**Phase-5**

**Smart public restroom**

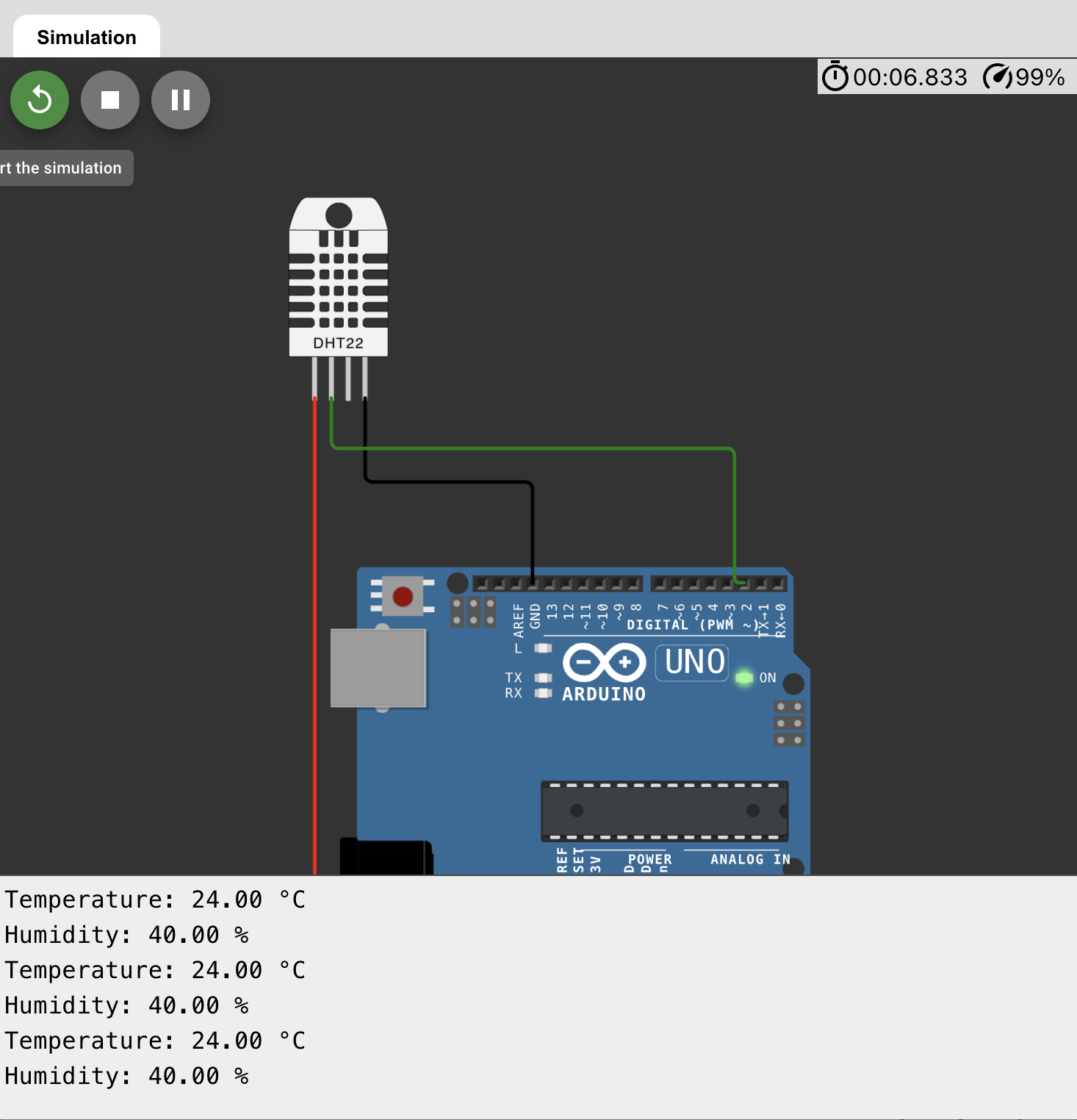
**Introduction**

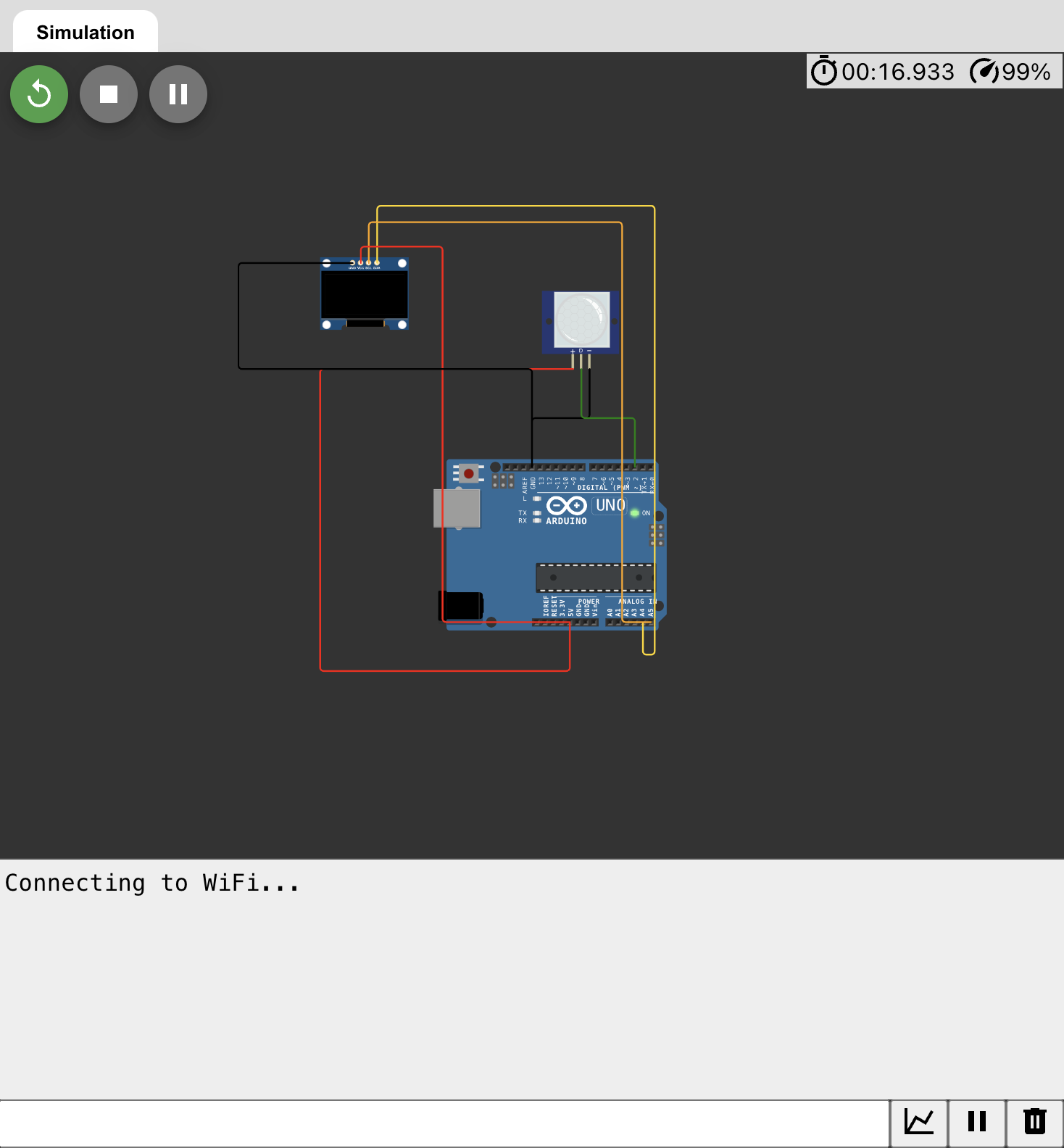
Welcome to the future of restroom experiences – the Smart Public Restroom. This cutting-edge facility combines IoT technology with a user-friendly app to redefine hygiene and accessibility. Say goodbye to restroom hassles – sensors ensure availability, cleanliness, and comfort. The app lets you find the nearest Smart Restroom, check real-time availability, and customize your preferences. Join us in creating a smarter, cleaner, and more enjoyable public restroom experience. Step into the future where innovation meets necessity, making every visit a seamless and connected encounter.

