SMART HOME AUTOMATION

Designed and Developed by

1. M. SAI RAM	**2211CS010379**
2. P.NIKHITHA	**2211CS010438**
3. M. VINAY	**2211CS010367**
4. G.KISHORE	**2211CS010209**

Guided by

MRS.PREETHI REDDY.V

Department of Computer Science & Engineering

MALLA REDDY UNIVERSITY, HYDERABAD

2024-2025

MALLA REDDY UNIVERSITY
(Telangana State Private Universities Act No.13 of 2020 and G.O.Ms.No.14, Higher Education (UE) Department)

CERTIFICATE

This is to certify that this is the Application development lab record entitled "**SMART HOME AUTOMATION**", submitted by **M. SAI RAM (2211CS010379), P.NIKHITHA (2211CS010438), M.VINAY (2211CS010367), G. KISHORE (2211CS010209)** B.Tech III year I semester, Department of CSE during the year 2024-25. The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma.

Internal Guide | **HOD-CSE**

MRS.PREETHI REDDY.V | **Dr.SHAIK MEERAVALI**

External Examiner

MALLA REDDY UNIVERSITY
(Telangana State Private Universities Act No.13 of 2020 and G.O.Ms.No.14, Higher Education (UE) Department)

DECLARATION

I declare that this project report titled **SMART HOME AUTOMATION** submitted in partial fulfillment of the degree of B. Tech in CSE is a record of original work carried out by me under the supervision of **MRS.PREETHI REDDY.V** and has not formed the basis for the award of any other degree or diploma, in this or any other Institution or University. In keeping with the ethical practice in reporting scientific information, due acknowledgements have been made wherever the findings of others have been cited.

Signature

M. SAI RAM (2211CS010379)
P.NIKHITHA (2211CS010438)
M. VINAY (2211CS010367)
G.KISHORE (2211CS010209)

ACKNOWLEDGEMENT

We would like to express our gratitude to all those who extended their support and suggestions to come up with this software. Special Thanks to our mentor **MRS.PREETHI REDDY.** whose help and stimulating suggestions and encouragement helped us all time in the due course project development.

We sincerely thank our Head of the Department **Dr. SHAIK MEERAVALI** for his constant support and motivation all the time. A special acknowledgement goes to a friend who enthused us from the backstage. Last but not the least our sincere appreciation goes to our family who has been tolerant, understanding our moods and extending timely support.

M.SAI RAM (2211CS010379)

P.NIKHITHA (2211CS010438)

M. VINAY (2211CS010367)

G.KISHORE (2211CS010209)

ABSTRACT

This project presents an advanced Smart Home Automation system powered by the Raspberry Pi Pico W, integrating Bluetooth and IoT technologies for enhanced home management. The primary goal is to provide users with convenient control over household devices such as lights and fans through a dedicated mobile application, promoting a flexible and user-centric approach to automation.

The system features multiple sensors, including a Light Dependent Resistor (LDR) for ambient light detection and a gas sensor for safety monitoring. These sensors continuously collect data to maintain optimal environmental conditions. For example, the LDR adjusts lighting automatically based on surrounding light levels, while the gas sensor triggers safety measures if gas concentrations exceed acceptable limits.User interaction is facilitated via a Liquid Crystal Display (LCD) that provides real-time feedback on sensor statuses, operational modes, and alerts. This visual interface enhances the user experience by making system states easily understandable at a glance. Moreover, manual control through Bluetooth commands allows users to override automatic functions as needed.Data from the sensors is periodically uploaded to a remote server via WiFi for logging and analysis, offering insights into usage patterns and environmental conditions over time. This functionality aids in monitoring and supports informed decisions regarding energy consumption and safety.The system architecture is designed for scalability and flexibility. Future enhancements may include additional sensors (e.g., temperature, humidity), advanced machine learning algorithms for predictive analytics, improved security protocols, and user authentication mechanisms. Such upgrades would enhance the system's robustness and appeal, making it a comprehensive solution for modern smart homes.

In summary, this Smart Home Automation system simplifies everyday tasks while prioritizing safety and energy efficiency, showcasing the potential of IoT technologies to transform residential living spaces into intelligent environments.

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER – 1

INTRODUCTION

1.1 Introduction

The rapid advancement of technology has transformed the way we interact with our living spaces, paving the way for innovative solutions in home automation. This project leverages the capabilities of the Raspberry Pi Pico W to create a Smart Home Automation system that enhances convenience, safety, and energy efficiency for users.

The primary aim of this project is to provide an intuitive platform that allows users to control household devices such as lights and fans through a mobile application using Bluetooth connectivity. By integrating various sensors—specifically a Light Dependent Resistor (LDR) for light detection and a gas sensor for safety monitoring—the system can automatically adjust to environmental conditions and respond to potential hazards, ensuring a safer and more comfortable living space.

The use of a Liquid Crystal Display (LCD) enhances user interaction by providing real-time updates on sensor readings and device statuses, making the system user-friendly and accessible. The combination of manual and automated controls empowers users to customize their home environment according to their preferences and needs.

Additionally, the system's ability to upload sensor data to a remote server for logging and analysis offers valuable insights into usage patterns and environmental conditions over time. This feature not only aids in proactive monitoring but also assists users in making informed decisions regarding energy consumption and safety practices.

As smart home technologies continue to evolve, this project serves as a foundational step toward developing a more interconnected and intelligent home environment. The system's design is modular, allowing for future enhancements such as the integration of additional sensors and advanced features, thereby ensuring its relevance in an ever-changing technological landscape.

Overall, this Smart Home Automation system exemplifies how IoT and microcontroller technologies can work together to create a more efficient and user-friendly home experience, ultimately improving the quality of life for its users.

1.2 Problem Statement

As urbanization increases and technology becomes more integrated into daily life, managing household environments effectively has become a significant challenge. Traditional home automation systems often lack flexibility, require complex setups, and may not fully address safety concerns. Many existing solutions do not provide real-time monitoring or intuitive user interfaces, limiting user engagement and control.

Additionally, homeowners face issues such as:

Limited Control: Conventional systems often rely on wired connections or cumbersome interfaces, making it difficult for users to manage devices remotely.

Safety Risks: Homes are vulnerable to gas leaks and changing light conditions, which can pose safety hazards if not monitored effectively.

Energy Inefficiency: Without real-time data on usage patterns, homeowners may struggle to optimize energy consumption, leading to higher utility bills and environmental impact.

Lack of Integration: Many automation systems operate in isolation, failing to communicate with other devices or sensors, which limits their functionality and effectiveness.

User Experience: Current systems may not provide adequate feedback or visual cues regarding device status, making it challenging for users to understand and interact with their environment.

To address these challenges, the Smart Home Automation system utilizing the Raspberry Pi Pico W aims to create an integrated, user-friendly solution that enhances control over household devices, prioritizes safety, and optimizes energy efficiency. By leveraging Bluetooth and IoT technologies, the system enables real-time monitoring, intuitive user interfaces, and the ability to respond dynamically to changing environmental conditions.

1.3 Objective of Project

The Smart Home Automation system powered by the Raspberry Pi Pico W aims to achieve the following objectives:

Enhanced Control: Provide users with seamless control over household devices, such as lights and fans, through a dedicated mobile application, facilitating both manual and automated operations.

Safety Monitoring: Integrate sensors like Light Dependent Resistors (LDR) and gas sensors to continuously monitor environmental conditions, ensuring prompt action in response to safety hazards, such as gas leaks or inadequate lighting.

Real-time Feedback: Utilize a Liquid Crystal Display (LCD) to offer real-time updates on sensor statuses, device operations, and alerts, enhancing user awareness and engagement with the system.

Data Logging and Analysis: Enable periodic data uploads to a remote server via WiFi, allowing for the collection and analysis of usage patterns and environmental conditions. This feature aids in optimizing energy consumption and improving safety measures.

User-Centric Design: Ensure the system is intuitive and user-friendly, making it easy for homeowners to navigate and utilize the automation features effectively.

Scalability and Flexibility: Design the architecture to support future enhancements, including the integration of additional sensors (e.g., temperature, humidity) and advanced functionalities such as machine learning for predictive analytics.

Energy Efficiency: Promote energy-saving practices by automating device management based on real-time data and user preferences, thereby reducing unnecessary power consumption.

Robust Security Features: Implement security protocols to safeguard user data and enhance the overall safety of the automation system.

By meeting these objectives, the project seeks to create a comprehensive smart home solution that improves convenience, safety, and energy efficiency in residential living spaces.

1.4 Goal of Project

The primary goal of the Smart Home Automation system using Raspberry Pi Pico W is to develop an intelligent and user-friendly home management solution that leverages Bluetooth and IoT technologies. This system aims to:

Facilitate Seamless Device Control: Enable users to easily control household appliances, such as lights and fans, remotely through a mobile application, ensuring convenience and accessibility.

Enhance Home Safety: Integrate sensor technologies to monitor environmental conditions (e.g., gas levels and light intensity) continuously, allowing for automatic responses to potential safety hazards.

Promote Energy Efficiency: Implement smart automation features that optimize energy consumption based on real-time data, reducing unnecessary power usage and supporting sustainable living.

Provide Real-time Monitoring: Offer users real-time feedback through an LCD display and remote server logging, enhancing awareness of home conditions and facilitating informed decision-making.

Ensure Future Scalability: Design the system with flexibility to incorporate additional sensors and features, ensuring adaptability to evolving user needs and technological advancements.

Ultimately, the project aims to transform traditional homes into smart environments, improving the quality of life through advanced automation, safety, and energy management.

CHAPTER 2

PROBLEM IDENTIFICATION

2.1 Existing System

The current landscape of home automation systems often includes standalone devices that provide limited functionality and integration. Most existing systems rely on separate hubs or apps for controlling devices, leading to several challenges:

Fragmentation: Users typically need multiple applications or interfaces to control different devices (e.g., smart lights, thermostats, security systems), resulting in a disjointed user experience.

Limited Control Options: Many systems offer only remote control via mobile apps, lacking manual override options or real-time feedback mechanisms, which can hinder usability.

Basic Sensor Integration: Existing systems may incorporate sensors, but often lack advanced features like real-time data analysis or automatic responses based on sensor readings, limiting their effectiveness in enhancing safety and energy efficiency.

High Costs: Many commercial solutions can be expensive, both in terms of initial investment and ongoing subscription fees for cloud services, making them less accessible to a broader audience.

Security Concerns: Some existing systems may not have robust security protocols, leaving them vulnerable to unauthorized access or cyber threats, raising concerns about user privacy and data protection.

Static Functionality: Existing systems may not easily accommodate additional sensors or features, making it challenging for users to adapt to their changing needs or technological advancements.

In summary, while current home automation systems offer some degree of control and convenience, they often fall short in terms of integration, user experience, adaptability, and security. The proposed Smart Home Automation system using Raspberry Pi Pico W aims to address these limitations by providing a more cohesive, user-friendly, and flexible solution for modern home management.

2.2 Proposed System

The proposed Smart Home Automation system leverages the Raspberry Pi Pico W to create a comprehensive and integrated home management solution. This system aims to overcome the limitations of existing automation solutions by incorporating advanced technologies and features:

1. Unified Control Interface: Users can control all household devices—such as lights, fans, and safety sensors—through a single mobile application, simplifying the user experience and reducing the need for multiple interfaces.

2. Bluetooth and IoT Integration: The system combines Bluetooth for local control and IoT capabilities for remote monitoring. Users can operate devices via Bluetooth commands, ensuring flexibility even when internet connectivity is limited.

3. Real-Time Sensor Feedback: The integration of sensors, including a Light Dependent Resistor (LDR) for ambient light detection and a gas sensor for safety monitoring, allows the system to make automatic adjustments based on environmental conditions. For example, the LDR can automatically turn lights on or off depending on the ambient light levels.

4. User-Friendly LCD Display: An LCD provides real-time feedback on sensor statuses, operational modes, and alerts. This visual interface enhances user interaction by making system states easily understandable at a glance.

5. Data Logging and Analysis: Sensor data is periodically uploaded to a remote server via WiFi. This feature enables users to monitor usage patterns and environmental conditions over time, supporting informed decision-making regarding energy consumption and safety.

6. Scalability and Flexibility: The architecture of the system is designed to be scalable, allowing for easy integration of additional sensors (e.g., temperature, humidity) and functionalities in the future. This adaptability ensures the system remains relevant as user needs evolve.

7. Enhanced Security Protocols: The system will implement robust security measures, including user authentication and secure data transmission, to safeguard against unauthorized access and protect user privacy.

8. Automation and Customization: Users can customize automation rules based on their preferences, allowing for a personalized smart home experience. For instance, users can set schedules for lights and fans or create conditions for alerts based on sensor readings.

In summary, the proposed Smart Home Automation system offers a holistic and user-centric approach to home management, combining convenience, safety, and energy efficiency. By addressing the shortcomings of existing systems, this solution exemplifies the potential of IoT technologies to transform residential living into a more intelligent and responsive environment.

CHAPTER – 3
REQUIREMENTS

3.1 Software Requirements:

1. Operating System:

- **Raspberry Pi OS:** The base operating system running on the Raspberry Pi Pico W.

2. Programming Environment:

- **IDE/Code Editor:** Tools like Arduino IDE or Visual Studio Code for writing and uploading code.

3.Device Communication:

- **Bluetooth Libraries:** For managing Bluetooth connections and commands.
- **WiFi Libraries**: To enable internet connectivity for data transmission and remote control.

4. Sensor Management:

- **Sensor Libraries:**
  - **LiquidCrystal Library** for interfacing with LCD.
  - Libraries for handling Light Dependent Resistor (LDR) and gas sensors.

5. User Interface:

- **LCD Display Management:** Code to manage the Liquid Crystal Display for real-time feedback.

6. Data Logging:

- **Data Transmission Scripts:** Code for periodically uploading sensor data to the server.

7. Security Protocols:

- Encryption Libraries: For securing data transmission over WiFi.

8. Testing Frameworks:

- Tools for unit testing and ensuring reliability of the software components.

9. Documentation Tools:

- Tools for maintaining project documentation and user manuals.

These components work together to create a seamless and interactive smart home automation experience, allowing for efficient management of household devices and monitoring of environmental conditions.

3.2 Hardware Requirements:

1. LCD Display: A liquid crystal display used for visual output, often to show information or user interfaces. It connects easily to microcontrollers for data display.

2. Raspberry Pi Pico W: A microcontroller board with Wi-Fi capabilities, ideal for IoT projects. It's programmable in Python and supports various peripherals.

3. Daughter or Link Board: A secondary board that expands the functionality of the main board (like the Pico W), allowing additional sensors or components to connect easily.

4. Speaker: An output device that converts electrical signals into sound. Useful for notifications or audio output in projects.

5. Fan: An electrical device used for cooling. It can be controlled via a relay to manage temperature in a system.

6. Gas Sensor: A sensor that detects gas leaks or concentrations of specific gases. It's essential for safety in various applications.

7. LDR Sensor: A Light Dependent Resistor that changes resistance based on light exposure. It's used for detecting ambient light levels.

8. 2 Channel Relay: A device that allows control of two separate circuits using a single control signal. Great for switching high-power devices.

9. Voice Module Speaker: A module that can playback recorded audio or respond to commands. Useful for interactive projects.

10. **IoT Server Module:** A module that enables connectivity and communication between IoT devices and a cloud server for data exchange.

11. Adapter: A device that converts power from one form to another, often used to power electronic components safely.

12.Holder: A physical support or enclosure for mounting or holding components securely in place.

13. Power Supply: A device that provides the necessary electrical power for components. Ensures stable operation.

14. Switch for Bluetooth and Sensors: A toggle switch that allows users to enable or disable Bluetooth and sensor functionalities, adding control to the setup.

15. Connecting Wires:

- Jumper wires for connecting sensors, relays, and the LCD to the Raspberry

These hardware components collectively enable the functionality of the smart home automation system, allowing for effective control and monitoring of household environments.

CHAPTER – 4
DESIGN AND IMPLEMENTATION

4.1 Design

System Architecture

The smart home automation system is designed with a modular architecture that integrates various components to create a cohesive environment. The main components include:

- **Microcontroller:** Acts as the central processing unit (CPU) for the system, managing inputs from sensors and controlling outputs to devices.
- **Sensors:** Various sensors, including:
  - **Light Sensor (LDR)**: Detects ambient light levels to control lighting.
  - **Gas Sensor:** Monitors air quality for harmful gases and activates ventilation.
- **Actuators:** Devices controlled by the microcontroller, such as:
  - **Lights:** Controlled based on light sensor readings.
  - **Fan:** Activated when gas is detected.
  - **Buzzer:** Alerts users to unsafe conditions.

User Interface

- **LCD Display**: Provides real-time feedback to users about the system's status, such as light and gas levels.

Connectivity:

- **Wi-Fi Module:** Enables communication with a remote server for data logging and user notifications.
- **Mobile App Integration:** Plans for a mobile app to control and monitor the system remotely, enhancing user convenience.

Power Supply

- The system is powered by a reliable power source with appropriate voltage

regulation to ensure the safe operation of all components.

Circuit Design

- A detailed circuit schematic outlines connections between the microcontroller, sensors, actuators, and the power supply.
- Breadboard or PCB layout is used for prototyping, ensuring efficient use of space and reliable connections.

Software Design

- Programming Language: The system is programmed using python.
- Control Logic: The software includes algorithms to process sensor data, control actuators based on predetermined conditions, and manage communication with the Wi-Fi module.
- User Interaction Logic: Code is implemented to handle user inputs from buttons and to update the LCD display accordingly.

This design provides a comprehensive foundation for the smart home automation system, ensuring functionality, user engagement, and potential for future expansion.

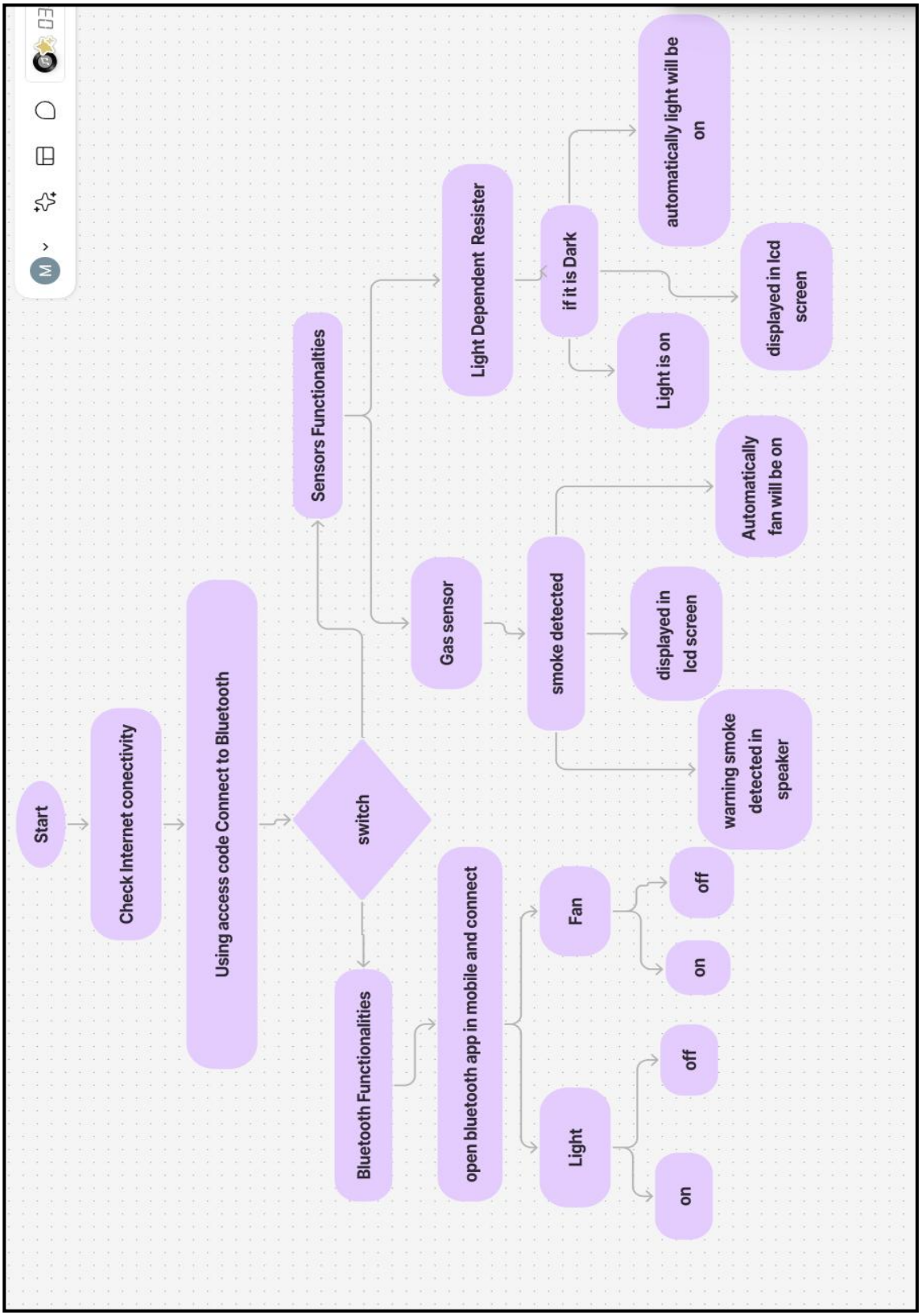


Fig.4.1.1:Architecture Design

4.2 IMPLEMENTATION

The implementation of our smart home automation project using Raspberry Pi involves several key steps, from hardware setup to software configuration.

