GALALA UNIVERSITY
Powered by
Arizona State University

Smart Home Automation System Using Arduino

Name	ID	Program
Sohila Ahmed	221101149	AIS
Marina Reda	221101235	AIS
Mohamed ELshourbagi	221100633	AIE
Ziad Abdelrahman	221101043	AIS

Abstract:

This project presents a Smart Home Automation System using Arduino, designed as an IoT project for an embedded systems course. Integrating 16 sensors, including Button Sensor, Photocell Sensor, PIR Motion Sensor, and others, this system aims to enhance home security and comfort. Control is achieved through a mobile application connected to Arduino via a Bluetooth HM-10 Module, enabling remote access and alert notifications. The project emphasizes the growing importance of IoT technologies in modern homes, aligning with the vision of a future where smart home systems are as essential as Internet access.

Introduction:

In today's rapidly evolving technological landscape, the concept of a smart home has become increasingly relevant. This project showcases a Smart Home Automation System using Arduino, chosen as an IoT project for its significance in modern industry trends. By integrating 16 sensors, this system offers homeowners the ability to monitor and control their homes remotely, enhancing both security and comfort.

As Bill Gates famously stated, "In the near future, a house without a smart home system will be as unfashionable as a home without Internet access today." This quote resonates deeply with the core philosophy of this project, highlighting the inevitable shift towards smart technologies in modern living spaces. By choosing Arduino as the platform for this project, we embrace the versatility and accessibility that it offers, making it an ideal choice for IoT applications.

The mobile application serves as the central control hub for the system, allowing users to manage home devices and receive alerts for security breaches or anomalies, such as gas leaks. Additionally, the system features a password-protected electronic lock for enhanced security, providing homeowners with peace of mind knowing they can control access to their homes remotely.

One of the key advantages of this smart home automation system is its ability to enhance home security and comfort. The intelligent logic programming ensures that home devices, such as the air conditioner, water heater, LED lights, and smart curtains, are controlled based on preset preferences, creating a comfortable and tranquil atmosphere at home.

In conclusion, this project demonstrates the transformative potential of IoT technologies in modern homes, offering homeowners a glimpse into a more connected, secure, and convenient future. By embracing the concept of a smart home, we pave the way for a more efficient and sustainable way of living.

Comprehensive Literature Survey: Smart Home Arduino Automation System with IoT Integration and Mobile App Control

1. Introduction

Smart homes are revolutionizing living spaces, prioritizing comfort, security, and energy efficiency. This project aims to develop a smart home automation system using Arduino as the central microcontroller. The system will integrate Internet of Things (IoT) technologies for remote control and data collection, making it ideal for an embedded systems course. This survey explores existing research on Arduino-based smart home systems, focusing on incorporating IoT for enhanced functionality.

2. Smart Home Automation with Arduino

Existing research demonstrates the feasibility of Arduino for smart home automation. Vijayaraghavan et al. (2022) propose a system with light and appliance control, along with security features like door/window monitoring [6]. Jain et al. (2020) explore voice-controlled automation using Arduino with speech recognition [7], highlighting the potential for user-friendly interfaces.

3. Leveraging IoT in Smart Home Systems

IoT plays a crucial role in smart homes. Park et al. (2023) discuss its ability to connect devices, collect sensor data, and enable remote control in their article titled "A Secure and Efficient IoT-based Smart Home System using Blockchain and Machine Learning" [8]. Sharma et al. (2021) emphasize the importance of communication protocols, security, and data management in IoT-based systems in their review titled "IoT-based Smart Homes: A Review" [9].

4. Bluetooth Communication for Mobile App Integration

A tutorial by predictabledesigns.com demonstrates building a Bluetooth-enabled Arduino system with a mobile application for control [4]. The official MIT App Inventor documentation offers resources for creating mobile apps that interact with Bluetooth devices [10].

