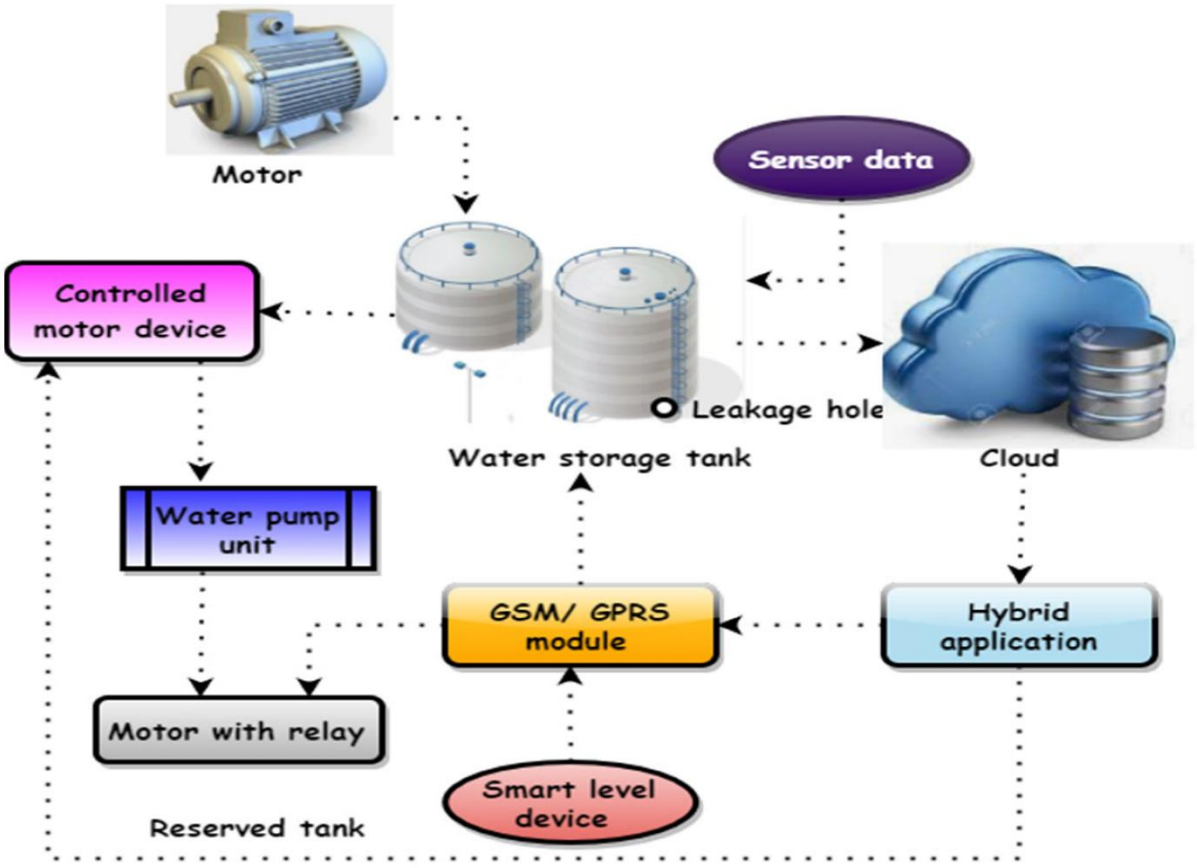


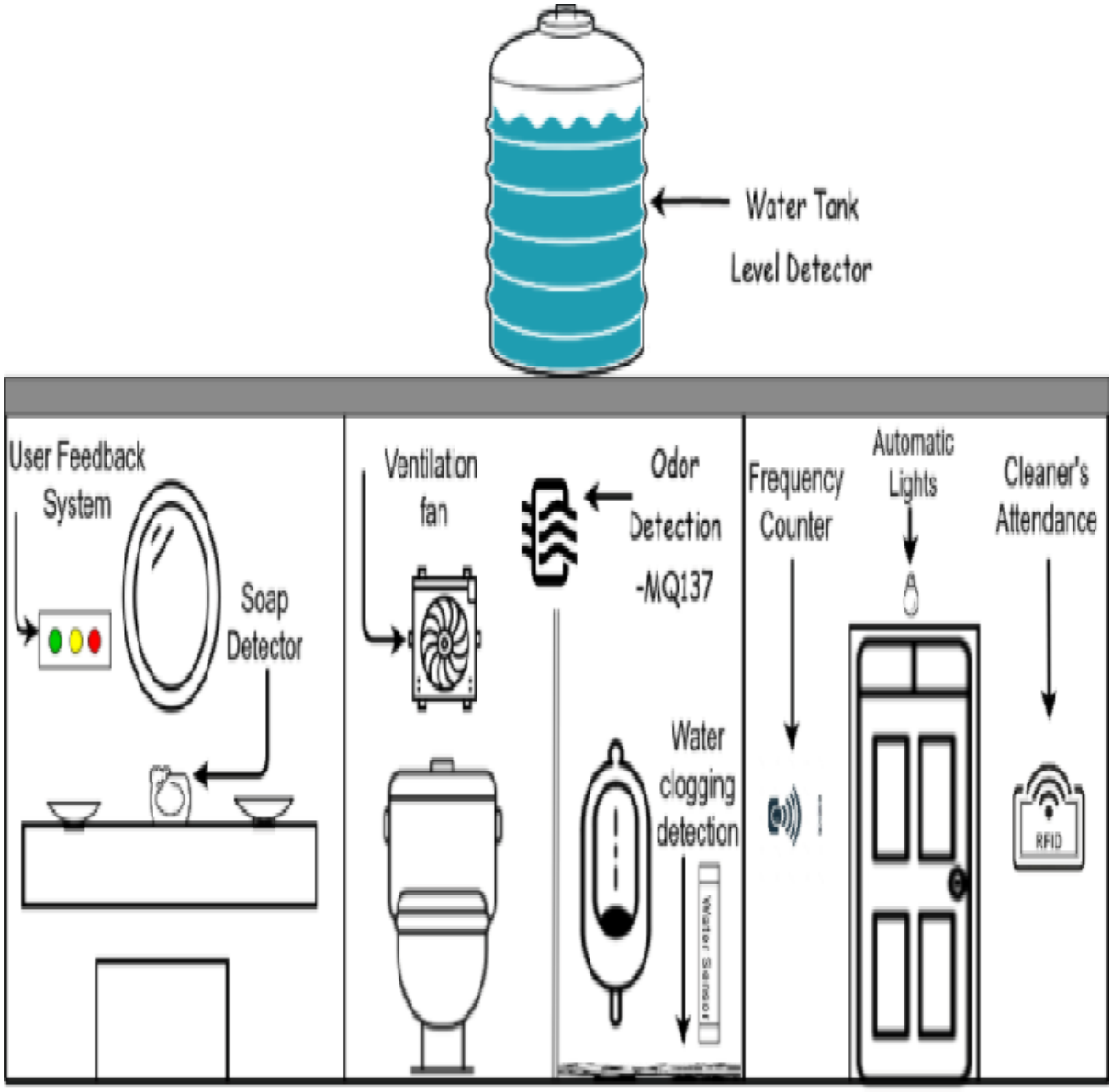
OUTPUT:



ENVIRONMENTAL SENSORS:


T-TENG
sensor array
Image
sensor


Motor
Sensor data
Controlled
motor device
Leakage hole
Water storage tank
Cloud
Water pump
unit
GSM/ GPRS
module
Hybrid
application
Motor with relay
Smart level
device
Reserved tank



SCHEMATIC DIAGRAM: