INTERNET OF THINGS
SMART HOME AUTOMATION SYSTEM

Submitted by:
Simran Goel 20BEE0359
Shreya Gupta 20BCT0165
Apoorv Gupta 20BCT0117
Veerayya Vastrad 20BEC0741

INTRODUCTION

Overview

Smart home automation refers to the integration of intelligent technologies and devices to automate and control various aspects of a home, transforming it into a connected and intelligent living space. It combines hardware, software, and communication technologies to enable homeowners to manage and monitor their homes remotely, enhance convenience, increase energy efficiency, improve security, and create personalized experiences.
In a smart home automation system, devices and systems such as lighting, thermostats, door locks, surveillance cameras, appliances, entertainment systems, and more are interconnected and can be controlled and monitored through a central hub or mobile applications.

Purpose

Smart home systems allow users to control their day-to-day activities, tasks with ease. The purpose of home automation systems worldwide is to provide convenience by automation rules i.e., specific actions carried out on triggers or events, such as turning off lights when a room is unoccupied.
Smart home automation systems offer compatibility and integration with a wide range of smart devices and brands. Hence, systems often can add devices, and sensors accordingly. For example, few systems integrate security features such as surveillance cameras, door/window sensors, and smart locks. Users can receive real-time alerts, monitor their homes remotely, and take action to enhance the security of their property.

LITERATURE SURVEY

In [2], the application has been developed based on the android system. An interface card has been developed to ensure communication between the remote user, server, raspberry pi card and the home Appliances. The application has been installed on an android Smartphone, a web server, and a raspberry pi card to control the shutter of windows. Android applications on a smartphone issue commands to raspberry pi cards. An interface card has been realized to update signals between the actuator sensors and the raspberry pi card.

The home network which monitors the appliances and sensors and transmits data to the cloud-based data server which manages the information and provides services for users by transmitting data and receiving user commands from mobile applications [7]. The proposed system has good modularity and configurability characteristics with very low power consumption in a cost efficient way.

Shih-Pang Tseng et al. [5] proposed Smart House Monitor & Manager (SHMM), based on the ZigBee, all sensors and actuators are connected by a ZigBee wireless network. They designed a simple smart socket, which can remote control via ZigBee. The PC host is used as a data collector and the motion sensing, all sensing data is transferred to the VM in the cloud. The user can use the PC or Android phone to monitor or control through the Internet to power-saving of the house

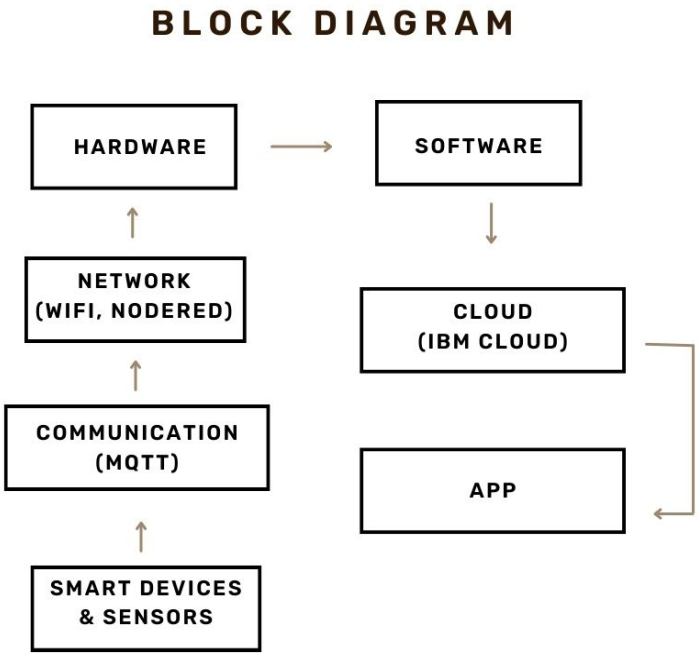
It has been designed with Arduino boards with Bluetooth boards developed for home automation [16]. A Python program is used on the cell phone to provide the user interface. The Bluetooth board has I/O ports and relays are used for interfacing with the devices which are to be controlled and monitored. The Bluetooth is password protected to ensure that the system is secure from intruders. The Bluetooth has a range of 10 to 100.

Embedded system Raspberry Pi to serve as a communication gateway between mobile devices and Konnex-Bus (KNX) home automation systems [9]. Store the information of all actors and sensors within a Smart Home, instead of using separate profiles. Ensures energy-consumption could be reduced, compared to a standard desktop computer.