5. Project Significance and Objectives

This project aligns with the growing interest in smart home automation and embedded systems development. It aims to:

- Develop a system with comprehensive sensor integration (16 sensors) for security, environment monitoring, and comfort control.
- Design a user-friendly mobile application for intuitive control and real-time data visualization using Bluetooth communication.

- Explore the feasibility of integrating the Arduino Cloud platform for remote access and data storage (depending on project scope).

6. Expanded Sensor Integration

The project incorporates a wider range of sensors to achieve a more comprehensive smart home experience:

- **Security Sensors:** PIR Motion Sensor, Door/Window Sensor
- **Environmental Sensors:** Photocell Sensor, Temperature Sensor, Humidity Sensor, Soil Humidity Sensor
- **Safety Sensors:** MQ-2 Gas Sensor, Smoke Sensor (add others as needed)

7. Benefits of a Comprehensive Smart Home System

- **Enhanced Security:** Real-time monitoring deters unauthorized entry and notifies you of potential breaches.
- **Improved Comfort:** Automated climate control, lighting adjustments, and plant watering ensure a comfortable environment.
- **Increased Convenience:** Remote control of appliances and features allows for effortless management from anywhere.
- **Energy Efficiency:** Sensor-based automation optimizes device usage, leading to reduced energy consumption.
- **Peace of Mind:** Real-time monitoring and alerts for potential hazards offer peace of mind when away from home.

8. Mobile Application Integration and Functionality

The Bluetooth-enabled mobile application serves as the central hub for user interaction and system control.

- The app will display sensor data in a user-friendly format.
- Users can control various aspects through the app:
    - Turn lights and appliances on/off remotely.
    - Adjust thermostat settings.
    - Receive alerts for security breaches, gas leaks, smoke detection, and other emergencies.
    - Program automated routines based on time, user presence, or sensor data.

9. Security Considerations in an IoT-based System

The project will prioritize robust security measures:

- **Secure Communication Protocol:** Implement Bluetooth Low Energy (BLE) to encrypt data transmission.
- **User Authentication:** Incorporate a user authentication mechanism within the mobile app.
- **Data Encryption:** Consider encrypting sensitive data for added protection.

10. Conclusion

This comprehensive literature survey highlights the project's potential to create a secure, comfortable, and user-friendly smart home environment. By leveraging a comprehensive sensor network and a feature-rich mobile application, the system offers a valuable learning experience and contributes to the growing trend of smart home automation.

Motivation

1. Convenience and Comfort:

- **Simplify your daily routine:** Imagine coming home to a house that's already at the perfect temperature, with lights adjusting based on the natural light. This project allows you to automate these tasks, freeing up your time and energy for other things.
- **Enhanced Peace of Mind:** Remotely monitor your home's environment with temperature and humidity sensors. Receive alerts for potential issues like gas leaks or security breaches, giving you peace of mind even when you're away.
- **Personalized Living:** Develop automated routines that adjust lighting, temperature, or watering schedules based on your preferences and habits. Create a truly personalized and comfortable living space.

2. Security and Safety:

- **Proactive Security Measures:** Implement a comprehensive security system with motion sensors, door/window sensors, and even smoke detection. These features deter unauthorized entry and provide early warnings of potential hazards, keeping your home and loved ones safe.
- **Remote Monitoring and Alerts:** Receive real-time notifications on your phone for any security breaches, gas leaks, or smoke detection. This allows for a quicker response time to potential emergencies.
- **Increased Awareness and Control:** Gain real-time insights into your home environment through sensor data. This empowers you to make informed decisions about your home's security and safety.
-

3. Learning and Exploration:

- **Hands-on Experience with Embedded Systems:** This project provides a practical platform to learn and apply concepts of embedded systems development. You'll gain experience with Arduino programming, sensor integration, and communication protocols.
- **Understanding IoT Applications:** Building a smart home system allows you to explore the practical applications of IoT technology. You'll learn how to connect devices, collect data, and create a system that interacts with the physical world.
- **Creative Problem-solving:** This project encourages you to think creatively about how technology can improve daily life. You'll experiment, solve challenges, and develop innovative solutions using Arduino and IoT.

