

SMART PUBLIC RESTROOM USING IOT

Team member

610821106002:ADIL AHMED.R

Phase 4 Project Submission

Project : 223930_Team_1_6108_Smart Public Restroom



Project Definition: The project aims to enhance public restroom management by installing IoT sensors to monitor occupancy and maintenance needs. The goal is to provide real-time data on restroom availability and cleanliness to the public through a platform or mobile app. This project includes defining objectives, designing the IoT sensor system, developing the restroom information platform, and integrating them using IoT technology and Python.

Phase 4 Development Part 2

Building The Project By Performing Different Activities Like Feature Engineering, Model Training, Evaluation For Smart Public Restroom

Designing a smart public restroom using IoT (Internet of Things) involves several stages, including feature engineering, model training, and evaluation. Here's a step-by-step guide on how to perform these activities:

Define the Problem:

Identify the specific problems and objectives you want to address with your smart public restroom, such as optimizing resource usage, improving cleanliness, or enhancing user experience.

Data Collection:

Set up IoT sensors and devices to collect relevant data. This may include occupancy sensors, temperature sensors, motion detectors, and even smart soap dispensers. Ensure data is securely and accurately transmitted to a central server.

Feature Engineering:

Process and transform raw IoT data into meaningful features for analysis. For a public restroom, features might include:

- Occupancy patterns (busy hours, peak usage times)
- Temperature and humidity levels
- Dispenser usage (soap, paper towels)
- Cleaning frequency and schedules
- User feedback data (e.g., via feedback kiosks)
- Consider using time series analysis to capture temporal patterns in the data.

Data Storage:

Set up a robust data storage system to store the collected data. You can use databases, data lakes, or cloud services for this purpose.

Data Preprocessing:

Clean and preprocess the data. Handle missing values, outliers, and ensure data consistency.

Model Selection:

Choose the appropriate machine learning or statistical models that fit the problem. For public restrooms, models like time series analysis, regression, or classification might be useful.

Model Training:

Split the data into training and validation sets.

Train your selected models on the training data, using the features you engineered.

Model Evaluation:

Assess the performance of your models using appropriate metrics. For a smart public restroom, metrics could include accuracy, F1 score, Mean Absolute Error (MAE), Mean Squared Error (MSE), etc.

Consider cross-validation to ensure robust model performance.

Optimization and Deployment:

Fine-tune your models to achieve the best results.

Once satisfied with the model's performance, deploy it to your smart public restroom system.

Real-time Monitoring:

Continuously monitor the IoT devices and the model's predictions in real-time. IoT platforms and dashboards can help with this.

Implement alert systems for anomalies and issues (e.g., low soap levels or restroom overuse).

Feedback Loop:

Collect and integrate user feedback to continuously improve the system. Adjust the model and system parameters based on feedback.

Maintenance and Updates:

Regularly maintain and update the IoT devices and the underlying software to ensure optimal performance.

Security:

Implement robust security measures to protect the IoT devices and data from unauthorized access or cyberattacks.

Compliance and Privacy:

Ensure that your smart public restroom system complies with relevant privacy and data protection regulations.

Scale and Expand:

If your smart restroom is successful, consider scaling the solution to more locations and expanding its features based on user needs and feedback.

Continue building the project by developing the restroom information platform and mobile app for smart public restroom using IoT

Restroom Information Platform:

Define Platform Features:

Identify the features you want to include in the restroom information platform. These might include real-time occupancy status, cleanliness ratings, wait time estimates, and user feedback.

Data Integration:

Connect the platform to the data collected from IoT devices in the restroom. Ensure that data is updated in real-time.

Database and API:

Set up a database to store the restroom data and create APIs to access this data. You can use technologies like RESTful APIs or GraphQL for this purpose.

User Authentication:

Implement a secure user authentication system. Users should be able to create accounts, log in, and personalize their experience.

Real-Time Updates:

Implement a real-time update mechanism, such as WebSockets, to provide users with instant data on restroom occupancy, cleanliness, and other relevant information.

User Interface (UI):

Develop a user-friendly web-based interface for the platform. Use modern design principles and responsive design to ensure usability on various devices.

Rating and Feedback System:

Allow users to rate the restroom's cleanliness and leave feedback. Implement a rating system and a comment section for each restroom location.

Search and Navigation:

Include a search feature that helps users find nearby smart restrooms. You can use geolocation to assist with this. Implement maps and navigation tools if necessary.

Notifications:

Enable push notifications to inform users about real-time restroom status changes or maintenance alerts.

Analytics and Reporting:

Add an analytics dashboard to monitor restroom usage patterns and user behavior. This data can be used for continuous improvement.

Mobile App:

Platform Selection

Decide whether to build separate apps for iOS and Android or create a cross-platform app using technologies like React Native or Flutter.

User Registration and Login

Allow users to register or log in to the mobile app to access the platform's features.

Dashboard:

Create a dashboard that displays the status of nearby smart restrooms, including occupancy, cleanliness ratings, and wait times.

Navigation:

Implement navigation features to guide users to their selected restroom locations. Utilize GPS and maps for this purpose.

Real-Time Updates:

Provide real-time updates on restroom availability, cleanliness, and user feedback.

Rating and Feedback

Enable users to rate restroom cleanliness and leave comments, just like on the web platform.

Push Notifications:

Send push notifications to keep users informed about restroom status changes and other important updates.

Profile Management:

Allow users to manage their profiles, preferences, and app settings.

Integration with Restroom Information Platform:

Ensure that the mobile app seamlessly integrates with the restroom information platform, sharing data and user preferences.

Testing and Quality Assurance

Thoroughly test the app on various devices and operating systems to ensure functionality, performance, and security.

