VELAMMAL
INSTITUTE OF TECHNOLOGY

Approved by AICTE - New Delhi
Affiliated to Anna University - Chennai
Accredited by NBA & NAAC

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROJECT NAME: SMART PUBLIC RESTROOM
TEAM NAME: proj_224780_Team_4
TEAM MEMBERS:

ABINAYA.P(113321104001)
ABHIRAMI.T(113321104002)
ALAGANENI SUMI(113321104003)
GEETHA.S.V

Innovation:

Smart public restrooms can greatly enhance user experience and hygiene. Here are some innovative ideas

- **Automated Cleaning:** Incorporate sensors and robotics to automatically clean and disinfect restroom fixtures and floors, reducing the need for human intervention.
- **Touchless Fixtures:** Use touchless faucets, soap dispensers, and flush mechanisms to minimize germ transmission.
- **Energy Efficiency:** Utilize smart lighting and HVAC systems that adjust based on occupancy to save energy

PROJECT OBJECTIVES:

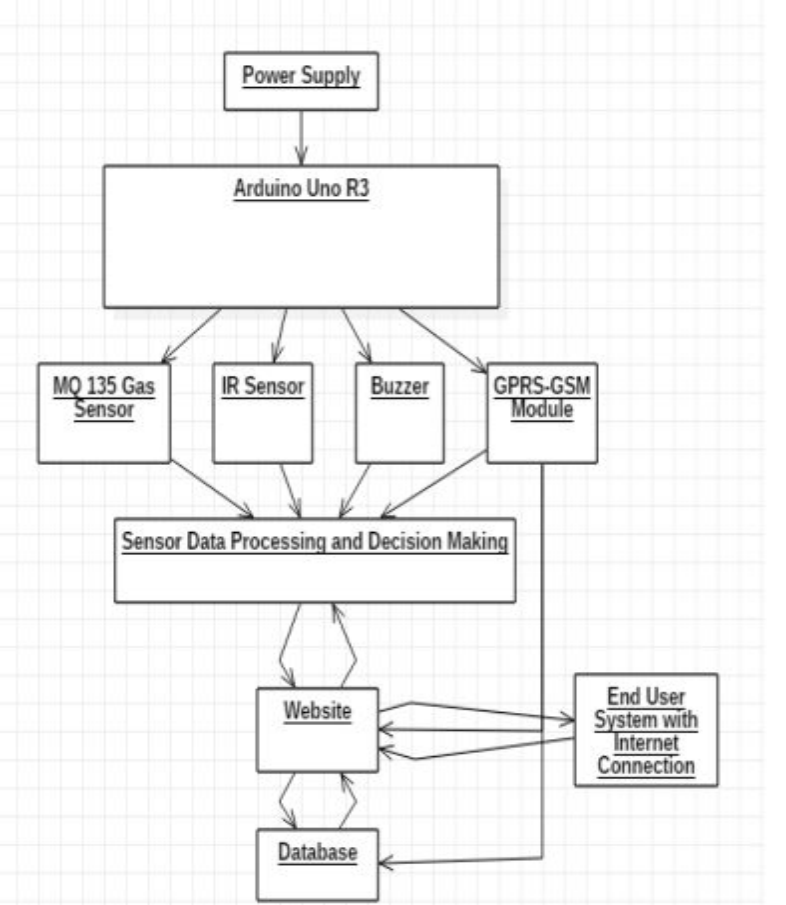
- Building a smart public restroom using IoT involves various components and technologies.
- Below, I'll provide you with a high level Python code example for a simplified smart public restroom system.
- Keep in mind that this is a basic example, and a real-world implementation would require more robust hardware, sensors, and a backend system for managing data.
- Now a days the smart Restroom is essential More in Hitech city and more are comfortable with this.

