**SMART PUBLIC RESTROOM**

**ABSTRACT:**

**Public toilets experience the perpetual problem of lack of hygiene. Appointing an all-time janitor is not possible at each and every toilet. Automation can reduce the janitor's burden by looking after the maintenance of the toilet and sending timely updates. We've tried to tackle this issue by developing an IoT-based smart system. We've used sensors to measure water level in tanks, water usage and to detect the presence of a person in the toilet. With the data from sensors, our system predicts the cleanliness of toilets. The sensors are interfaced with Raspberry Pi, which processes the sensor data and uploads it to the cloud. The necessary action is initiated by the Pi. The data collected by the sensors is fed to ThingSpeak, which generates graphs. With its ready-made machine learning tools, ThingSpeak identifies patterns in data and provides an analysis, which can uncover some useful information.**

**FLOW CHART:**



**Data sets:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S.No | Zone | Target(state) | GHMC TARGET | Work in progress | Completed |
| 1 | LB Nagar | 500 | 1200 | 542 | 458 |
| 2 | Kukatpalli | 500 | 1200 | 870 | 130 |
| 3 | Chaminar | 500 | 1200 | 644 | 388 |
| 4 | Secunderbad | 500 | 1200 | 860 | 142 |
| 5 | Serilingampalli | 500 | 1200 | 658 | 204 |
| 6 | Khairatabad | 500 | 1200 | 697 | 214 |
| 7 | Qutubullapur | 500 | 1200 | 558 | 345 |
| 8 | Rajendranagar | 500 | 1200 | 479 | 678 |
| 9 | Patancheru | 500 | 1200 | 276 | 195 |
| 10 | Uppal kalan | 500 | 1200 | 119 | 298 |
|  | Total | 5000 | 12000 | 5703 | 3052 |

A smart restroom, also known as an intelligent restroom, is a technologically advanced facility that integrates various sensors, devices, and IoT technologies to enhance the overall restroom experience. These smart solutions are designed to address the challenges faced by traditional restrooms, such as cleanliness, efficiency, and user satisfaction.

By leveraging IoT technology, smart restrooms can collect and analyze real-time data on various aspects of restroom operations, including occupancy, cleanliness, supply levels, and maintenance requirements. This data-driven approach allows facility managers to make informed decisions, optimize resource allocation, and improve overall restroom management.

Smart restrooms come equipped with a wide range of features that enhance hygiene, convenience, and user experience. Let’s explore some of the key features that make these restrooms truly smart

One of the fundamental features of a smart restroom is the presence of occupancy sensors. These sensors use advanced technologies like passive infrared (PIR) or time-of-flight (ToF) to accurately detect the number of users entering and exiting the restroom. This data helps facility managers monitor restroom traffic, optimize cleaning schedules, and ensure that there are enough resources available to meet user demands.

Under process

200

200

168

198

338

289

726

420

534

842

3915

**Program(python):**

#define BLYNK\_TEMPLATE\_ID "TMPLgCeV0y1b"

#define BLYNK\_DEVICE\_NAME "Home"

#define BLYNK\_AUTH\_TOKEN "93h-1b23ewIQooDTdB2y2COGacfYkbdO"

#include <LiquidCrystal\_I2C.h>

LiquidCrystal\_I2C lcd(0x27, 20, 4);

#define BLYNK\_PRINT **Serial**

#include <WiFi.h>

#include <WiFiClient.h>

#include <BlynkSimpleEsp32.h>

#include "DHTesp.h"

BlynkTimer timer;

char auth[] = BLYNK\_AUTH\_TOKEN;

char ssid[] = "Wokwi-GUEST";

char pass[] = "";

int val = 0, va1,va2,va3,va4,va5,ge, t =15 ;

float tmp,hum = 0;

int ledPin = 33;

int inputPin = 27;

int pirState,k;

int v = 0;

//temp symbol

byte t1[8]={B00000, B00001, B00010, B00100, B00100, B00100, B00100, B00111,};

byte t2[8]={B00111, B00111, B00111, B01111,B11111, B11111, B01111, B00011,};

byte t3[8]={B00000, B10000, B01011, B00100, B00111, B00100, B00111, B11100,};

byte t4[8]={B11111, B11100, B11100, B11110,B11111, B11111, B11110, B11000,};

//humidity symbol

byte hum1[8]={B00000, B00001, B00011, B00011,B00111, B01111, B01111, B11111,};

byte hum2[8]={B11111, B11111, B11111, B01111,B00011, B00000, B00000, B00000,};

byte hum3[8]={B00000, B10000, B11000, B11000, B11100, B11110, B11110, B11111,};

byte hum4[8]={B11111, B11111, B11111, B11110, B11100, B00000, B00000, B00000,};

//Home Symbol

byte house1[8]={B00000, B00001, B00011, B00011, B00111, B01111, B01111, B11111,};

byte house2[8]={B11111, B11111, B11100, B11100, B11100, B11100, B11100, B11100,};

byte house3[8]={B00000, B10010, B11010, B11010, B11110, B11110, B11110, B11111,};

byte house4[8]={B11111, B11111, B11111, B10001, B10001, B10001, B11111, B11111,};

byte d[8] = { 0b00011,0b00011,0b00000,0b00000,0b00000,0b00000,0b00000,0b00000 };

byte Lck[] = { B01110, B10001, B10001, B11111, B11011, B11011, B11111, B00000 };