Launch and Marketing:

Launch the mobile app on app stores (Google Play Store and Apple App Store) and promote it to your target audience.

User Support and Feedback:

Provide customer support channels for users who encounter issues and collect feedback to improve the app.

Updates and Maintenance:

Regularly update the app to add new features, improve performance, and fix bugs.

Use web development technologies (e.g., HTML, CSS, JavaScript) to create a platform that displays real-time restroom availability and cleanliness data for smart public restroom using IoT

HTML (index.html):

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Smart Restroom Dashboard</title>

  <link rel="stylesheet" href="style.css">

</head>

<body>

  <header>

    <h1>Smart Restroom Dashboard</h1>

  </header>

  <main>

    <section id="restroom-info">

      <h2>Restroom Information</h2>

      <div id="occupancy-status">

        <h3>Occupancy Status</h3>

        <p id="occupancy-data">Loading...</p>

      </div>

      <div id="cleanliness-status">

        <h3>Cleanliness Status</h3>

        <p id="cleanliness-data">Loading...</p>

      </div>

    </section>

  </main>

</body>

</html>
```



```
        </div>
    </section>
</main>
<script src="script.js"></script>
</body>
</html>
```

CSS (style.css):

```
body {
    font-family: Arial, sans-serif;
}
```

```
header {
    background-color: #0077b6;
    color: white;
    text-align: center;
    padding: 20px;
}
```

```
h1 {
    margin: 0;
}
```

```
main {
    padding: 20px;
}
```

```
h2 {  
    font-size: 24px;  
    margin-top: 0;  
}
```

```
#occupancy-status, #cleanliness-status {  
    border: 1px solid #ddd;  
    padding: 10px;  
    margin-bottom: 20px;  
    border-radius: 5px;  
}
```

```
#occupancy-data, #cleanliness-data {  
    font-size: 18px;  
}
```

JavaScript (script.js):

```
// Simulated real-time data (replace with actual IoT data)
```

```
const simulatedData = {  
    occupancy: "Occupied",  
    cleanliness: "Good",  
};
```

```
// Update the occupancy and cleanliness data
```

```
function updateData() {
```

```
const occupancyStatus = document.getElementById("occupancy-data");
const cleanlinessStatus = document.getElementById("cleanliness-data");

occupancyStatus.textContent = simulatedData.occupancy;
cleanlinessStatus.textContent = simulatedData.cleanliness;
}

// Update data every few seconds (simulated real-time updates)
setInterval(updateData, 5000);

// Initialize data on page load
updateData();
```

HTML defines the structure of the webpage, including placeholders for real-time occupancy and cleanliness data.

CSS provides basic styling to make the dashboard visually appealing.

JavaScript updates the data displayed on the dashboard every few seconds (simulated real-time updates). You should replace the simulated data with actual IoT data from your smart public restroom.

Design mobile apps for iOS and Android platforms that provide users with access to realtime restroom information for smart public restroom using IoT

Designing mobile apps for iOS and Android platforms that provide users with access to real-time restroom information for a smart public restroom using IoT involves several steps, including user interface design, app functionality, and integration with IoT data sources. Below is a high-level overview of the design process:

1. User Interface (UI) Design:

Define the app's visual style, color scheme, and overall look and feel.

Create wireframes and mockups for key app screens, including the dashboard, restroom details, user feedback, and settings.

Ensure a user-friendly and intuitive design with a focus on simplicity and ease of use.

2. Features and Functionality:

a. User Registration and Authentication

Allow users to create accounts and log in for a personalized experience.

Implement secure authentication methods.

b. Dashboard:

The main screen should display nearby smart restrooms.

Provide real-time information on occupancy status, cleanliness ratings, and wait times.

c. Restroom Details:

Users can select a specific restroom for more information.

Show real-time details, including occupancy, cleanliness, and user feedback.

d. User Feedback

Allow users to rate restroom cleanliness and leave comments.

Provide a way for users to report issues or leave suggestions.

e. Navigation:

Include GPS and mapping features for users to navigate to selected restrooms.

f. Notifications

Send push notifications to inform users about restroom status changes and maintenance alerts.

g. User Profiles

Enable users to manage their profiles, view their history, and customize their preferences.

3. Data Integration:

Connect the app to the IoT data sources and the restroom information platform. Ensure real-time data updates.

4. Platform-Specific Development:

a. iOS (Swift):

Develop the app for iOS devices, following Apple's guidelines and best practices.

Optimize the user experience for iPhones and iPads.

b. Android (Java/Kotlin):

Develop the app for Android devices, following Google's guidelines and best practices.

Optimize the user experience for various Android smartphones and tablets.

5. User Experience (UX) Testing:

Test the app on different iOS and Android devices to ensure it functions correctly and looks great on each platform.

Pay attention to responsiveness, performance, and usability.

6. Security and Privacy:

Implement strong security measures to protect user data and privacy.

Comply with relevant data protection regulations, such as GDPR or CCPA.

7. Quality Assurance:

Conduct thorough testing to identify and fix any bugs, crashes, or performance issues.

Ensure that the app functions seamlessly on different device models and operating system versions.

8. App Store Deployment:

Submit the app to the Apple App Store for iOS and Google Play Store for Android.

Prepare app descriptions, screenshots, and other assets.

9. Marketing and User Onboarding:

Plan a marketing strategy to promote the app to your target audience.

Create user onboarding materials to help users understand how to use the app effectively.

10. User Support and Feedback:

Provide customer support channels for users who encounter issues.

Collect user feedback to make continuous improvements.

11. Updates and Maintenance:

Regularly update the app to add new features, improve performance, and address user feedback.