Project Requirements:

The Components that are required are:

- 1.Raspberry Pi (or any other IoT device)
- 2.Sensors (e.g., occupancy sensor, door sensor, ultrasonic sensor)
- 3.IoT Platform (e.g., ThingSpeak for data visualization)
- 4.Actuators (e.g., LED lights, fans)
- 5.Relay module for controlling actuators
- 6.Internet connectivity

RASPBERRY PI INTEGRATION:



Our proposed system is a smart monitoring system designed to monitor the hygiene of public toilets. Unhygienic toilets can be detected by different parameters such as water levels, and various gases evolved, humidity, temperature etc. Ammonia gas is the most dominant gas that can be sensed in an unhygienic toilet. We will be using the MQ-135 gas sensor the determine the amount of ammonia present in the room

CODE IMPLEMENTATION:

Hardware Components:

- ❑ Raspberry Pi (or similar single-board computer)
- ❑ Water Flow Sensor
- ❑ Solenoid Valve (for controlling water flow)
- ❑ Wi-Fi Module (for internet connectivity)
- ❑ Power Supply

Software Components:

- ❑ Python (for programming)
- ❑ MQTT (for communication)
- ❑ Cloud server (for data storage and remote control)

PYTHON CODE:

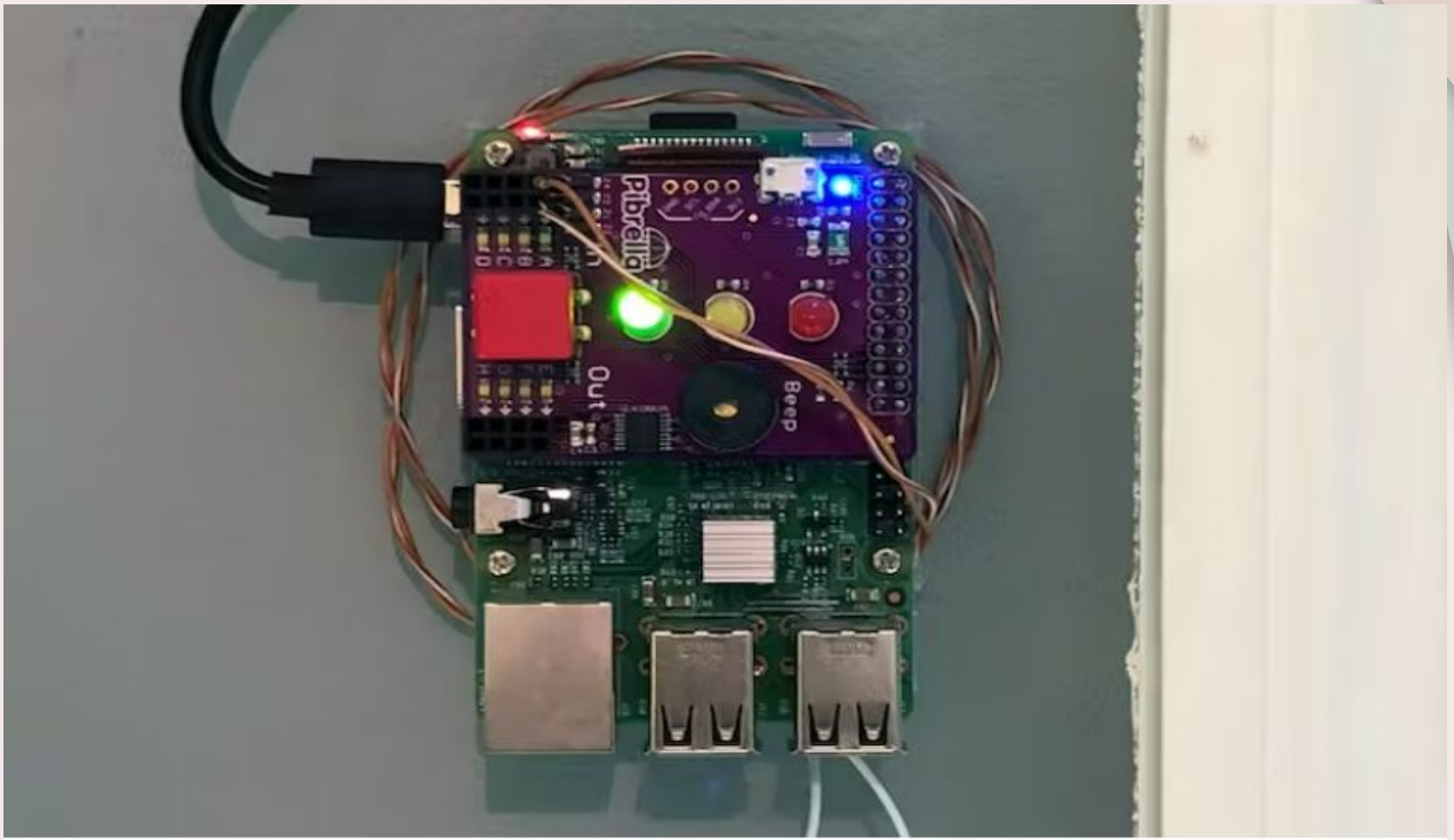
```
import RPi.GPIO as GPIO import time import requests #
Set up GPIO pins OCCUPANCY_SENSOR_PIN = 18
DOOR_SENSOR_PIN = 23
ULTRASONIC_TRIGGER_PIN = 24
ULTRASONIC_ECHO_PIN = 25
GPIO.setmode(GPIO.BCM)
GPIO.setup(OCCUPANCY_SENSOR_PIN, GPIO.IN)
GPIO.setup(DOOR_SENSOR_PIN, GPIO.IN)
GPIO.setup(ULTRASONIC_TRIGGER_PIN,
GPIO.OUT) GPIO.setup(ULTRASONIC_ECHO_PIN,
GPIO.IN)
```

```
# Function to read ultrasonic sensor def
read_ultrasonic_sensor():