Application developed using the Android platform controlled and monitored from a remote location using the smart home app and an Arduino Ethernet based micro web-server [8]. The sensors and actuators/relays are directly interfaced to the main controller. Proposed design offers are the control of energy management systems such as lightings, heating, air conditioning, security, fire detection and intrusion detection with siren and email notifications

It has been implemented with Raspberry Pi through reading the subject of Email and the algorithm. Raspberry Pi proves to be a powerful, economic and efficient platform for implementing smart home automation [4].

THEORETICAL ANALYSIS



Hardware Requirements

Central Control Hub/Controller: A main hardware component like ESP32 would connect to various devices and sensors and allow for centralized control and management.
Smart Devices and Sensors: Depending on the automation tasks and functionalities, we require specific smart devices and sensors. In our project, we have used Wokwi for the simulation of smart locks, ultrasonic sensor, buzzer, servomotor (for demonstrating door lock), LED(for Smart lighting), keypad, push button, buzzer and LCDs.
Communication Technologies: The choice of communication technologies depends on the specific devices and protocols that we use in our project. We have used MQTT(Message Queuing Telemetry Transport), an extremely lightweight, publish-subscribe messaging transport protocol and Wi-Fi for communication.
Network Infrastructure: The network infrastructure consists of the home network with the devices. A Wi-Fi network or wired connection ensures the stability of the network. In our project, we have used Node-Red for wiring together hardware devices, APIs and online services.

Software Requirements

Mobile App or Web Interface: A user-friendly mobile application or web interface is necessary to control and monitor the home automation system. The software should provide an intuitive interface for

managing devices, setting up automation rules, and accessing system settings.We have used MIT app inventor for our project.

Cloud Connectivity: To provide remote access and control of our home automation system, we require cloud connectivity. Cloud-based services allow us to access and control the system remotely over the internet, providing flexibility and convenience. We have used the IBM cloud for our project. We have also used Firebase for storing data.

Security and Encryption: Robust software security measures are necessary to protect home automation systems from unauthorized access. This includes implementing secure communication protocols(like TCP, HTTPS protocol for secure web communication, etc), user authentication mechanisms(login in the app), and encryption of sensitive data(in rest and during transit).

EXPERIMENTAL INVESTIGATIONS

Various components in this project “Home Automation System”, were discussed and then implemented on Wokwi. Firstly, the sensors and devices were connected with ESP32. Once all the parts like Doorbell, Door lock, lights and Security system were working all together as hardware in Wokwi, we needed to publish the data in the IBM cloud.

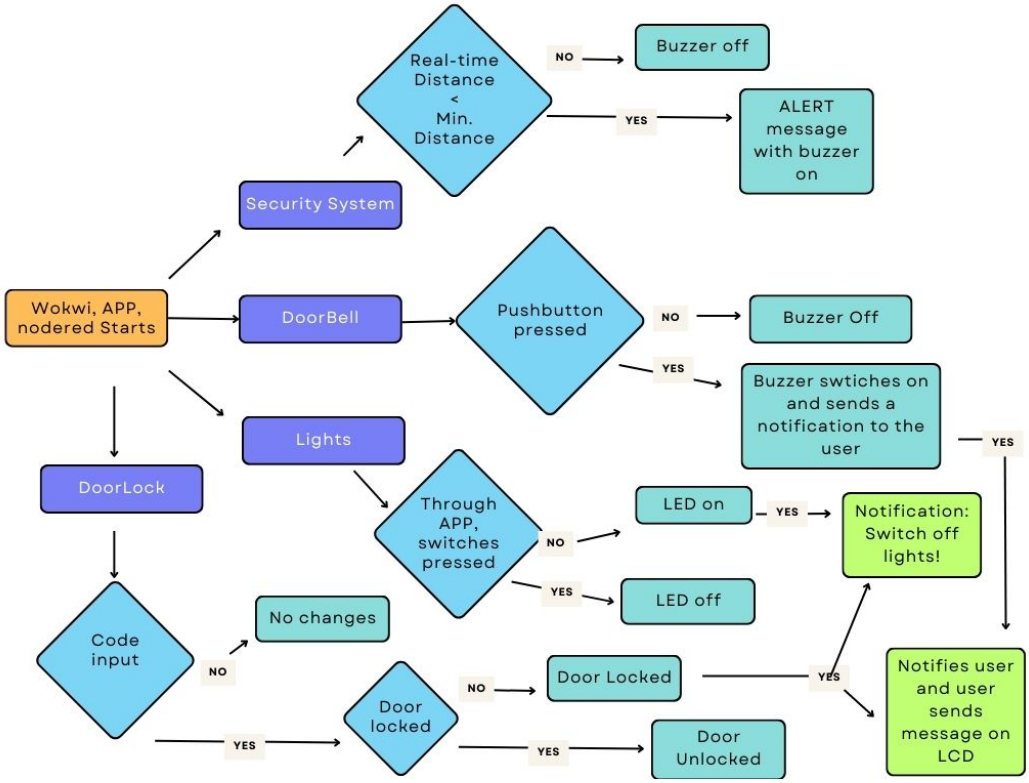
As ESP32 has fewer GPIO pins compared to an Arduino UNO, the team’s choice of using two buzzers (one for the DoorBell and one for the Alert) was not possible as we needed to accommodate two LCDs. Even though we connected the former, it led to a more congested circuit. Hence, we resorted to using a single buzzer with different frequencies (for each sound).

The required variables like ‘result’, ‘dis’, ‘pos’ and ‘value’ were set and sent to the IBM cloud. For the backend of the application, these variables could be manipulated through NODE-RED and its nodes.

The required number of screens, notifications, buttons, text boxes etc required for each screen were discussed and then implemented in the app accordingly. Due to the limitations imposed on the number of screens in the MIT App Inventor i.e., 10, the screens were reduced and instead, Vertical arrangements per screen were increased and made visible/invisible accordingly.

To plot the distance vs time chart, firebase was used to store values and simultaneously update the same. One implementation was to clear the whole chart after the time domain reached its limit and the other approach was to update values as we got them. We needed a continuous, moving chart which updated the values every second. We tried implementing a ThingSpeak chart using ‘WebViewer’ in the App but due to this, there was an interruption between sending data to IBM cloud and data to ThingSpeak. Hence, it was decided to use ‘Charts’ for this purpose. This was a more preferred option over Node red chart and ThingSpeak chart as we could plot both the minimum distance specified by the user and the real distance of the person from the object.

FLOWCHART



RESULT

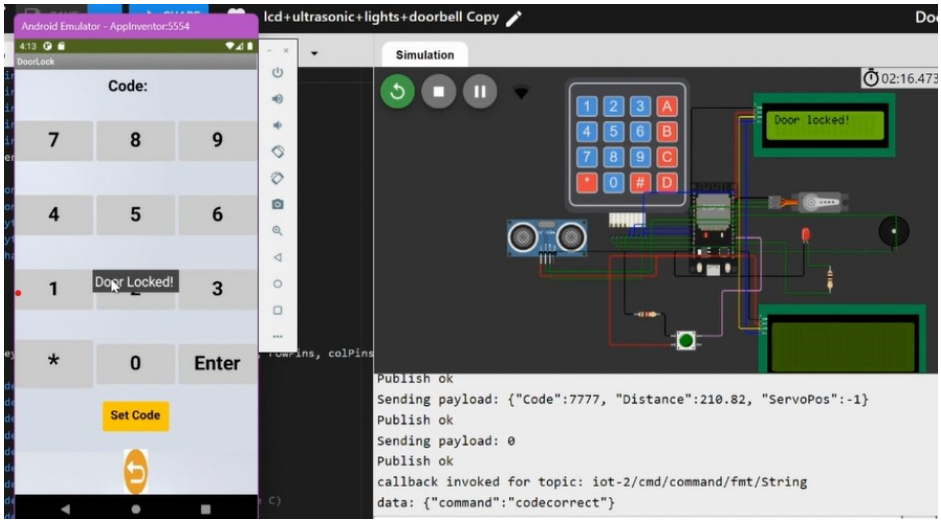


Fig. 1: Door Lock

The passcode inputted is correct, Door locks/unlocks accordingly

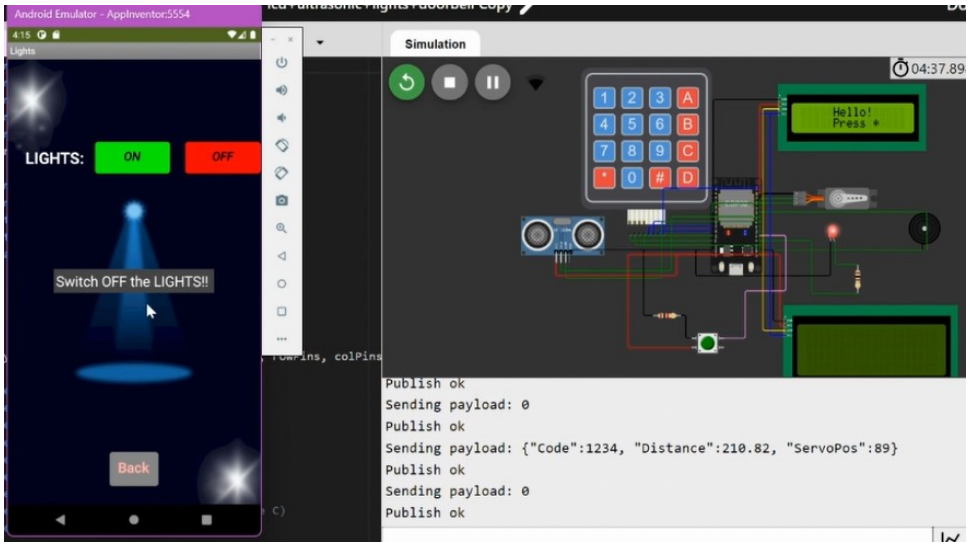


Fig. 2: Lights

The door is locked and the lights on, notification/reminder send to the user

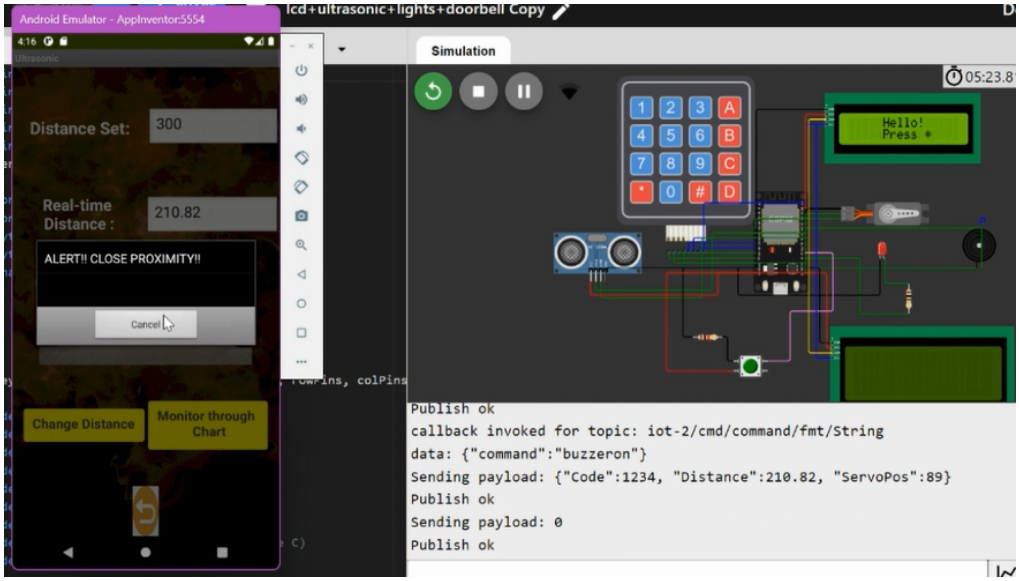


Fig. 3: Security System

Real-time distance < Minimum distance, ALERT notification

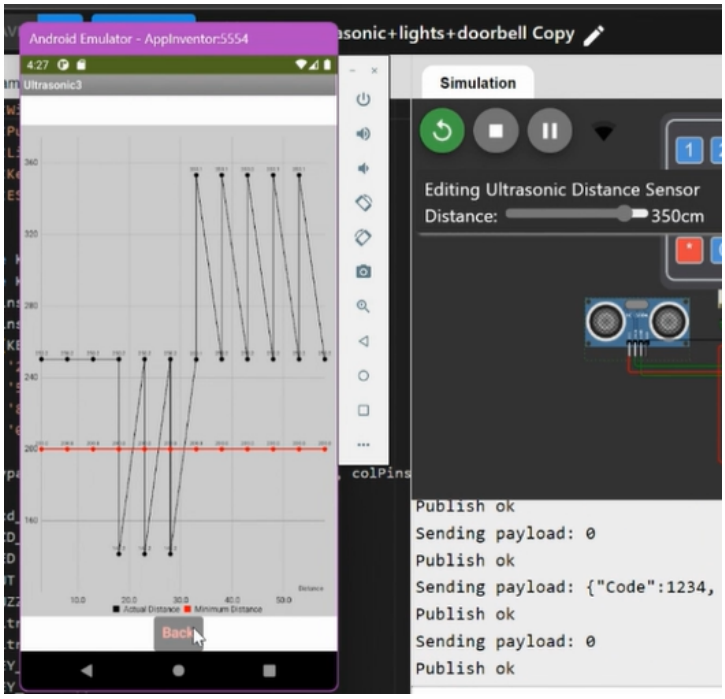


Fig. 4: DoorBell

Chart displaying ‘Actual Distance’ (black) vs ‘Minimum Distance’ (red)

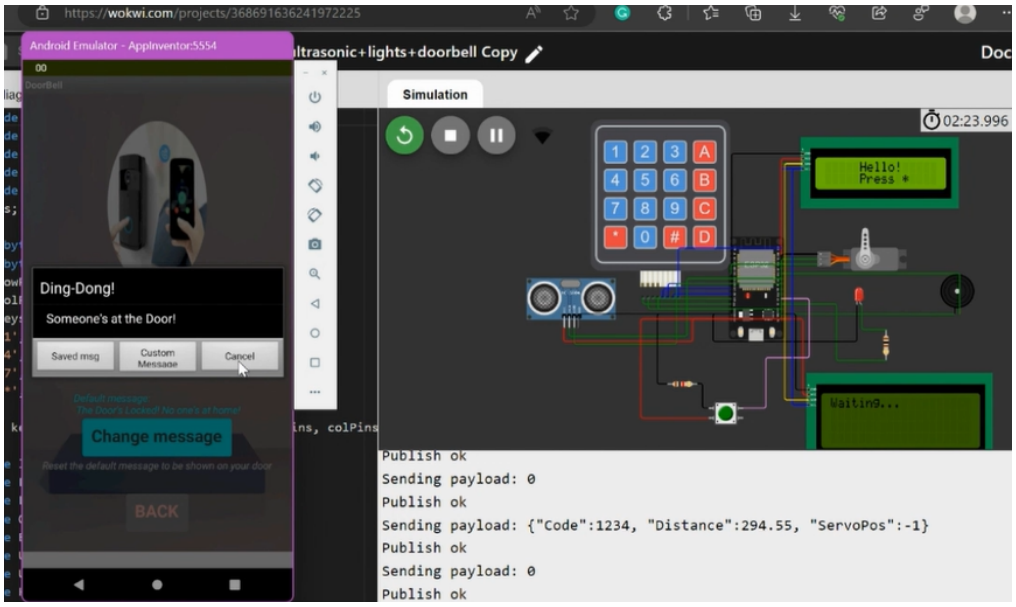


Fig. 5: DoorBell

Pushbutton pressed, LCD displays “Waiting…” until the user chooses an option as displayed

ADVANTAGES AND DISADVANTAGES

Advantages

1. **Convenience**: One of the significant advantages of smart home automation is the convenience it provides. We can control and monitor various aspects of our home remotely, with just a few taps on our smartphone.
2. **Energy Efficiency**: Smart home automation enables energy-saving features by optimizing the use of lighting, heating, cooling, and other devices. Automation rules, occupancy sensors, and scheduling capabilities help reduce energy waste, leading to lower utility bills and a more environmentally friendly home.
3. **Integration and Interoperability**: Smart home systems offer compatibility and integration with a wide range of devices and platforms. This allows for seamless communication and interoperability between different smart devices and technologies, enhancing the overall functionality and possibilities of the system.
4. **Enhanced Security:** Smart home automation systems often integrate security features such as door/window sensors, and smart locks. Users can receive real-time alerts, monitor their homes remotely, and take actions to enhance security, providing increased peace of mind.
5. **Customization and Personalization**: Smart home automation allows users to tailor their home environment to suit their needs, such as setting up lighting scenes for different moods or automating specific actions based on specific triggers or events.

Disadvantages

1. **Cost**: Smart devices and systems can be more expensive than their traditional counterparts, and the costs can add up, especially when outfitting an entire home with automation capabilities.
2. **Complexity and Technical Knowledge**: Setting up and configuring a smart home automation system can be complex, especially for users who are not familiar with technology or have limited technical knowledge. Some technical expertise may be required for installation, troubleshooting, and integrating different devices and platforms.
3. **Reliability and Dependence on Technology**: Smart home automation systems rely on technology, and if there are any connectivity issues, power outages, or technical malfunctions, the system's reliability may be compromised. It's essential to have backup plans or redundant systems in place to mitigate potential disruptions.
4. **Privacy and Security Risks**: Connected devices and systems can introduce privacy and security risks. Smart home automation systems collect and transmit data, which may raise concerns about data privacy and potential vulnerabilities if the system is not adequately secured. It's crucial to implement strong security measures, including encryption and secure network configurations, to mitigate these risks.
5. **Compatibility and Fragmentation**: With numerous brands, communication protocols, and platforms available, achieving seamless integration and compatibility between different devices and systems can be challenging. Fragmentation in the smart home market can lead to limited interoperability, requiring users to research and ensure compatibility when adding new devices to an existing setup.