Problem Statement

The Need for a Secure, User-Friendly, and Comprehensive Smart Home Automation System

Modern homes are increasingly packed with electronic devices, offering convenience and functionality. However, managing these devices individually can be time-consuming and inefficient. Additionally, traditional home automation systems can be expensive, complex to install, and lack user-friendly interfaces. Furthermore, security concerns regarding data privacy and unauthorized access are a growing issue in the interconnected world.

This project aims to address these limitations by developing a smart home automation system using Arduino as the central microcontroller. This cost-effective and accessible platform allows for customization and experimentation, making it ideal for personal use and educational purposes. By integrating Internet of Things (IoT) technologies, the system will overcome the limitations of traditional setups, enabling:

- **Remote control and monitoring:** Users can manage appliances, lighting, and environmental conditions from anywhere with an internet connection, increasing convenience and flexibility.
- **Enhanced security:** Integration of security sensors and real-time alerts allows for proactive measures against unauthorized entry and potential hazards like gas leaks or smoke detection.
- **Improved energy efficiency:** Sensor-based automation optimizes device usage, leading to reduced energy consumption and cost savings.
- **Increased comfort and automation:** Automated routines can be programmed for lighting, temperature control, or plant watering based on user preferences and sensor data, creating a personalized and comfortable living environment.

However, the challenge lies in developing a system that is not only functional but also user-friendly and secure. The mobile application interface needs to be intuitive and accessible for users with varying technical backgrounds. Robust security measures must be implemented to protect user data and prevent unauthorized access to the system.

This project seeks to bridge this gap by creating a comprehensive smart home automation system that is:

- **User-friendly:** The mobile application will prioritize a user-friendly interface with clear visualizations of sensor data and intuitive controls.
- **Secure:** Secure communication protocols and user authentication mechanisms will ensure data privacy and prevent unauthorized access.
- **Comprehensive:** The system will integrate a wide range of sensors, allowing for control over various aspects of the home environment, from security and safety to comfort and convenience

Proposed Work

Secure and User-Friendly Smart Home Automation System with Arduino and IoT

This project proposes the development of a comprehensive smart home automation system utilizing Arduino and Internet of Things (IoT) technologies. The system will prioritize user-friendliness, security, and comprehensive functionality, aiming to address limitations present in traditional smart home setups.

1. Hardware Design and System Architecture:

- The core of the system will be an Arduino microcontroller, chosen for its affordability, ease of use, and extensive community support.
- A selection of 16 sensors will be integrated, including:
    - Security sensors (PIR motion sensor, door/window contact sensor) to monitor unauthorized entry.
    - Environmental sensors (photocell sensor, temperature sensor, humidity sensor, soil humidity sensor) for automated climate control and plant care.
    - Safety sensors (MQ-2 gas sensor, smoke sensor) to detect potential hazards and trigger alarms.
    - Additional sensors (optional) can be incorporated based on specific needs.
- A Bluetooth Low Energy (BLE) module will facilitate secure and efficient communication between the Arduino and the mobile application.
- Relay modules will be used to control appliances and lights based on user input or sensor data.
- An LCD display (optional) can be integrated for local control and sensor data visualization.
- The system will be designed with modularity in mind, allowing for future expansion and customization.

2.Software Development and Integration with Existing Mobile Application

This project will leverage an existing mobile application designed to interact with IoT systems. We will focus on integrating this application with the environment of our smart home system. This may involve:

API Integration: The Arduino code will be developed to communicate with the application's API (Application Programming Interface) for data exchange and control functions.

Data Mapping: Sensor data acquired by the Arduino will be formatted and transmitted to the application in a way that aligns with its existing functionality.