**Sensors**

1. **Occupancy Sensors**: These sensors are used to detect the presence of individuals within the restroom. They help in monitoring restroom usage in real-time, which is valuable for managing cleaning schedules and assessing occupancy levels. Occupancy data can also be used to display real-time availability and estimated wait times for users.
2. **Water Flow Sensors**: Water flow sensors are installed in water supply lines to monitor water usage. They help in promoting water efficiency by detecting leaks, monitoring water flow rates, and ensuring that water-saving fixtures are functioning correctly. This data can also be used to optimize water consumption.
3. **Air Quality Sensors**: Air quality sensors measure parameters such as humidity, temperature, and air quality (e.g., CO2 levels). These sensors help maintain a comfortable and healthy restroom environment by regulating temperature and humidity. Additionally, they ensure adequate ventilation and air quality for user comfort.
4. **Touchless Sensors**: Infrared or capacitive touchless sensors are used in faucets, soap dispensers, and flush systems. These sensors enable touchless operation, minimizing contact with surfaces and improving hygiene. They are crucial for reducing the spread of germs and ensuring user safety.
5. **Occupancy Indicator Sensors**: These sensors are often used outside the restroom to display occupancy status to users. They help users quickly determine if the restroom is available or occupied, reducing wait times and improving user experience.
6. **Security Cameras**: While not traditional sensors, security cameras are essential for maintaining security and ensuring user safety within the restroom. They deter vandalism and provide a record of activities inside the restroom, enhancing overall security.
7. **Light and Motion Sensors**: Light and motion sensors are used for energy efficiency. They ensure that lights are turned on only when needed, reducing energy consumption. Additionally, motion sensors can be used to trigger exhaust fans, ensuring proper ventilation.
8. **Rainwater Harvesting Sensors**: In restrooms equipped with rainwater harvesting systems for toilet flushing, sensors monitor the availability and level of collected rainwater. When rainwater is available, the system can automatically switch to using rainwater, reducing reliance on municipal water supply and promoting sustainability.
9. **Sound Sensors**: Sound sensors can be used to monitor sound levels within the restroom. They help in providing a more pleasant and private atmosphere by controlling ambient sounds, such as music or white noise.
10. **Temperature Sensors**: Temperature sensors help maintain a comfortable restroom environment. They ensure that heating and cooling systems are regulated effectively, ensuring user comfort.

**Simulator**

****

****

#include <Wire.h>

#include <Adafruit\_SSD1306.h>

#include <WiFi.h>

#include <ThingSpeak.h> // Include the ThingSpeak library

#define SCREEN\_WIDTH 128

#define SCREEN\_HEIGHT 64

#define OLED\_ADDR 0x3C

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &**Wire**, OLED\_ADDR);

const int pirPin = 2;

int pirState = LOW;

const char\* ssid = "Wokwi-GUEST";

const char\* password = "";

const unsigned long channelID = 2310712;

const char\* writeAPIKey = "9PMHPSNFQEMZXR";

void setup() {

// Initialize Wi-Fi

**Serial**.begin(115200);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(10000);

**Serial**.println("Connecting to WiFi...");

}

**Serial**.println("Connected to WiFi");

pinMode(pirPin, INPUT);

if (!display.begin(SSD1306\_SWITCHCAPVCC, OLED\_ADDR)) {

for (;;);

}

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

}

void loop() {

int pirValue = digitalRead(pirPin);

if (pirValue == HIGH) {

if (pirState == LOW) {

// Clear the display

display.clearDisplay();

display.setCursor(0, 0);

display.println("Occupied");

display.display();

// Send data to ThingSpeak

sendToThingSpeak(1); // Send 1 to indicate "Occupied"

delay(500);

pirState = HIGH;

}

} else {

display.clearDisplay();

display.setCursor(0, 0);

display.println("Free");

display.display();

// Send data to ThingSpeak

sendToThingSpeak(0); // Send 0 to indicate "Free"

delay(500);

pirState = LOW;

}

}

void sendToThingSpeak(int occupancyStatus) {

// ThingSpeak.begin(client); // You don't need to initialize the client

ThingSpeak.setField(1, occupancyStatus); // Set the value for field 1 (Occupancy Status)

int httpStatus = ThingSpeak.writeFields(channelID, writeAPIKey);