APPLICATIONS

Appliance Control: Homeowners can turn devices on or off, set schedules. They can send messages to visitors if not at home and even unlock the door if relatives or guests arrive untimely.
Environmental Monitoring: Smart home automation systems can include sensors that monitor environmental conditions such as motion sensing, temperature, humidity, and air quality. Homeowners can receive real-time information and take action to maintain a comfortable and healthy indoor environment.
Security and Surveillance: Homeowners can monitor their homes in real time, receive alerts for unauthorized access, and remotely control access to their property. Integration with alarm systems adds an extra layer of security.

CONCLUSION

By bringing convenience, energy efficiency, improved security, and individualized comfort, smart home automation has completely changed how we interact with our living spaces.

Users can create unique smartphone applications for managing and controlling smart home devices using MIT App Inventor. Users with varying degrees of coding expertise can use the user-friendly drag-and-drop interface to create user-friendly interfaces.

By offering real-time data synchronisation, authentication, and hosting services, Firebase, a comprehensive mobile and online development platform, improves the smart home system. It makes it easier to manage users, store data securely, and integrate with other Firebase services, resulting in a stable infrastructure.

Hence, using tools and platforms like Wokwi, IBM Watson IoT, Node-RED, MIT App Inventor, and Firebase, it's possible to build an extensive network of interconnected smart homes.
To ensure a successful and secure implementation of the smart home automation system, it is essential to carefully design and take into account compatibility, privacy, and security measures.

FUTURE SCOPE

Integration of Artificial Intelligence (AI) and Machine Learning: Machine learning algorithms can analyze data collected from sensors, user behaviour patterns, and external factors to make intelligent predictions and automate tasks more efficiently. AI-powered virtual assistants can provide personalized recommendations, optimize energy usage, and enhance user experiences.

Enhanced Energy Management and Sustainability: Integration with renewable energy sources, such as solar panels and energy storage systems, will also become more seamless.
Voice and Gesture Control Advancements: Voice control has already gained significant popularity in home automation, but future advancements will make it even more accurate, context-aware, and natural. Gesture recognition technologies will also play a larger role, allowing users to control devices and systems through intuitive hand movements and gestures.
Smart Security Systems: Integrating advanced technologies such as facial recognition, biometrics, and machine vision will enhance security. AI algorithms will analyze video feeds, detect anomalies, and provide real-time alerts for potential security breaches. Integration with external security systems and emergency services will further enhance home security.
Interoperability and Standardization: This is to enable seamless integration between different devices, platforms, and ecosystems. Common protocols, open APIs, and industry alliances will facilitate the development of comprehensive and interconnected smart home solutions.
Environmental Monitoring: Smart home automation systems can include sensors that monitor environmental conditions such as temperature, humidity, and air quality. Homeowners can receive real-time information and take action to maintain a comfortable and healthy indoor environment.

BIBLIOGRAPHY

References:

[1] Romero, M., Guédria, W., Panetto, H., & Barafort, B. (2020). Towards a characterisation of smart systems: A systematic literature review. Computers in industry, 120, 103224.
[2] Al-Kuwari, M., Ramadan, A., Ismael, Y., Al-Sughair, L., Gastli, A., & Benammar, M. (2018, April). Smart-home automation using IoT-based sensing and monitoring platform. In *2018 IEEE 12th International Conference on Compatibility, Power Electronics and Power Engineering (CPE-POWERENG 2018)* (pp. 1-6). IEEE.
[3] Enerst, E., Udoka, E. V. H., Musiimenta, I., & Wantimba, J. (2023). Design and Implementation of a Smart Surveillance Security System. *IDOSR Journal of Science and Technology*, *9*(1), 98-106.
[4] Aminu, M., Yerima, A., Salisu, A., & Adamu, A. (2023). Design and Implementation of an IOT Based Smart Home Monitoring and Control System Using NodeMCU. *Journal of Engineering Research and Reports*, *25*(2), 78-88.
[5] Anshio, S. V., Abinesh, K., & Jayanth, S. (2023, April). A Web of Things-Specific IFTTT Applet: Integration of a Trigger Service with Action Services. In *2023 7th International Conference on Trends in Electronics and Informatics (ICOEI)* (pp. 492-496). IEEE.
[6] Cobo, M., Žiga, A., & Cabaravdic, M. (2023, May). Construction of an Automated Door as a Smart Device. In *International Conference "New Technologies, Development and Applications"* (pp. 213-220). Cham: Springer Nature Switzerland.