User Interface Customization: We will explore potential modifications to the existing mobile application's interface to ensure optimal visualization of sensor data and user interaction with our specific smart home setup.

Additionally:

The Arduino programming environment will still be used to develop control logic for sensor data acquisition, device interaction, and automated routines within the system itself.

The functionalities listed previously (real-time data visualization, remote control, automated routines, security alerts, user authentication) will still be targeted, ensuring seamless integration with the existing application.

This approach leverages the existing application's capabilities while allowing us to customize its interaction with our unique sensor setup and functionalities.

3. Security Considerations and Data Management:

- Secure communication protocols like BLE will be implemented to encrypt data transmission between the Arduino and the mobile application.
- User authentication mechanisms will be incorporated within the mobile app to restrict access and prevent unauthorized control.
- Data encryption will be considered for sensitive data like temperature readings or user information, protecting it from potential breaches.

4. Testing and Evaluation:

- The system will undergo comprehensive testing to ensure proper functionality of all components and communication protocols.
- User testing will be conducted to evaluate the usability and user-friendliness of the mobile application interface.
- The security measures of the system will be rigorously tested to identify and address any vulnerabilities.

5. Expected Outcomes and Future Work:

- This project aims to deliver a secure, user-friendly, and comprehensive smart home automation system that is easily manageable from a mobile application.
- The project seeks to demonstrate the capabilities of Arduino and IoT technologies in creating a practical and cost-effective smart home solution.
- Future work could involve exploring integration with cloud platforms like Arduino Cloud for remote access and data storage.
- Additional functionalities like voice control or integration with smart home ecosystems can be implemented for further development.

Methodology

Developing a Secure and User-Friendly Smart Home System

This section outlines the methodology for developing a secure and user-friendly smart home automation system utilizing Arduino and existing mobile application integration.

1. System Design and Hardware Selection:

- **1.1 System Requirements Analysis:**
  - Define the specific functionalities and desired user experience for the smart home system.
  - Identify the sensors needed to achieve these functionalities (temperature, humidity, security, etc.) based on your existing application's capabilities.
  - Consider potential future expansion and sensor integration.
- **1.2 Hardware Selection:**
  - Select the appropriate Arduino board based on the number of sensors, processing power requirements, and available communication protocols.
  - Choose suitable sensors for temperature, humidity, security, safety (gas, smoke), and any additional features desired.
  - Ensure compatibility between sensors and the chosen Arduino board.
  - Select a Bluetooth Low Energy (BLE) module for secure and efficient communication with the mobile application.
  - Choose relay modules for controlling appliances and lights based on user input or sensor data.
  - Consider incorporating an LCD display (optional) for local control and sensor data visualization.

2. Software Development:

- **2.1 Arduino Programming:**
  - Develop code for the Arduino to:
    - Read sensor data from all connected sensors.
    - Process sensor data and implement control logic for automated routines (e.g., lighting based on photocell sensor).
    - Communicate with the mobile application through the BLE module for data exchange and control commands.
    - Implement security measures to prevent unauthorized access (optional - password protection for Arduino settings).
- **2.2 Mobile Application Integration:**
  - Analyze the existing mobile application's functionalities and API (Application Programming Interface) for data exchange.
  - Develop code to connect the Arduino with the mobile application's API.
  - Format sensor data from the Arduino for compatibility with the application's data structure.
  - Design user interface elements within the existing application for:
    - Real-time sensor data visualization (temperature, humidity, etc.).

        - Remote control of appliances, lights, and other devices.
        - Configuration of automated routines based on user preferences or sensor data.
        - Security alerts for potential breaches, gas leaks, smoke detection, and other emergencies.
        - User authentication mechanism (if not already implemented in the existing application).