· **Hardware Setup**

Components Required:

- Raspberry Pi
- Gas sensors (like MQ-2)
- Light-dependent resistor (LDR)
- LCD display
- Relay module for controlling devices (like fans)
- Buzzer for alarms
- Jumper wires and breadboard

· **Wiring**

- · Connect the gas sensor and LDR to the GPIO pins of the Raspberry Pi.
- Connect the LCD display using the appropriate pins (data and control).
- Wire the relay module to control the fan based on gas detection.
- Connect the buzzer to a GPIO pin for alerting purposes.

· **Software Setup**

- · **Libraries**: Install necessary libraries for handling the sensors and LCD.
- **Programming Environment**: Use an IDE like Thonny or directly edit files in the terminal.

· **Code Structure**·

Initialization:

- Set up GPIO pins for input (sensors) and output (LCD, relay, buzzer).
- Initialize the LCD display.
- Set up Wi-Fi or network connection for data logging (if applicable).

Main Loop:

- Continuously read values from the gas sensors and LDR.

- Display the status (light/dark and gas levels) on the LCD.
- Activate the fan and buzzer if gas levels exceed a certain threshold.
- Log data to a server or local file for analysis (optional).

· **Sensor Calibration**

- · Test the gas sensor in different environments to calibrate it for accurate readings.
- Adjust thresholds in the code based on the calibration results.

· **Testing**

- · Run the complete system to ensure that all components work as expected.
- Simulate gas detection and check if the fan activates and the buzzer sounds.

· **User Interface**

- · Design a simple web interface or use an existing dashboard to monitor the sensor data remotely.

Conclusion

By following these steps, we successfully implemented a smart home automation system using Raspberry Pi, enhancing both comfort and safety in our living environment. The flexibility of the Raspberry Pi allows for future expansions and modifications, making it an excellent choice for DIY smart home projects.

UML DIAGRAMS:

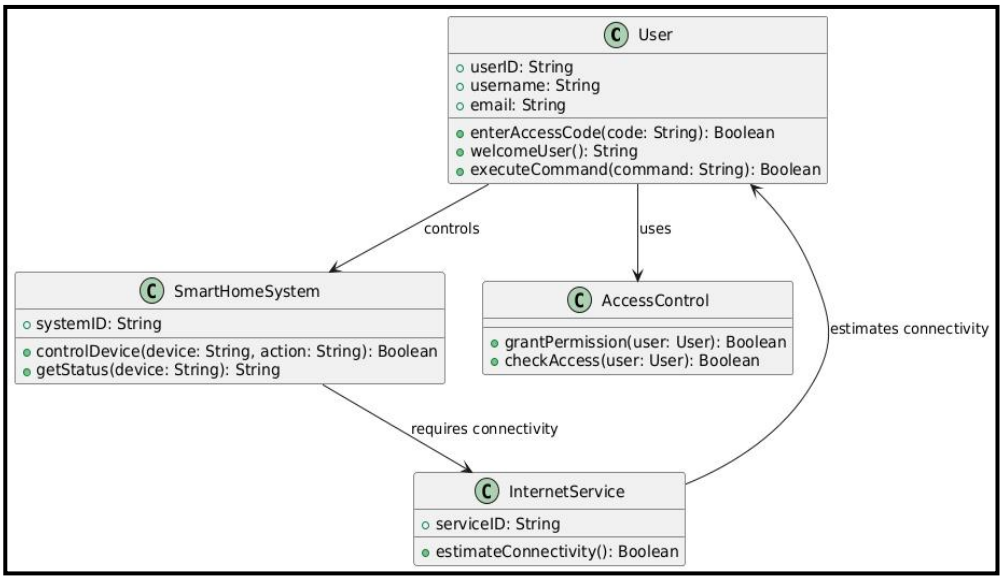


Fig.4.2.1:Class Diagram

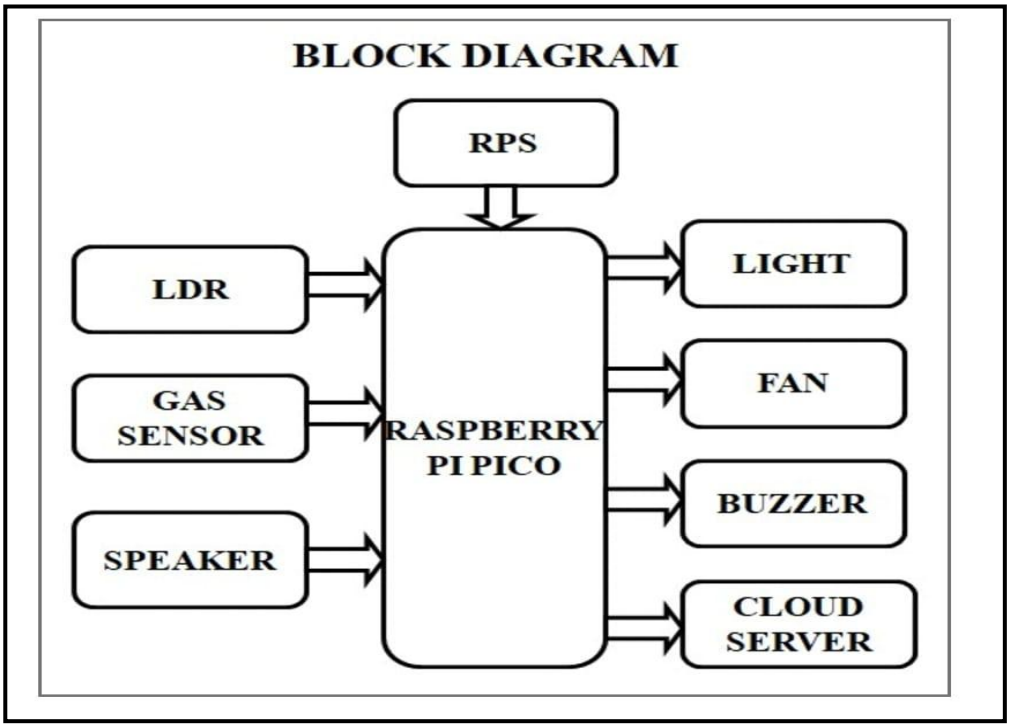


Fig.4.2.2:Block Diagram

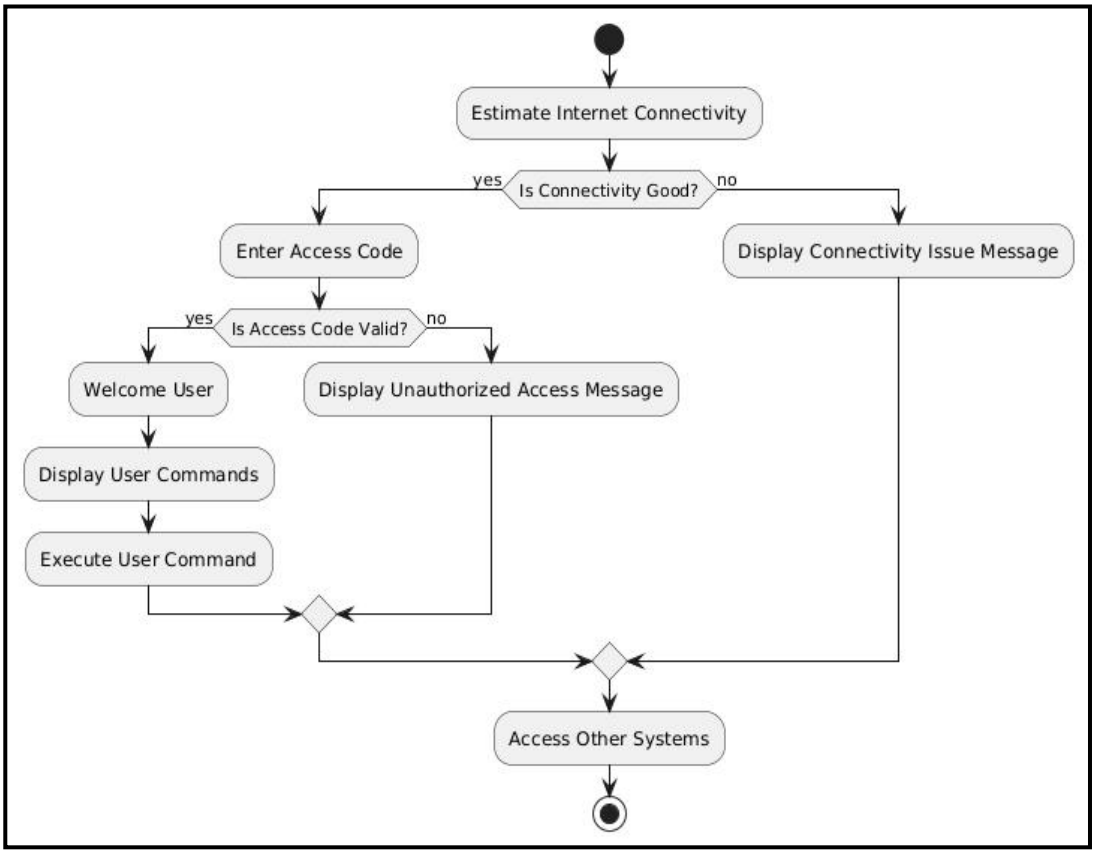


Fig.4.2.3:Activity Diagram

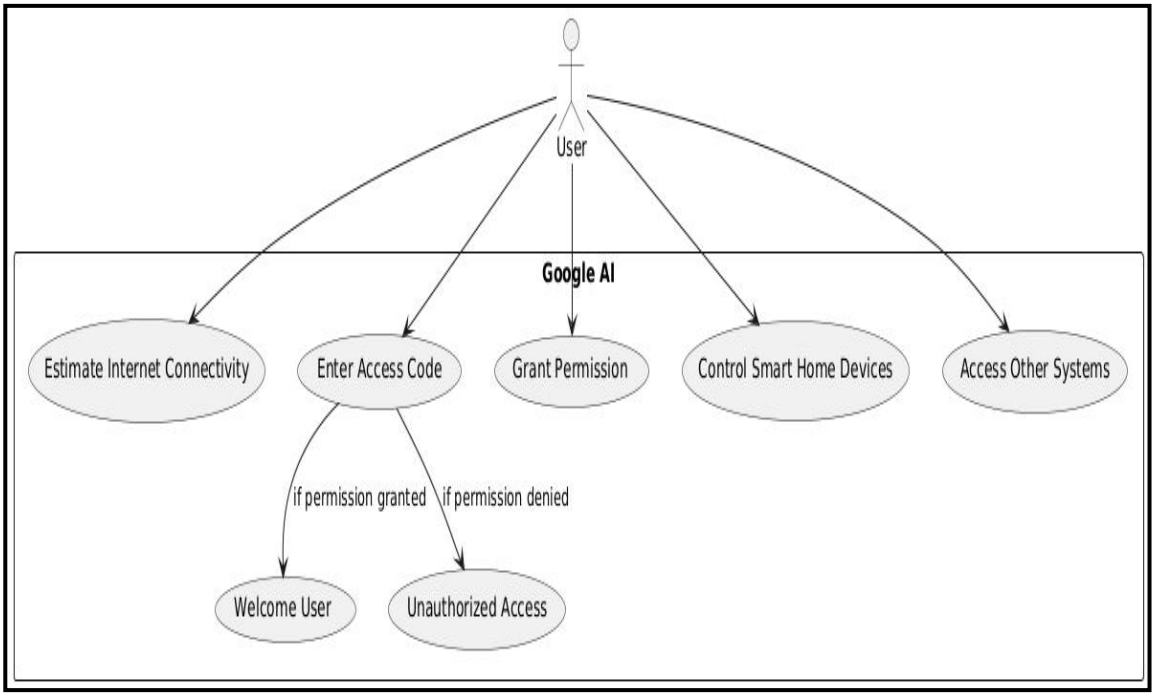


Fig.4.2.4:UseCaseDiagram

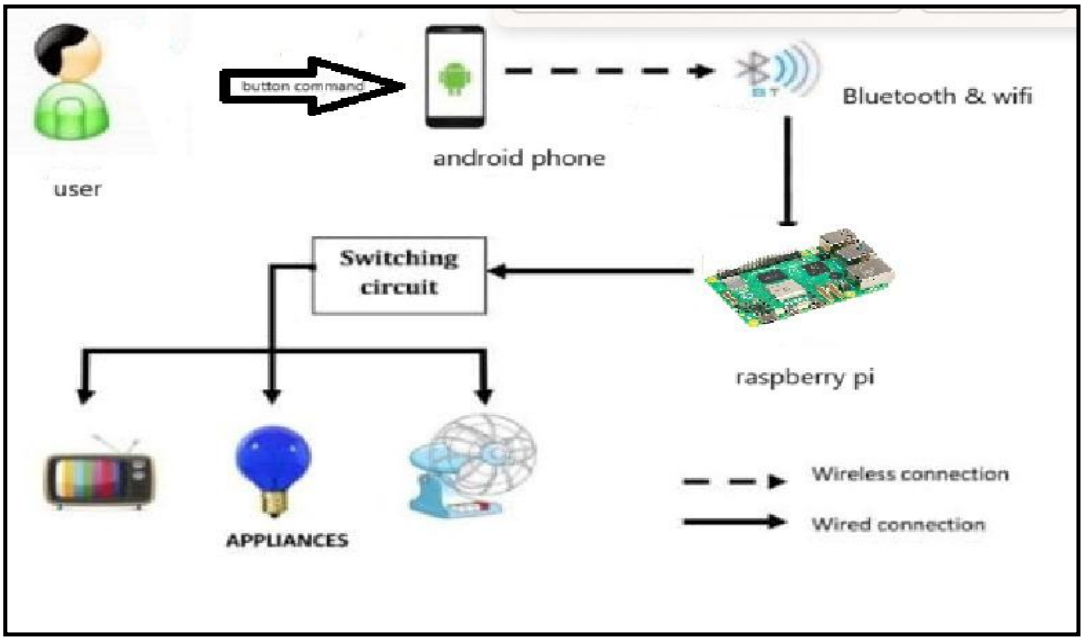


Fig.4.2.5:StructureDiagram

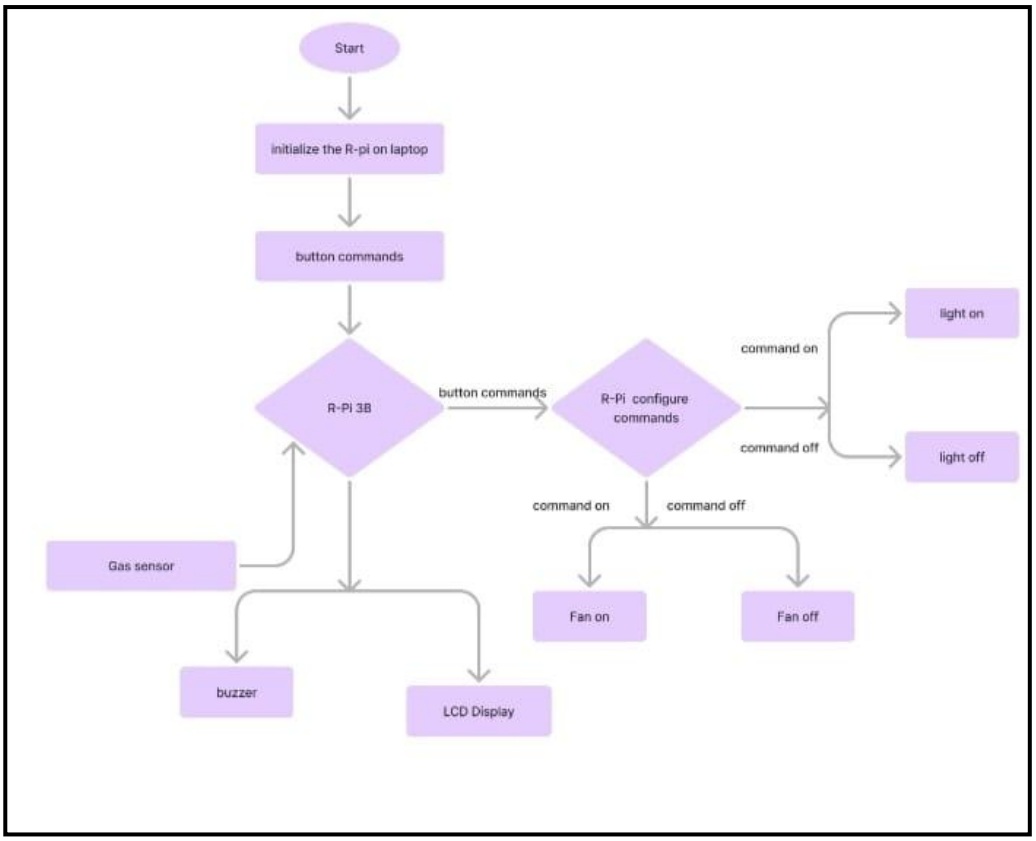


Fig.4.2.6:Flow Chart

CHAPTER – 5
CODE

5.1 Source Code

```
#include <LiquidCrystal.h>
#include <stdio.h>
LiquidCrystal lcd(2, 3, 4, 5, 6, 7);
/*
#include <Wire.h>
#include "dht.h"
#define dht_apin 10
dht DHT;
*/
unsigned char rcv,gchr;
char pastnumber[11];
int cntlmk=0;
char modes='x';
int tx1 = 0;
int rx1 = 1;
int tx2 = 8;
int rx2 = 9;
int ldr  = 11;
int gas  = 10;
int light = 12;
int fan   = 13;
int mode_sw  = 16;
int voice1 = 14;
int voice2 = 15;
int buzzer   = 22;
int sts1=0,sts2=0;
float tempc=0,humc=0;
float mq135v=0,soundv=0;
char res[130];
void serial1Flush()
{
  while(Serial1.available() > 0)
  {
    char t = Serial1.read();
  }
}
char check(char* ex,int timeout)
{
  int i=0;
  int j = 0,k=0;
  while (1)
  {
    sl:
    if(Serial1.available() > 0)
    {
      res[i] = Serial1.read();
```

```
      if(res[i] == 0x0a || res[i]=='>' || i == 100)
      {
        i++;
        res[i] = 0;break;
      }
      i++;
    }
    j++;
    if(j == 30000)
    {
      k++;
     // Serial2.println("kk");
      j = 0;
    }
    if(k > timeout)
    {
      //Serial2.println("timeout");
      return 1;
     }
  }//while 1
  if(!strncmp(ex,res,strlen(ex)))
  {
   // Serial2.println("ok..");
    return 0;
   }
  else
  {
   // Serial2.print("Wrong  ");
   // Serial2.println(res);
    i=0;
    goto sl;
   }
}
char buff[200],k=0;
void upload(unsigned int s1,unsigned int s2);
char readserver(void);
void clearserver(void);
const char* ssid = "iotserver";
const char* password = "iotserver123";
int sti=0;
String inputString = "";         // a string to hold incoming data
boolean stringComplete = false;  // whether the string is complete

int sti1=0;
String inputString1 = "";         // a string to hold incoming data
boolean stringComplete1 = false;  // whether the string is complete
void beep()
{
  digitalWrite(buzzer,HIGH);delay(2000);digitalWrite(buzzer,LOW);
}

void okcheck1()
{
```

```
  unsigned char rcr;
  do{
     rcr = Serial1.read();
    }while(rcr != 'K');
}
void setup()
{
  Serial1.setRX(rx1);
  Serial1.setTX(tx1);
  Serial1.begin(9600);
  Serial2.setRX(rx2);
  Serial2.setTX(tx2);
  Serial2.begin(9600);
  pinMode(ldr, INPUT);pinMode(gas, INPUT);
  pinMode(light, OUTPUT);pinMode(fan, OUTPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(mode_sw, INPUT_PULLUP);
  pinMode(voice1, OUTPUT);pinMode(voice2, OUTPUT);

  digitalWrite(light, LOW);digitalWrite(fan, LOW);
  digitalWrite(buzzer, LOW);
  digitalWrite(voice1, HIGH);digitalWrite(voice2, HIGH);
  //IOT based cola mine
  lcd.begin(16, 2);lcd.cursor();
  lcd.print("    Welcome   ");
  //lcd.setCursor(0,1);
  //lcd.print("Safety Monitoring");
     delay(2000);
   wifiinit();
     delay(2500);
   lcd.clear();
   lcd.setCursor(0,0);
   lcd.print("L:");//2,0
   lcd.setCursor(8,0);
   lcd.print("G:");//10,0
}
char bf3[50];
int g=0,f=0,count=0,lc=0;
char ldr_string[15];
char gas_string[15];
void loop()
{
    memset(ldr_string,'\0',strlen(ldr_string));
    if(digitalRead(ldr) == HIGH)
      {
        lcd.setCursor(2,0);lcd.print("Dark ");
        strcpy(ldr_string,"Dark");
        if(modes == 's')
          {
           digitalWrite(light, HIGH);
           sts1++;
           if(sts1 >= 2){sts1=2;}
           if(sts1 == 1)
```

```
        {
         digitalWrite(voice1, LOW);
                  delay(2500);
         digitalWrite(voice1, HIGH);
        }
      }
  }
if(digitalRead(ldr) == LOW)
  {
   lcd.setCursor(2,0);lcd.print("Light ");
   strcpy(ldr_string,"Light");
   if(modes == 's')
     {
      digitalWrite(light, LOW);
      sts1=0;
     }
  }
memset(gas_string,'\0',strlen(gas_string));
if(digitalRead(gas) == LOW)
  {
   lcd.setCursor(10,0);lcd.print("ON ");
   strcpy(gas_string,"ON");
   if(modes == 's')
     {
      digitalWrite(fan, HIGH);
      sts2++;
      if(sts2 >= 2){sts2=2;}
      if(sts2 == 1)
        {
         digitalWrite(voice2, LOW);
                  delay(2500);
         digitalWrite(voice2, HIGH);
        }
     }
  }
if(digitalRead(gas) == HIGH)
  {
   lcd.setCursor(10,0);lcd.print("OFF");
   strcpy(gas_string,"OFF");
   if(modes == 's')
     {
      digitalWrite(fan, LOW);
      sts2=0;
     }
  }
if(digitalRead(mode_sw) == LOW)
  {
   modes='b';
   lcd.setCursor(15,0);lcd.print("B");
  }
if(digitalRead(mode_sw) == HIGH)
  {
   modes='s';
```

```
      lcd.setCursor(15,0);lcd.print("S");
    }

 while(Serial2.available() > 0)
    {
     char chrt = (char)Serial2.read();
     if(chrt == '1' && modes == 'b')
       {
        digitalWrite(light, HIGH);
       }
     if(chrt == '2' && modes == 'b')
       {
        digitalWrite(light, LOW);