Appendix

(A) Source Code:

```
#include <WiFi.h>          //library for wifi
#include <PubSubClient.h> //library for MQtt
#include <LiquidCrystal_I2C.h>
#include <Keypad.h>
#include <ESP32Servo.h>
Servo s;
const byte KEYPAD_ROWS = 4;
const byte KEYPAD_COLS = 4;
byte rowPins[KEYPAD_ROWS] = {12, 13, 14, 15};
byte colPins[KEYPAD_COLS] = {26, 25, 27, 23};
char keys[KEYPAD_ROWS][KEYPAD_COLS] = {
    {'1', '2', '3', 'A'},
    {'4', '5', '6', 'B'},
    {'7', '8', '9', 'C'},
    {'*', '0', '#', 'D'}};
Keypad keypad = Keypad(makeKeymap(keys), rowPins,
colPins, KEYPAD_ROWS, KEYPAD_COLS);

#define lcd_ADDR 0x27
#define LCD_ADDR 0x26
#define LED 2
#define OUT 4
#define BUZZER_PIN 19
#define UltraTrig 18
#define UltraEcho 5
#define KEY_C 262 // 261.6256 Hz (middle C)
#define KEY_E 330 // 329.6276 Hz
#define ORG "bko7qo"        // IBM ORGANITION ID


#define DEVICE_TYPE "abcd" // Device type mentioned in
ibm watson IOT Platform
#define DEVICE_ID "1234"   // Device ID mentioned in
ibm watson IOT Platform
#define TOKEN "12345678"   // Token

LiquidCrystal_I2C LCD = LiquidCrystal_I2C(LCD_ADDR, 20,
4);
LiquidCrystal_I2C lcd(lcd_ADDR, 16, 2);
String result = "";
void callback(char *subscribetopic, byte *payload,
unsigned int payloadLength);
String data;
```

```
char server[] = ORG
".messaging.internetofthings.ibmcloud.com"; // Server
Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char publishTopic1[] = "iot-2/evt/Data/fmt/String";
char subscribetopic[] = "iot-2/cmd/command/fmt/String";
char authMethod[] = "use-token-auth";  //
authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":"
DEVICE_ID; // client id

WiFiClient wifiClient;
PubSubClient client(server, 1883, callback,
wifiClient);
int pos;

void setup(){
    Serial.begin(115200);
    s.attach(32);
    lcd.init();
    LCD.init();
    lcd.backlight();
    startMessage();
    pinMode(OUT, INPUT);
    pinMode(LED, OUTPUT);
    pinMode(UltraTrig, OUTPUT);
    pinMode(UltraEcho, INPUT);
    pinMode(BUZZER_PIN, OUTPUT);
    wificonnect();
    mqttconnect();
}

void startMessage(){
    lcd.clear();
    lcd.setCursor(5, 0);
    lcd.print("Hello!");
    lcd.setCursor(5, 1);
    lcd.print("Press *");
}

void lock(int pos){
    if (pos == -1){
        s.write(90);
```

```
        lcd.clear();
        lcd.print("Door locked!");
    }
    else if (pos = 89){
        s.write(0);
        lcd.clear();
        lcd.print("Door Unlocked!");
    }
}

String inputCode(){
    lcd.setCursor(5, 0);
    lcd.print("New Code");
    lcd.setCursor(5, 1);
    lcd.print("[____]");
    lcd.setCursor(6, 1);
    while (result.length() < 4){
        char key = keypad.getKey();
        if (key >= '0' && key <= '9'){
            lcd.print('*');
            result += key;
        }
    }
    lcd.clear();
    lcd.setCursor(2, 0);
    lcd.print("..........");
    delay(1000);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("New Code saved!");
    delay(1000);
    return result;
}

void actualCode(){
    String realCode = result;
    int i = 0;
    String code = "";
    lcd.clear();
    lcd.setCursor(5, 0);
    lcd.print("Enter Code:");
    lcd.setCursor(5, 1);
    lcd.print("[....]");
    lcd.setCursor(6, 1);
    while (code.length() < 4){
        char key = keypad.getKey();
        if (key >= '0' && key <= '9'){
            lcd.print('*');
            code += key;
        }
    }
    if (code == realCode){
        pos = s.read();
        lock(pos);
        startMessage();
    }
    else{
        lcd.clear();
        lcd.setCursor(4, 0);
        lcd.print("Wrong Code!");
        lcd.setCursor(3, 1);
        lcd.print("BREAK IN!!");
        delay(1000);
        startMessage();
    }
}

void loop(){
    char key = keypad.getKey();
    if (key == '*'){
        lcd.clear();
        inputCode();
        actualCode();
    }
    pos = s.read();