GPIO.output(ULTRASONIC_TRIGGER_PIN, True)
time.sleep(0.00001)
GPIO.output(ULTRASONIC_TRIGGER_PIN, False)
pulse_start_time = time.time() pulse_end_time =
time.time() while
GPIO.input(ULTRASONIC_ECHO_PIN) == 0:
pulse_start_time = time.time() while
GPIO.input(ULTRASONIC_ECHO_PIN) == 1:
pulse_end_time = time.time() pulse_duration =
pulse_end_time - pulse_start_time distance =
(pulse_duration * 34300) / 2 # Speed of sound =
34300 cm/s return distance
```

```
pulse_duration = pulse_end_time - pulse_start_time
distance = (pulse_duration * 34300) / 2 # Speed of
sound = 34300 cm/s return distance # Function to
send data to IoT platform def
send_data_to_iot(occupancy, door_status, distance):
url = "https://api.thingspeak.com/update" params = {
"api_key": "YOUR_API_KEY", "field1": occupancy,
"field2": door_status, "field3": distance } response =
requests.get(url, params=params) print("Data sent to
IoT platform:", response.text) try: while True:
occupancy=
GPIO.input(OCCUPANCY_SENSOR_PIN)
door_status = GPIO.input(DOOR_SENSOR_PIN)
distance = read_ultrasonic_sensor()
```

Control actuators based on sensor data # For example, turn on lights and fans when occupancy is detected if occupancy == 1: # Activate actuators GPIO.output(LED_PIN, GPIO.HIGH) GPIO.output(FAN_PIN, GPIO.HIGH) else: # Deactivate actuators GPIO.output(LED_PIN, GPIO.LOW) GPIO.output(FAN_PIN, GPIO.LOW) # Send data to the IoT platform send_data_to_iot(occupancy, door_status, distance) time.sleep(5) # Update data every 5 seconds except KeyboardInterrupt: GPIO.cleanup()

IMPLEMENTATION AND SIMULATION:



THANK YOU