       }

     if(chrt == '3' && modes == 'b')
       {
        digitalWrite(fan, HIGH);
       }
     if(chrt == '4' && modes == 'b')
       {
        digitalWrite(fan, LOW);
       }
    }
   delay(1000);
   cntlmk++;
   if(cntlmk > 40)
    {
     upload(ldr_string,gas_string);
    }
}

char bf2[50];
void upload(const char *s1,const char *s2)
{
 delay(2000);
 lcd.setCursor(15, 1);lcd.print("U");
 serial1Flush();
 Serial1.println("AT+CIPSTART=4,\"TCP\",\"projectsfactoryserver.in\",80");

 //http://projectsfactoryserver.in/storedata.php?name=pf5&s1=25&s2=35
 //sprintf(buff,"GET
http://embeddedspot.top/iot/storedata.php?name=iot139&s1=%u&s2=%u&s3=%u\r\n\
r\n",s1,s2);
    delay(8000);
//https://projectsfactoryserver.in/storedata.php?name=iotgps&lat=17.167898&lan=79.
785643
    memset(buff,0,strlen(buff));
    sprintf(buff,"GET
http://projectsfactoryserver.in/storedata.php?name=iot962&s1=%s&s2=%s\r\n\r\n",s1,
s2);

//     memset(buff,0,strlen(buff));
```

```
  //    sprintf(buff,"GET
http://projectsfactoryserver.in/storedata.php?name=iot4&s1=%s\r\n\r\n",s1);

      serial1Flush()
         serial1Flush();
         Serial1.print(buff);
            delay(2000);
         Serial1.println("AT+CIPCLOSE");
       lcd.setCursor(15, 1);lcd.print(" ");
}
char readserver(void)
{
  char t;
  delay(2000);
  lcd.setCursor(15, 1);lcd.print("R");
  serial1Flush();
  Serial1.println("AT+CIPSTART=4,\"TCP\",\"projectsfactoryserver.in\",80");
       delay(8000);
      memset(buff,0,strlen(buff));
      sprintf(buff,"GET http://projectsfactoryserver.in/last.php?name=iot4L\r\n\r\n");
      serial1Flush();
      sprintf(bf2,"AT+CIPSEND=4,%u",strlen(buff));
      Serial1.println(bf2);
         delay(5000);
         serial1Flush();
         Serial1.print(buff);
       //read status
        while(1)
        {
          while(!Serial1.available());
          // Serial2.print(t);
           if(t == '*' || t == '#')
           {
            if(t == '#')return 0;
            while(!Serial1.available());
             t = Serial1.read();
           //  Serial.print(t);
             delay(1000);
              serial1Flush();
             return t;
           }
        }
             delay(2000);

      Serial1.println("AT+CIPCLOSE");
      lcd.setCursor(15, 1);lcd.print(" ");
      delay(2000);
return t;
}
void clearserver(void)
{
  delay(2000);
  lcd.setCursor(15, 1);lcd.print("C");
```

```
  serial1Flush();
  Serial1.println("AT+CIPSTART=4,\"TCP\",\"projectsfactoryserver.in\",80");

  //sprintf(buff,"GET
http://projectsfactoryserver.in/storedata.php?name=iot1&s10=0\r\n\r\n");
      delay(8000);
      memset(buff,0,strlen(buff));
      sprintf(buff,"GET
http://projectsfactoryserver.in/storedata.php?name=iot4&s10=0\r\n\r\n");
      serial1Flush();
      sprintf(bf2,"AT+CIPSEND=4,%u",strlen(buff));
      Serial1.println(bf2);
         delay(5000);
          serial1Flush();
          Serial1.print(buff);
          delay(2000);
          serial1Flush();
       Serial1.println("AT+CIPCLOSE");
       lcd.setCursor(15, 1);lcd.print(" ");
       delay(2000);
}
void wifiinit()
{
 char ret;
  st:
  Serial1.println("ATE0");
  ret  = check((char*)"OK",50);
  Serial1.println("AT");
     lcd.clear();lcd.setCursor(0, 0);lcd.print("CONNECTING");
  Serial1.println("AT+CWMODE=1");
   ret  = check((char*)"OK",50);
 cagain:
  serial1Flush();
  Serial1.print("AT+CWJAP=\"");
  Serial1.print(ssid);
  Serial1.print("\",\"");
  Serial1.print(password);
  Serial1.println("\"");
  if(check((char*)"OK",300))goto cagain;
  Serial1.println("AT+CIPMUX=1");
  delay(1000);
  lcd.clear();lcd.setCursor(0, 0);lcd.print("WIFI READY");
}
/*
void wifiinit()
{
  Serial1.write("AT\r\n");              delay(3000);lcd.clear();lcd.print("WIFI-1");
  Serial1.write("ATE0\r\n");            delay(3000);lcd.clear();lcd.print("WIFI-2");
  Serial1.write("AT+CWMODE=3\r\n");     delay(3000);lcd.clear();lcd.print("WIFI-
3");
  Serial1.write("AT+CWJAP=\"iotserver\",\"iotserver123\"\r\n");
delay(5000);lcd.clear();lcd.print("WIFI-4");
  Serial1.write("AT+CIPMUX=1\r\n");delay(3000);     lcd.clear();lcd.print("WIFI-
```

```
5");delay(2000);
  lcd.clear();
  lcd.print("WIFI Connected");
  delay(1000);
}
*/
void converts(unsigned int value)
{
  unsigned int a,b,c,d,e,f,g,h;
      a=value/10000;
      b=value%10000;
      c=b/1000;
      d=b%1000;
      e=d/100;
      f=d%100;
      g=f/10;
      h=f%10;
      a=a|0x30;
      c=c|0x30;
      e=e|0x30;
      g=g|0x30;
      h=h|0x30;
//   Serial1.write(a);
   Serial1.write(c);
   Serial1.write(e);
   Serial1.write(g);
   Serial1.write(h);
}

void convertl(unsigned int value)
{
  unsigned int a,b,c,d,e,f,g,h;

      a=value/10000;
      b=value%10000;
      c=b/1000;
      d=b%1000;
      e=d/100;
      f=d%100;
      g=f/10;
      h=f%10;
      a=a|0x30;
      c=c|0x30;
      e=e|0x30;
      g=g|0x30;
      h=h|0x30;
  // lcd.write(a);
   lcd.write(c);
   lcd.write(e);
   lcd.write(g);
   lcd.write(h);
}
```