    digitalWrite(18, LOW);
    digitalWrite(18, HIGH);
    delayMicroseconds(10);
    digitalWrite(18, LOW);
    float dur = pulseIn(5, HIGH);
    float dis = (dur * 0.0343) / 2;
    int state = digitalRead(OUT);
    PublishData(result, dis, pos);
    PublishData(state);
    if (state == 1){
        tone(BUZZER_PIN, KEY_E, 500);
        delay(100);
        tone(BUZZER_PIN, KEY_C, 700);
        LCD.backlight();
        LCD.setCursor(0, 0);
        LCD.print("Waiting...");
    }
    delay(1000);
    LCD.noBacklight();
    if (!client.loop())
    {
        mqttconnect();
    }
```

```
}

// for doorbell
void PublishData(int value){
    mqttconnect();
    String payload = String(value);

    Serial.print("Sending payload: ");
    Serial.println(payload);
    if (client.publish(publishTopic1, (char
*)payload.c_str()))
        Serial.println("Publish ok");
    else
        Serial.println("Publish failed");
}

// for doorlock and alarm
void PublishData(String result, float dis, int pos){
    mqttconnect();
    // creating the String in in form JSon to update
the data to ibm cloud
    String payload = "{\"Code\":";
    payload += result;
    payload += ", \"Distance\":";
    payload += dis;
    payload += ", \"ServoPos\":";
    payload += pos;
    payload += "}";

    Serial.print("Sending payload: ");
    Serial.println(payload);
    if (client.publish(publishTopic, (char
*)payload.c_str()))
        Serial.println("Publish ok");
    else
        Serial.println("Publish failed");
}

void mqttconnect(){
    if (!client.connected()){
        Serial.print("Reconnecting client to ");
        Serial.println(server);
        while (!!!client.connect(clientId, authMethod,
token)){
            Serial.print(".");
            delay(500);
        }
        initManagedDevice();
        Serial.println();
    }
}

void wificonnect(){
    Serial.println();
    Serial.print("Connecting to ");
    WiFi.begin("Wokwi-GUEST", "", 6);
    while (WiFi.status() != WL_CONNECTED){
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
}

void initManagedDevice() {
    if (client.subscribe(subscribetopic)) {
        Serial.println(subscribetopic);
        Serial.println("subscribe to cmd OK");
    } else {
        Serial.println("subscribe to cmd FAILED");
    }
}

void callback(char* subscribetopic, byte* payload,
unsigned int payloadLength) {
    Serial.print("callback invoked for topic: ");
    Serial.println(subscribetopic);

    for (int i = 0; i < payloadLength; i++) {
        // Serial.print((char)payload[i]);
        data += (char)payload[i];
    }
    Serial.println("data: " + data);

    if (data == "{\"command\":\"lighton\"}") {
        Serial.println(data);
        digitalWrite(LED, HIGH);
    } else if (data == "{\"command\":\"lightoff\"}") {
        Serial.println(data);
        digitalWrite(LED, 0);
    } else if (data == "{\"command\":\"codecorrect\"}")
{
        int pos = s.read();
        lock(pos);

        startMessage();
```

```
    } else if (data.substring(12, 22) == "codechange")
{
        result = data.substring(22, 26);
    } else if (data == "{\"command\":\"buzzeron\"}") {
        tone(BUZZER_PIN, 1000);
    } else if (data == "{\"command\":\"buzzeroff\"}") {
        noTone(BUZZER_PIN);
    } else {
        String text = data;
        LCD.clear();
        LCD.backlight();
        int len = sizeof(text);
        int diff = 80 - len;
        char str = ' ';

        for (int i = 0; i < diff; i++)
            text = text + str;
```

```
        for (int line = 0; line < 4; line++) {
            LCD.setCursor(0, line);
            LCD.print(text.substring(line * 20, (line +
1) * 20));
        }
        Serial.println("data: " + text);
        delay(5000);
        LCD.clear();
        LCD.noBacklight();
        Serial.println(data);
    }
    data = "";
}
```