DHTesp temps;

BLYNK\_WRITE(V0){

 va1 = param.asInt();

 digitalWrite(5, va1);

}

BLYNK\_WRITE(V1){

 va2 = param.asInt();

 digitalWrite(18, va2);

}

BLYNK\_WRITE(V2){

 va3 = param.asInt();

 digitalWrite(19, va3);

}

BLYNK\_WRITE(V3){

 va4 = param.asInt();

 digitalWrite(4, va4);

}

BLYNK\_WRITE(V4){

 va5 = param.asInt();

 digitalWrite(2, va5);

}

BLYNK\_WRITE(V7) {

  pirState = param.asInt();

  if(pirState == 0){

    digitalWrite(ledPin, LOW);

    k = 1;

    ge = 0;

  }

  else {

    digitalWrite(ledPin, HIGH);

    k= 0;

    ge = 1;

  }

}

void myTimer()

{

  Blynk.virtualWrite(V5,tmp);

  Blynk.virtualWrite(V6,hum);

}

void setup()

{

**Serial**.begin(115200);

 Blynk.begin(auth, ssid, pass);

pinMode(5, OUTPUT);

pinMode(18, OUTPUT);

pinMode(19, OUTPUT);

pinMode(4, OUTPUT);

pinMode(23,INPUT);

pinMode(2,OUTPUT);

temps.setup(t, DHTesp::DHT22);

pinMode(ledPin, OUTPUT);

pinMode(inputPin, INPUT\_PULLUP);

lcd.init();

lcd.backlight();

digitalWrite(5, LOW);

digitalWrite(18, LOW);

digitalWrite(19, LOW);

digitalWrite(21, LOW);

lcd.setCursor(0,0);

lcd.print("CircuitDesignContest");

lcd.setCursor(8,1);

lcd.print("2022");

lcd.setCursor(0,2);

lcd.print("--------------------");

lcd.setCursor(9,3);

lcd.print("- eDiYLaBs");

delay(3000);

lcd.clear();

lcd.createChar(6, Lck);

lcd.createChar(1,house1);

lcd.createChar(2,house2);

lcd.createChar(3,house3);

lcd.createChar(4,house4);

lcd.setCursor(1,2);

lcd.write(1);

lcd.setCursor(1,3);

lcd.write(2);

lcd.setCursor(2,2);

lcd.write(3);

lcd.setCursor(2,3);

lcd.write(4);

lcd.setCursor(17,2);

lcd.write(1);

lcd.setCursor(17,3);

lcd.write(2);

lcd.setCursor(18,2);

lcd.write(3);

lcd.setCursor(18,3);

lcd.write(4);

lcd.setCursor(19,0);

lcd.write(6);

lcd.setCursor(9,0);

lcd.print("connected-");

lcd.setCursor(2,1);

lcd.print("HOME AUTOMATION");

lcd.setCursor(6,2);

lcd.print("USING IOT");

delay(3000);

Blynk.virtualWrite(V7, pirState);

timer.setInterval(1000L, myTimer);

}

**LIBRARY FILE:**

# Wokwi Library List

# See <https://docs.wokwi.com/guides/libraries>

Blynk

LiquidCrystal I2C

DHT sensor library for ESPx

**Output:**



**PROGRAM(python):**

#include<ESP32Servo.h>

#define TRIGGERPIN 32

#define ECHOPIN 35

#define RED\_LED 33

#define GREEN\_LED 25

Servo servo\_1;

long duration;

int pos, distance, i=0;

void setup()

{

servo\_1.attach(18);

Serial.begin(115200);

pinMode(TRIGGERPIN, OUTPUT);

pinMode(ECHOPIN, INPUT);

pinMode(RED\_LED, OUTPUT);

pinMode(GREEN\_LED, OUTPUT);

Serial.println(" ");

Serial.println("Sensing the Height");

digitalWrite(RED\_LED, HIGH);

digitalWrite(GREEN\_LED, LOW);

pos = 0;

servo\_1.write(pos);

}

void loop()

{

digitalWrite(TRIGGERPIN, LOW);

delayMicroseconds(3);

digitalWrite(TRIGGERPIN, HIGH);

delayMicroseconds(12); // it may be 10 us

digitalWrite(TRIGGERPIN, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(ECHOPIN, HIGH);

// Calculating the distance

distance = (duration/2) / 29.1;

// for Adult

if (distance >= 100 && distance <= 150)

{

i = 1;

if (pos != 180)

{

servo\_1.write(180);

pos = 180;

i = 1;

}

}

// for Child

else if (distance >= 200 && distance <= 250)

{

i = 1;

if (pos != 0)

{

servo\_1.write(0);

pos = 0;

i = 1;

}

}

else if (distance > 300 && i == 1)

{

digitalWrite(RED\_LED, LOW);

digitalWrite(GREEN\_LED, HIGH);

delay(5000);

digitalWrite(RED\_LED, HIGH);

digitalWrite(GREEN\_LED, LOW);

i = 0;

}

delay (500);

Serial.println(" ");

Serial.print("Free Level : ");

Serial.print(distance);

Serial.print(" ");

Serial.print("Position : ");

Serial.print(pos);

delay (500);

}

**LIBRARY FILES:**

# Wokwi Library List

# See <https://docs.wokwi.com/guides/libraries>

# Automatically added based on includes:

ESP32Servo

**OUTPUT:**