5.2 Screenshot of Application

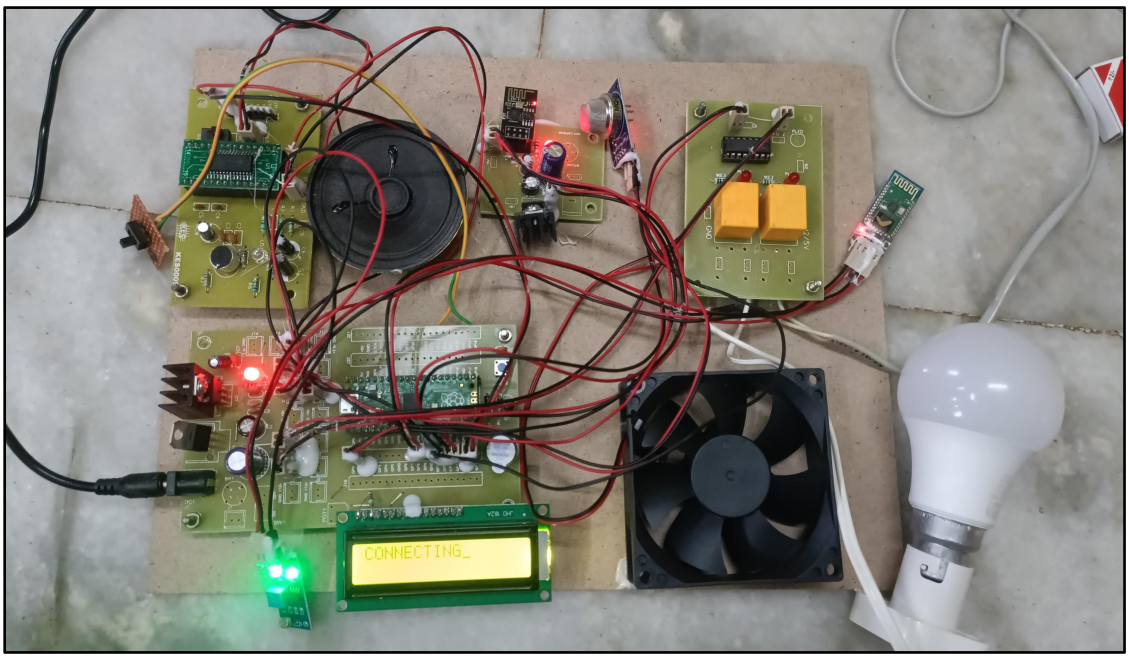


Fig.5.2.1:Circuit Digram

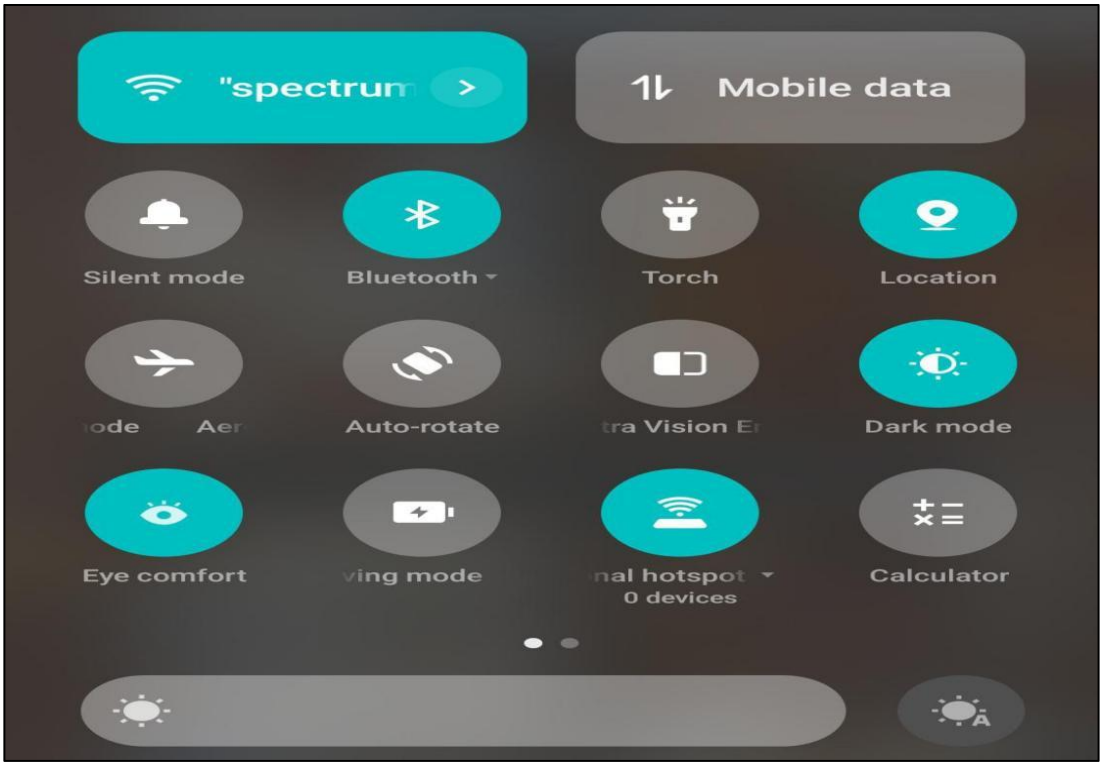


Fig.5.2.2:Enabling Wifi ,Hotspot and Bluetooth

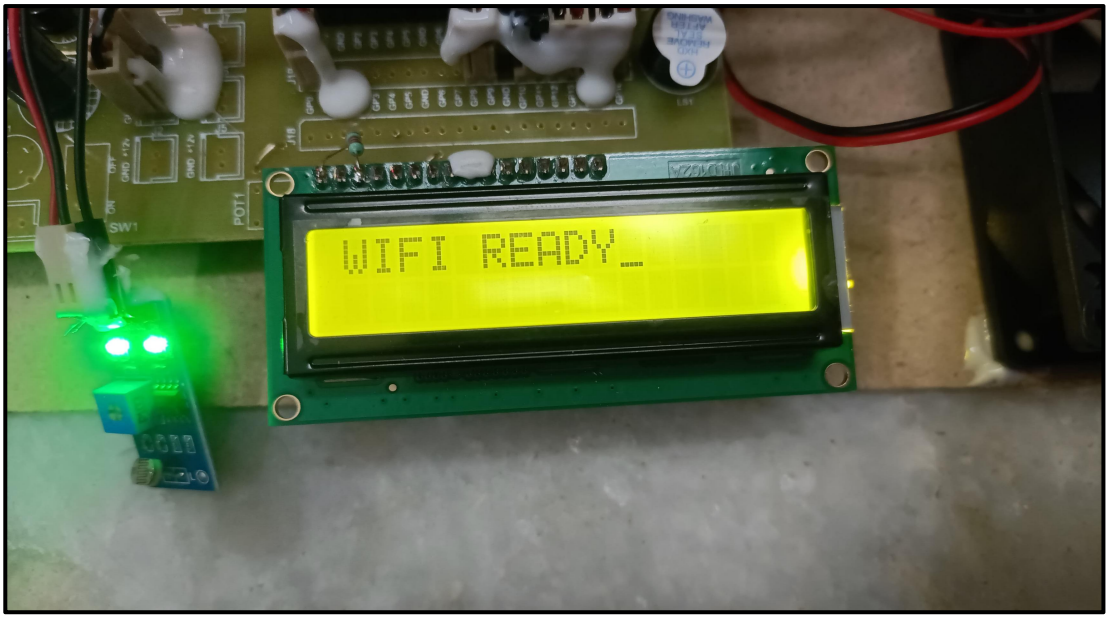


Fig.5.2.3: Wifi Ready

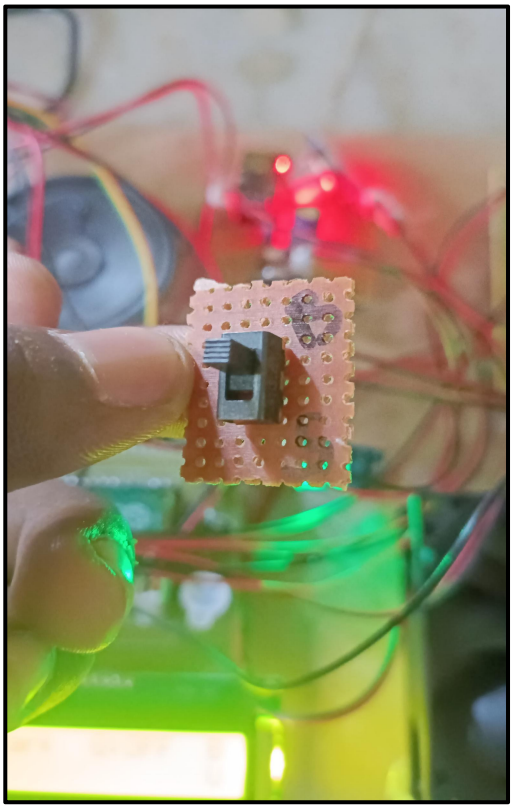


Fig.5.2.4: Switching to Bluetooth

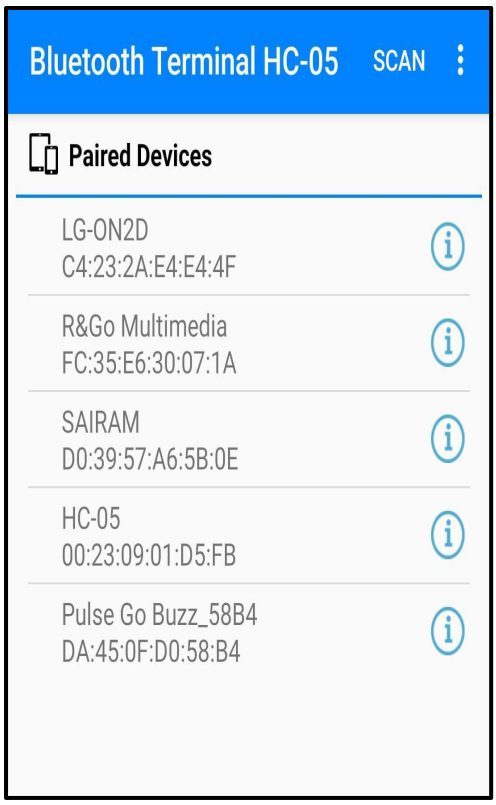


Fig.5.2.5: App UserInterface

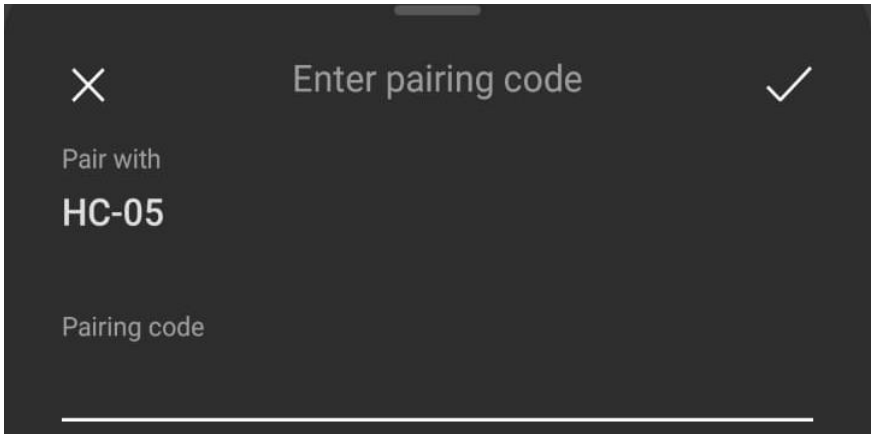


Fig.5.2.6: Enter Access Code

Fig.5.2.7:Commond for Light On

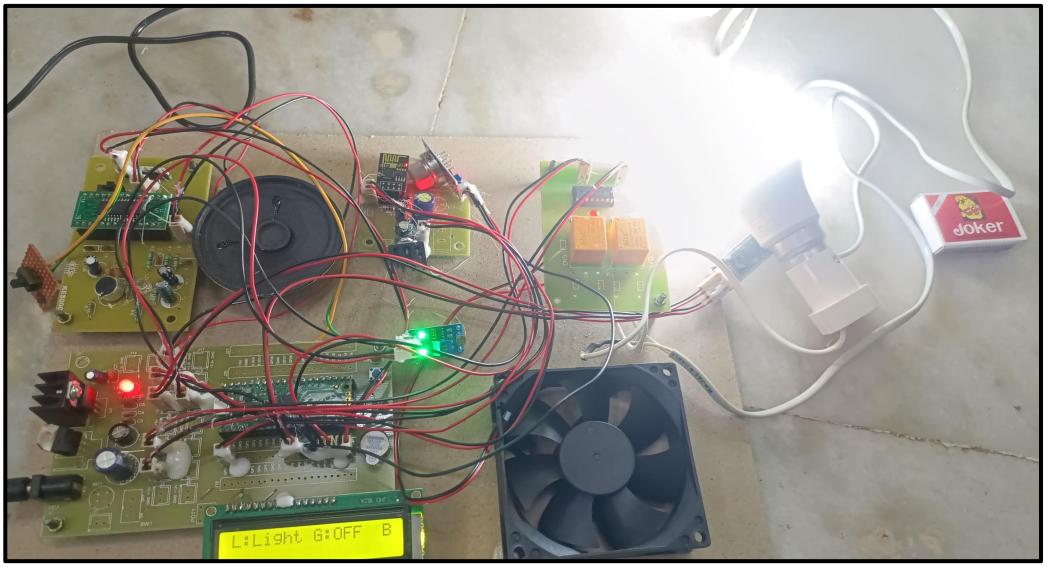


Fig.5.2.8:Light On

Fig.5.2.9:Commond for Light off

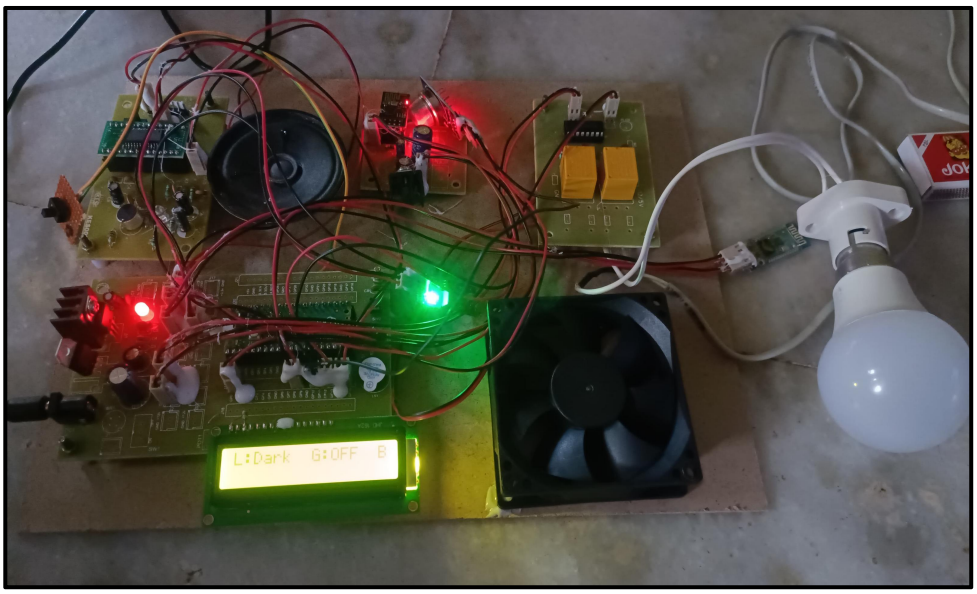


Fig.5.2.10:Light Off

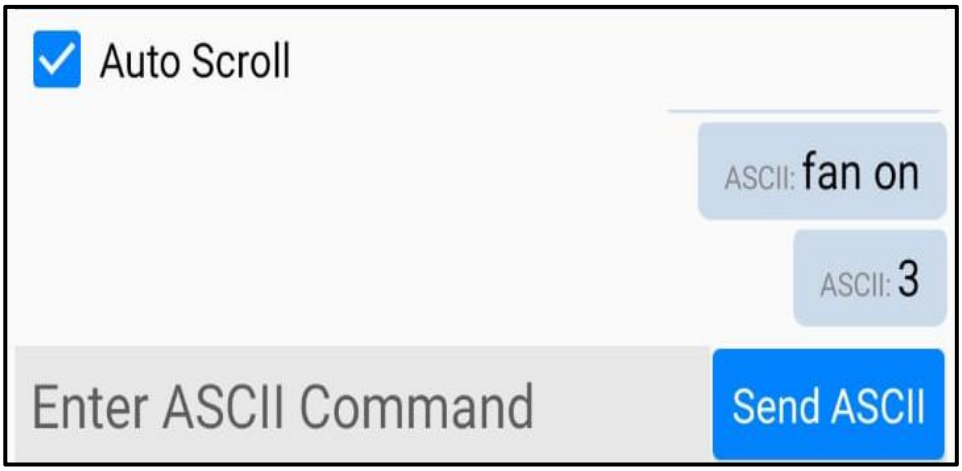


Fig.5.2.11:Commond for Fan On

Fig.5.2.12:Fan On

Fig.5.2.13:Commond for Fan Off

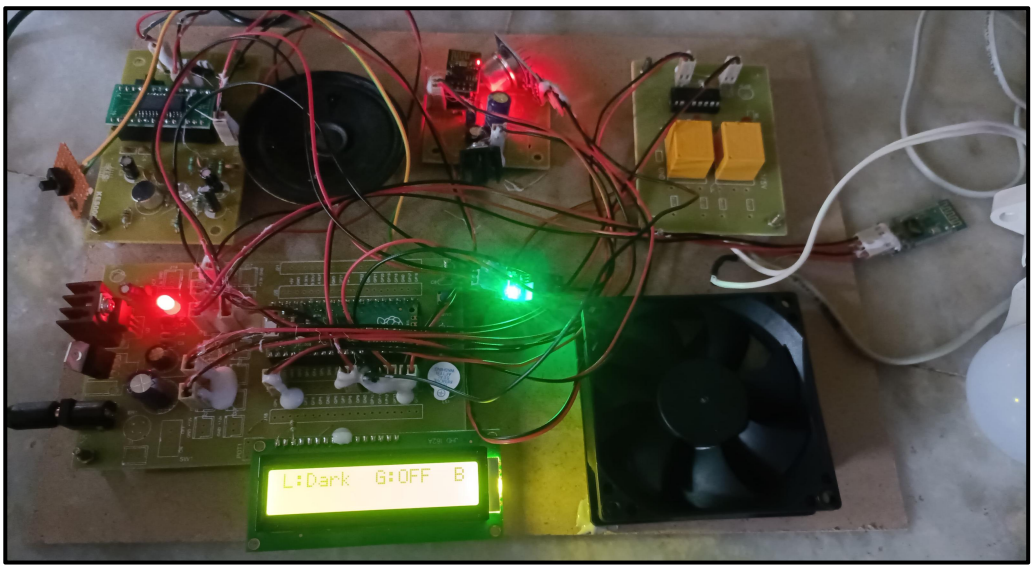


Fig.5.2.14:Fan Off

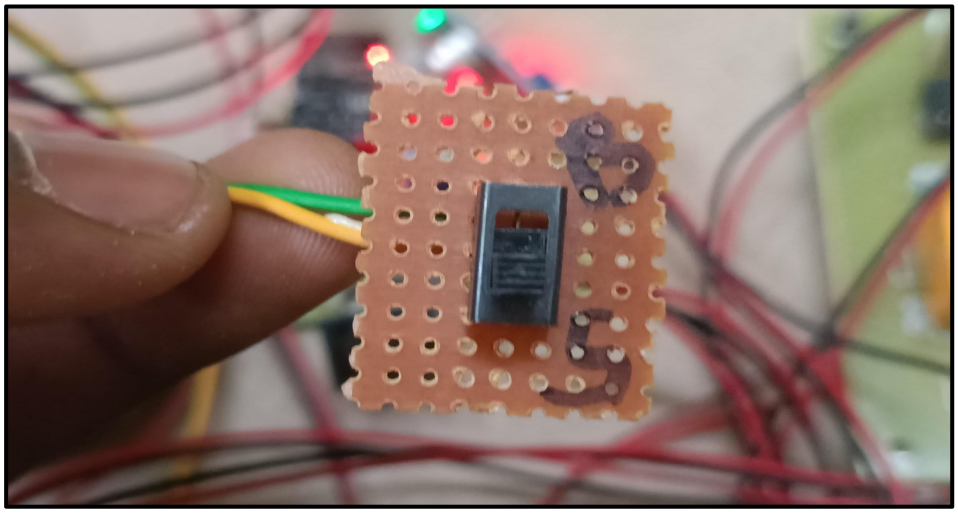


Fig.5.2.15: Switching to Sensors

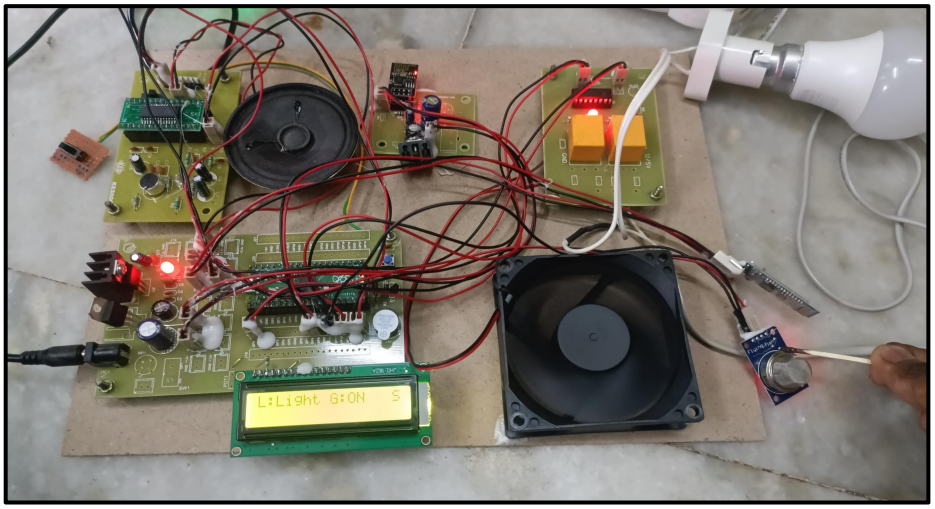


Fig.5.2.16: Testing Gas Sensor

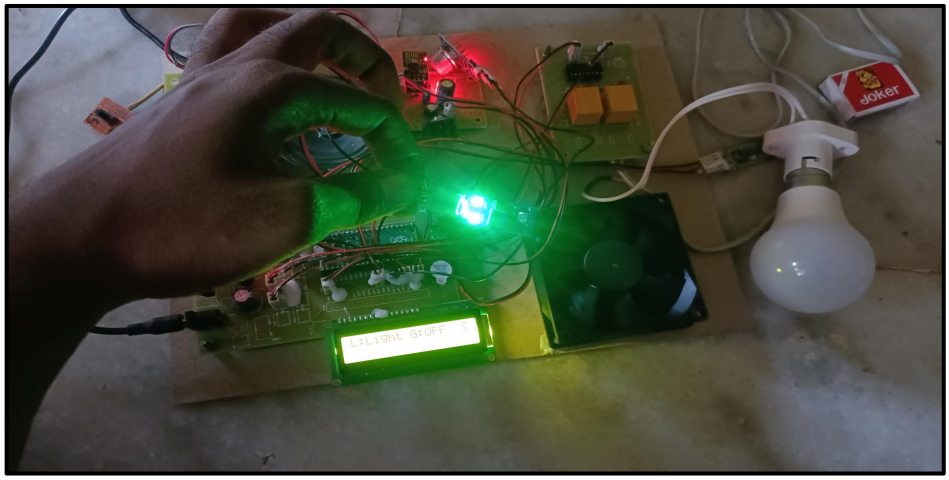
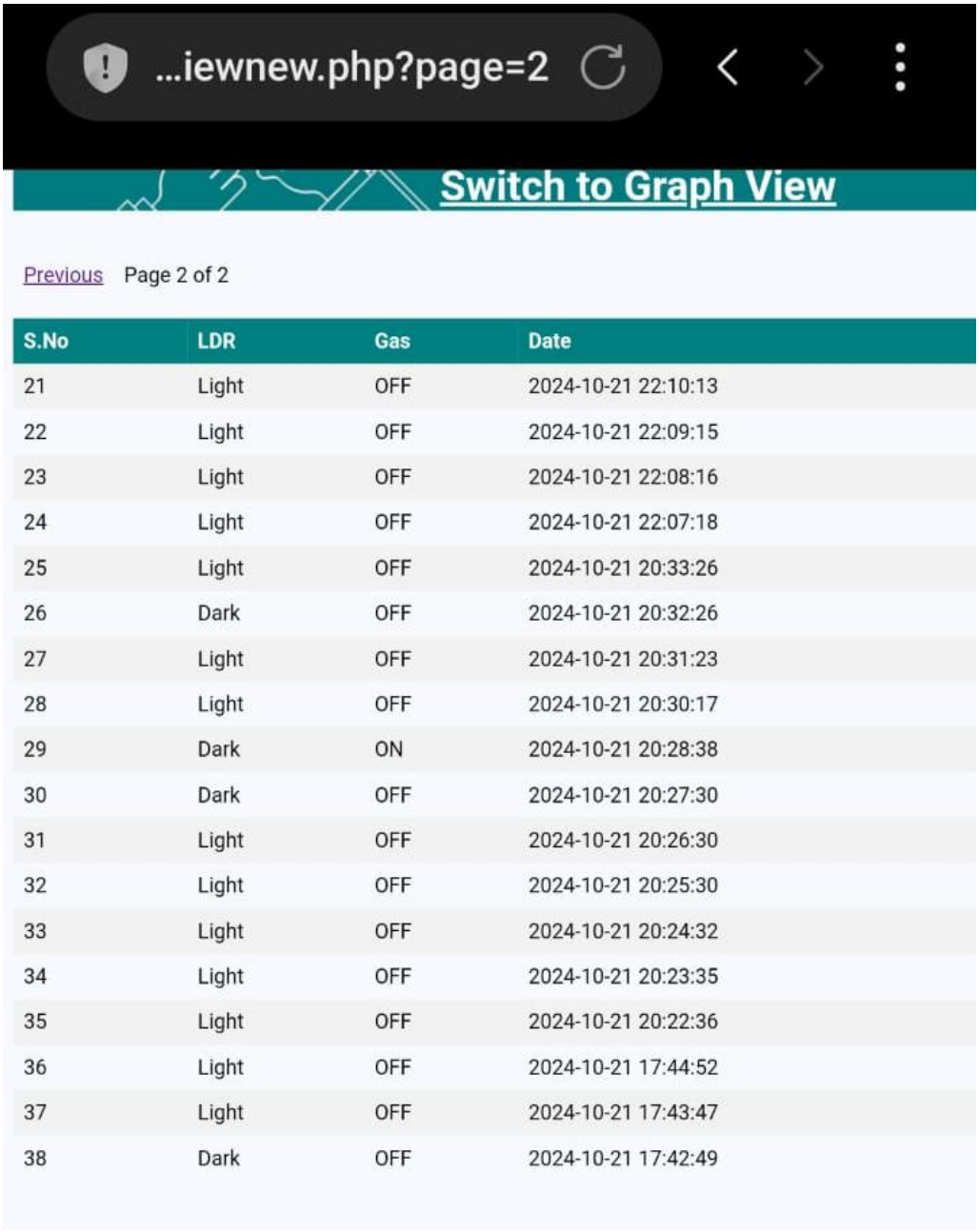


Fig.5.2.17: Testing LDR Sensor



S.No	LDR	Gas	Date
21	Light	OFF	2024-10-21 22:10:13
22	Light	OFF	2024-10-21 22:09:15
23	Light	OFF	2024-10-21 22:08:16
24	Light	OFF	2024-10-21 22:07:18
25	Light	OFF	2024-10-21 20:33:26
26	Dark	OFF	2024-10-21 20:32:26
27	Light	OFF	2024-10-21 20:31:23
28	Light	OFF	2024-10-21 20:30:17
29	Dark	ON	2024-10-21 20:28:38
30	Dark	OFF	2024-10-21 20:27:30
31	Light	OFF	2024-10-21 20:26:30