if (httpStatus == 200) {

**Serial**.println("Data sent to ThingSpeak successfully.");

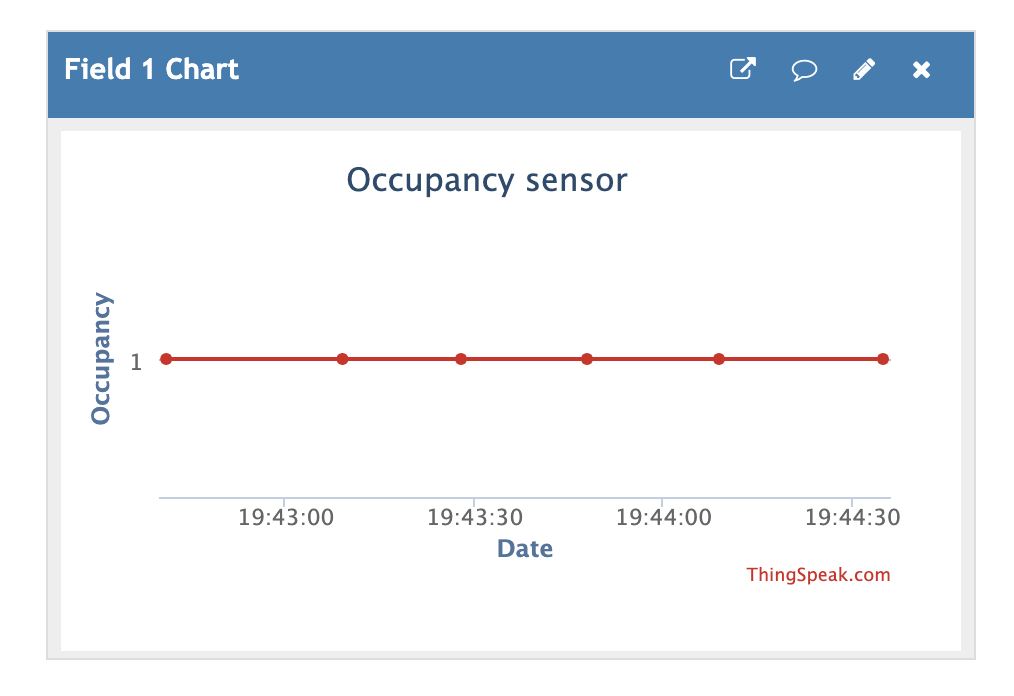
} else {

**Serial**.println("Data send to ThingSpeak failed. HTTP error code: " + String(httpStatus));

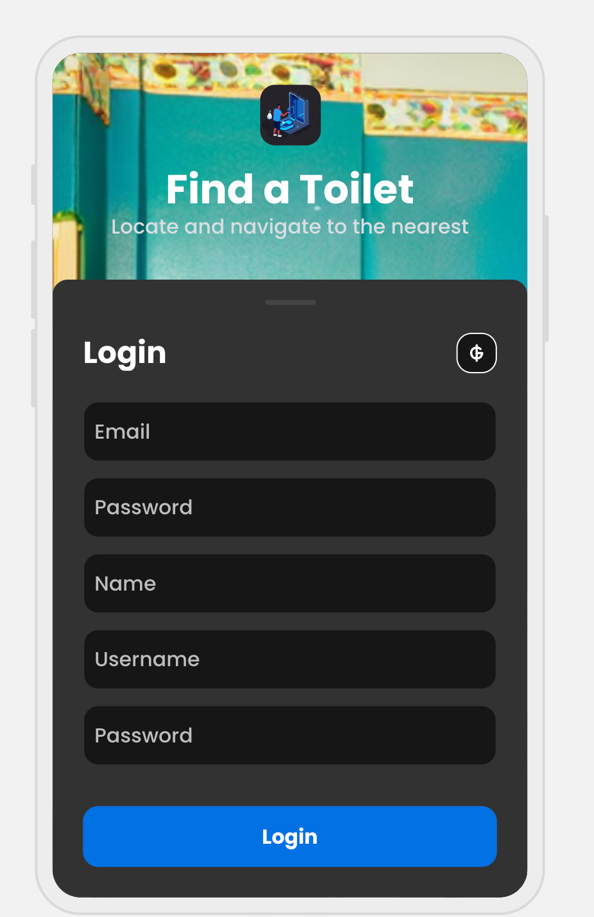
}

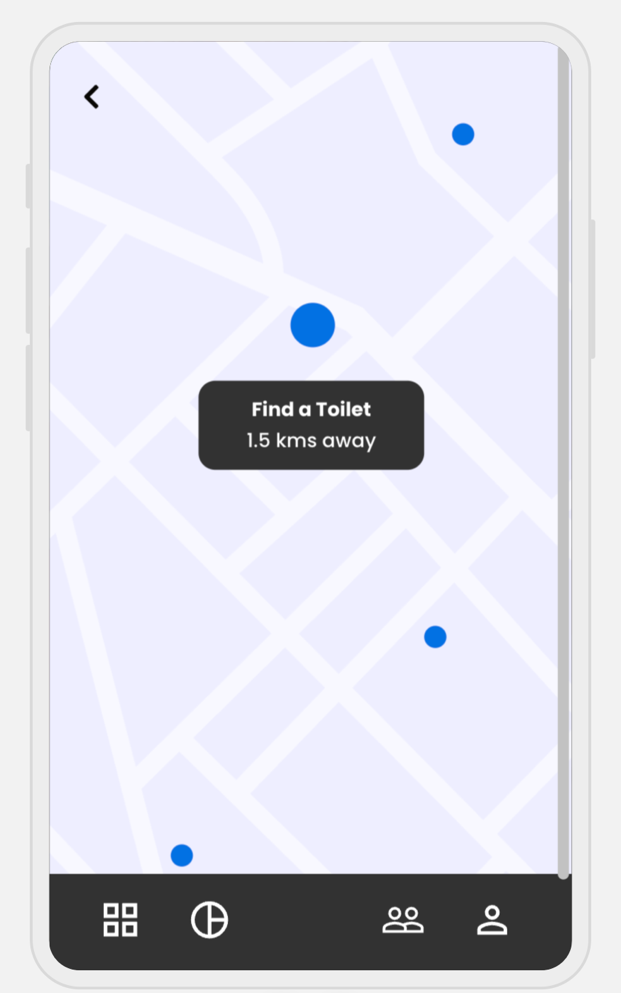
}

**Cloud:**

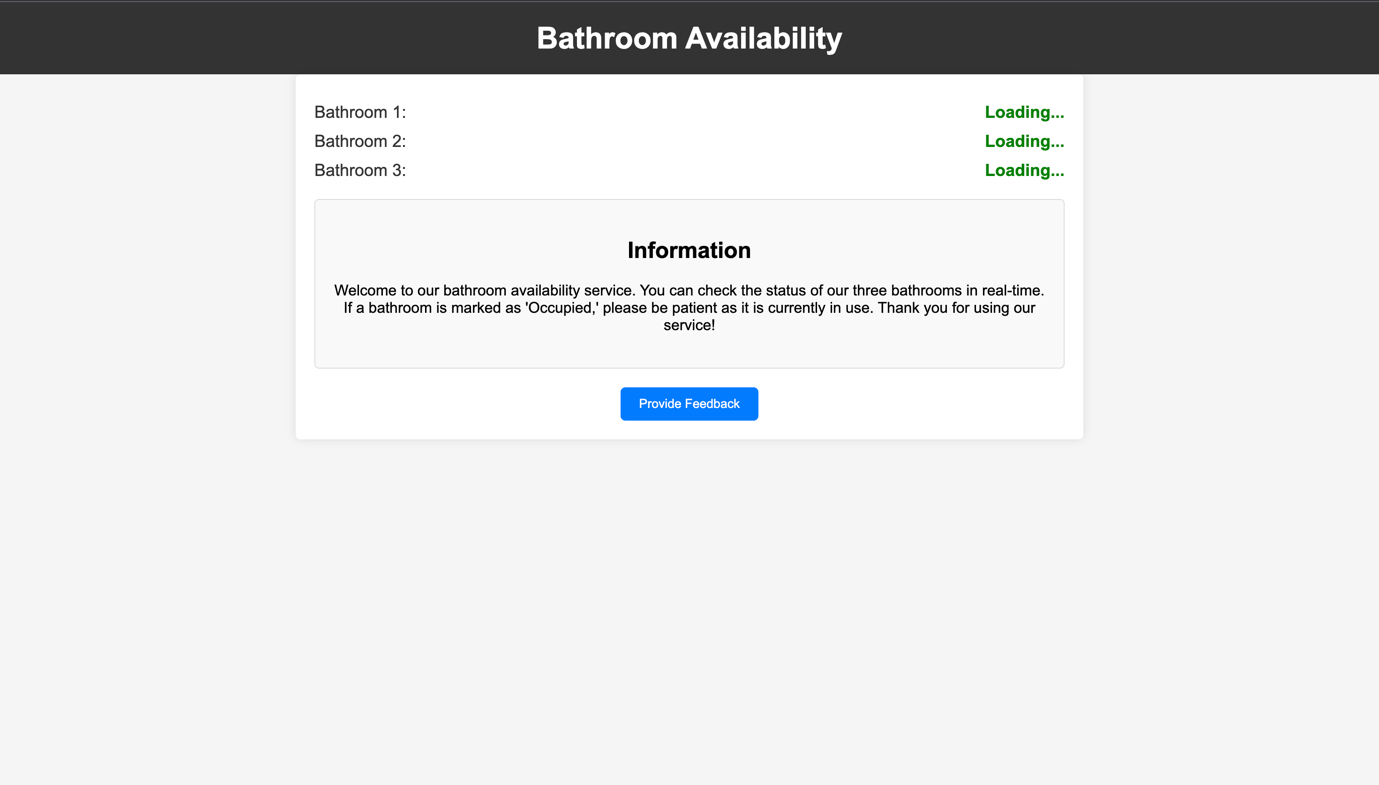
****

**APPLICATION**

****

****

**WEBSITE**

****