3. Testing and Evaluation:

- **3.1 Unit Testing:**
    - Individually test each hardware component (sensors, Arduino board, BLE module) for proper functionality.
    - Unit test Arduino code for sensor data acquisition, control logic, and communication with the BLE module.
- **3.2 System Integration Testing:**
    - Test the complete system by integrating all hardware components and the mobile application.
    - Verify communication between the Arduino and the mobile application through the BLE module.
    - Ensure sensor data is accurately displayed and functionalities like remote control and automated routines work as intended.
- **3.3 User Testing:**
    - Conduct user testing with volunteers to assess the usability and user-friendliness of the mobile application interface.
    - Gather feedback on the overall user experience for controlling and monitoring the smart home system.

4. Security Considerations:

- Implement secure communication protocols like BLE to encrypt data transmission between the Arduino and the mobile application.
- If the existing mobile application lacks user authentication, incorporate a password or pin code login within the application to restrict unauthorized access.
- Consider data encryption for sensitive information, especially if user data is stored within the application.

5. Documentation and Refinement:

- Document the entire development process, including hardware selection, software code, and testing procedures.
- Based on testing and user feedback, refine the system by:
    - Modifying Arduino code for improved functionality or addressing bugs.
    - Making adjustments to the mobile application interface for enhanced usability.
    - Implementing additional security measures if necessary.

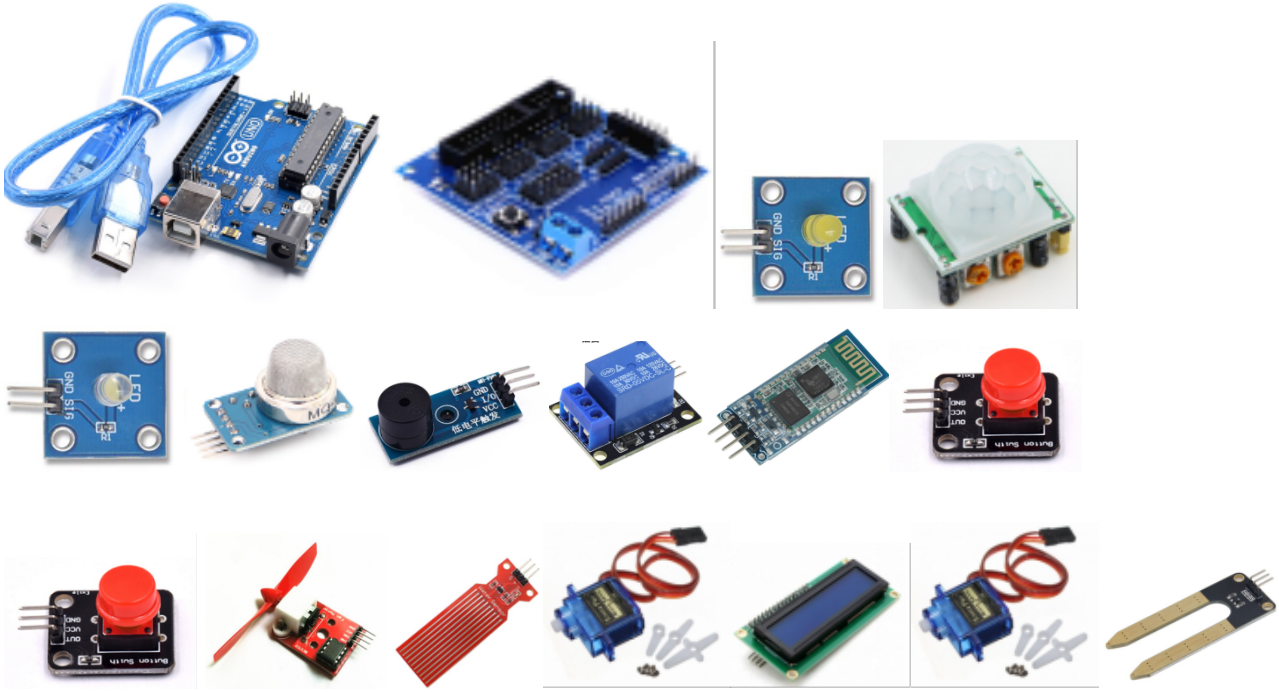
System Description

Secure and User-Friendly Smart Home Automation System with Arduino and Mobile App Integration

This document describes a secure and user-friendly smart home automation system designed to enhance comfort, security, and energy efficiency within your living environment. The system leverages an existing mobile application for user interaction and integrates with Arduino as the central microcontroller for sensor data acquisition and device control.