32	Light	OFF	2024-10-21 20:25:30
33	Light	OFF	2024-10-21 20:24:32
34	Light	OFF	2024-10-21 20:23:35
35	Light	OFF	2024-10-21 20:22:36
36	Light	OFF	2024-10-21 17:44:52
37	Light	OFF	2024-10-21 17:43:47
38	Dark	OFF	2024-10-21 17:42:49

Fig.5.2.18:Server

CHAPTER – 6
RESULTS & CONCLUSION

6.1 Results

Enhanced Safety

The system effectively monitored gas levels and ambient light conditions, triggering alarms and automatic responses to ensure user safety. The gas detection feature activated the ventilation fan immediately upon detecting hazardous levels, significantly reducing the risk of accidents.

Increased Convenience

Users reported a notable increase in comfort and ease of living. The ability to control lights and fans remotely allowed for tailored environmental settings based on personal preferences, enhancing daily routines.

Energy Efficiency

The automation of lighting based on occupancy and ambient light levels resulted in a measurable decrease in energy consumption. Users observed reduced electricity bills due to the system's ability to turn off lights when rooms were unoccupied.

Scalability and Flexibility

The modular design of the system allowed for easy integration of additional components, such as more sensors or smart appliances. This flexibility supports future upgrades and the ability to expand the system based on user needs and technological advancements.

6.2 Conclusion