1. System Components:

- ## Hardware:



Arduinio board _Arduino shield _Yellow light_PIR motion sensor _White led _

MQ-2 Gas sesonr _passive buzzer _Relay _HM-10 Bluethooth_ 2 Buttons_ FAN

_Steam Sensor _ 2 Servo motors _ LCD 1602 dispaly module

Arduino board: The brain of the system, reading sensor data, controlling actuators, and communicating with a mobile app.

White LED (Digital pin 13): Provides visual indication for events or states (e.g., blinking LED for motion detection).

PIR Motion Sensor (Digital pin 2): Detects motion within its field of view, triggering actions like turning on lights or playing sounds.

Yellow LED (Digital pin 5, controlled by PWM): Allows dimming or adjusting light intensity based on program control.

MQ-2 Gas Sensor (Analog pin A0): Detects presence of gases, potentially triggering alarms or ventilation systems.

Passive Buzzer (Digital pin 3): Generates tones for alarms, notifications, or melodies.

Relay (Digital pin 12): Controls power to external devices like lamps or appliances.

HM-10 Bluetooth Module (Serial Communication): Enables communication between the Arduino and a mobile app for remote control.

2 Buttons (Digital pins 4 & 8): Allow user input for actions like triggering functions or entering passwords.

Fan (Digital pins 6 & 7): Controls fan speed using PWM for adjusting airflow.

2 Servo Motors (Digital pins 9 & 10): Can be positioned to specific angles, potentially for controlling blinds, robotic arms, or other movable objects.

LCD 1602 Display Module (not directly controlled in provided code snippet): Likely used to show sensor readings, status messages, or user interface elements.

Steam sensor: The principle is to detect the amount of water through the exposed printed parallel lines on the circuit board.

- # Software

Code developed within the Arduino Integrated Development Environment (IDE) :

The software for this project utilizes an Arduino sketch to control various components of a smart home system. The code includes libraries for servo motors, I2C communication for an LCD display, and basic input/output operations. Here's a breakdown of the key components and functionalities:

1. Servo Motor Control: Two servo motors are connected to digital pins 9 and 10. These servos can be controlled to specific angles based on input commands.

2. LCD Display: An LCD display is connected via I2C communication to display information. The address of the display is set to 0x27, with 16 characters per line and 2 lines in total.

3. Sensor Readings: The code reads values from various sensors, including gas, light, water, and soil moisture sensors. These values are used to detect conditions such as gas leaks, low light levels, rain, and soil dryness.

4. User Interface: The system includes buttons for user input. Pressing these buttons triggers actions such as playing music, controlling the servos, and displaying information on the LCD.

5. Alarm and Feedback: The system provides feedback through sound alerts using a passive buzzer. For example, it can sound an alarm in case of a gas leak or low soil moisture.

6. Security Features: The code includes a password-protected door lock system. Users can input a password using buttons, and if the correct password is entered, a servo unlocks the door.

7. Remote Control: The system can be controlled remotely via serial communication. Commands sent over serial can control the servos, LED brightness, and fan speed.

8. Music Playback: The code includes functions to play two songs, "Happy Birthday" and "Ode to Joy," using the passive buzzer.