In conclusion, the smart home automation system successfully demonstrates the integration of advanced technologies to enhance the comfort, security, and energy efficiency of residential spaces. By utilizing sensors to monitor environmental conditions and automated responses to potential hazards, the system not only ensures a safer living environment but also provides users with greater control over their home settings. The use of an intuitive interface and Wi-Fi connectivity further simplifies operation, making it accessible to a wide range of users.

Future developments in smart home automation could incorporate artificial intelligence to create adaptive systems that learn user preferences and adjust settings automatically for maximum comfort and efficiency. Furthermore, the integration of renewable energy sources and smart grids will pave the way for sustainable living, allowing homes to not only consume but also generate energy.

REFERENCES

[1] https://www.hackster.io/JuanIgnacioSanchezLara/ras pberry-pi-home-automation- with-wireless-sensors- 37d47d

[2] https://www.hackster.io/arturo182/home- automation-with-raspberry-pi-4-1447d1

[3] https://www.hackster.io/ben-nuttall/raspberry-pi-home-automation-control-lights- computers-cctv- and-more-725cc0

[4] Andrew K. Dennis. (2016). "Home Automation with Raspberry Pi: Projects Using Google Home, Amazon Echo, and Other Intelligent Personal Assistants." Apress.

[5] MarcoSchwartz.(2014)."HomeAutomationwith Raspberry Pi: Build Simple Raspberry Pi Projectsand Learn Home Automation." CreateSpace Independent Publishing Platform.