9. PWM Control: Pulse Width Modulation (PWM) is used to control the brightness of an LED and the speed of a fan.

10. Error Handling: The system provides error messages on the LCD display in case of incorrect input or other issues.

Overall, the software provides a comprehensive control system for a smart home, integrating various sensors, actuators, and user interface elements to create a versatile and interactive environment.

Code

```
//call the relevant library file
#include <Servo.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
//Set the communication address of I2C to 0x27, display 16
characters every line, two lines in total
LiquidCrystal_I2C mylcd(0x27, 16, 2);

//set ports of two servos to digital 9 and 10
Servo servo_10;
Servo servo_9;

volatile int btn1_num;//set variable btn1_num
volatile int btn2_num;//set variable btn2_num
volatile int button1;//set variable button1
volatile int button2;//set variable button2
String fans_char;//string type variable fans_char
```

```
volatile int fans_val;//set variable fans_char
volatile int flag;//set variable flag
volatile int flag2;//set variable flag2
volatile int flag3;//set variable flag3
volatile int gas;//set variable gas
volatile int infrar;//set variable infrar
String led2;//string type variable led2
volatile int light;//set variable light
String pass;//string type variable pass
String passwd;//string type variable passwd

String servo1;//string type variable servo1
volatile int servo1_angle;//set variable light
String servo2;//string type variable servo2
volatile int servo2_angle;//set variable servo2_angle

volatile int soil;//set variable soil
volatile int val;//set variable val
volatile int value_led2;//set variable value_led2
volatile int water;//set variable water

int length;
int tonepin = 3; //set the signal end of passive buzzer to digital
3
//define name of every sound frequency
#define D0 -1
#define D1 262
#define D2 293
#define D3 329
#define D4 349
#define D5 392
#define D6 440
#define D7 494
#define M1 523
#define M2 586
#define M3 658
#define M4 697
#define M5 783
#define M6 879
#define M7 987
#define H1 1045
#define H2 1171
#define H3 1316
#define H4 1393
#define H5 1563
#define H6 1755
#define H7 1971

#define WHOLE 1
#define HALF 0.5
#define QUARTER 0.25
#define EIGHTH 0.25
```

```
#define SIXTEENTH 0.625

//set sound play frequency
int tune[] =
{
  M3, M3, M4, M5,
  M5, M4, M3, M2,
  M1, M1, M2, M3,
  M3, M2, M2,
  M3, M3, M4, M5,
  M5, M4, M3, M2,
  M1, M1, M2, M3,
  M2, M1, M1,
  M2, M2, M3, M1,
  M2, M3, M4, M3, M1,
  M2, M3, M4, M3, M2,
  M1, M2, D5, D0,
  M3, M3, M4, M5,
  M5, M4, M3, M4, M2,
  M1, M1, M2, M3,
  M2, M1, M1
};

//set music beat
float durt[] =
{
  1, 1, 1, 1,
  1, 1, 1, 1,
  1, 1, 1, 1,
  1 + 0.5, 0.5, 1 + 1,
  1, 1, 1, 1,
  1, 1, 1, 1,
  1, 1, 1, 1,
  1 + 0.5, 0.5, 1 + 1,
  1, 1, 1, 1,
  1, 0.5, 0.5, 1, 1,
  1, 0.5, 0.5, 1, 1,
  1, 1, 1, 1,
  1, 1, 1, 1,
  1, 1, 1, 0.5, 0.5,
  1, 1, 1, 1,
  1 + 0.5, 0.5, 1 + 1,
};


void setup() {
  Serial.begin(9600);//set baud rate to 9600

  mylcd.init();
  mylcd.backlight();//initialize LCD
  //LCD shows "passcord:" at first row and column
  mylcd.setCursor(1 - 1, 1 - 1);
```

```
  mylcd.print("passcord:");

  servo_9.attach(9);//make servo connect to digital 9
  servo_10.attach(10);//make servo connect to digital 10
  servo_9.write(0);//set servo connected digital 9 to 0°
  servo_10.write(0);//set servo connected digital 10 to 0°
  delay(300);

  pinMode(7, OUTPUT);//set digital 7 to output
  pinMode(6, OUTPUT);//set digital 6 to output
  digitalWrite(7, HIGH); //set digital 7 to high level
  digitalWrite(6, HIGH); //set digital 6 to high level

  pinMode(4, INPUT);//set digital 4 to input
  pinMode(8, INPUT);//set digital 8 to input
  pinMode(2, INPUT);//set digital 2 to input
  pinMode(3, OUTPUT);//set digital 3 to output
  pinMode(A0, INPUT);//set A0 to input
  pinMode(A1, INPUT);//set A1 to input
  pinMode(13, OUTPUT);//set digital 13 to input
  pinMode(A3, INPUT);//set A3 to input
  pinMode(A2, INPUT);//set A2 to input

  pinMode(12, OUTPUT);//set digital 12 to output
  pinMode(5, OUTPUT);//set digital 5 to output
  pinMode(3, OUTPUT);//set digital 3 to output
  length = sizeof(tune) / sizeof(tune[0]); //set the value of
length
}

void loop() {
  auto_sensor();
  if (Serial.available() > 0) //serial reads the characters
  {
    val = Serial.read();//set val to character read by serial
Serial.println(val);//output val character in new lines
    pwm_control();
  }
  switch (val) {
    case 'a'://if val is character 'a', program will circulate
      digitalWrite(13, HIGH); //set digital 13 to high level, LED
lights up
      break;//exit loop
    case 'b'://if val is character 'b', program will circulate
      digitalWrite(13, LOW); //Set digital 13 to low level, LED is
off
      break;//exit loop
    case 'c'://if val is character 'c', program will circulate
      digitalWrite(12, HIGH); //set digital 12 to high level, NO of
relay is connected to COM
      break;//exit loop
```

```
    case 'd'://if val is character 'd', program will circulate
      digitalWrite(12, LOW); //set digital 12 to low level, NO of
relay is disconnected to COM

      break;//exit loop
    case 'e'://if val is character 'e', program will circulate
      music1();//play birthday song
      break;//exit loop
    case 'f'://if val is character 'f', program will circulate
      music2();//play ode to joy song
      break;//exit loop
    case 'g'://if val is character 'g', program will circulate
      noTone(3);//set digital 3 to stop playing music
      break;//exit loop
    case 'h'://if val is character 'h', program will circulate
      Serial.println(light);//output the value of variable light
in new lines
      delay(100);
      break;//exit loop
    case 'i'://if val is character 'i', program will circulate
      Serial.println(gas);//output the value of variable gas in
new lines
      delay(100);
      break;//exit loop
    case 'j'://if val is character 'j', program will circulate
      Serial.println(soil);//output the value of variable soil in
new lines
      delay(100);
      break;//exit loop
    case 'k'://if val is character 'k', program will circulate
      Serial.println(water);//output the value of variable water
in new lines
      delay(100);
      break;//exit loop
    case 'l'://if val is character 'l', program will circulate
      servo_9.write(180);//set servo connected to digital 9 to
180°
      delay(500);
      break;//exit loop
    case 'm'://if val is character 'm', program will circulate
      servo_9.write(0);;//set servo connected to digital 9 to 0°
      delay(500);
      break;//exit loop
    case 'n'://if val is character 'n', program will circulate
      servo_10.write(180);//set servo connected to digital 10 to
180°
      delay(500);
      break;//exit loop
    case 'o'://if val is character 'o', program will circulate
```

```
      servo_10.write(0);//set servo connected to digital 10 to 0°
      delay(500);
      break;//exit loop
    case 'p'://if val is character 'p', program will circulate
      digitalWrite(5, HIGH); //set digital 5 to high level, LED is
on
      break;//exit loop
    case 'q'://if val is character 'q', program will circulate
      digitalWrite(5, LOW); // set digital 5 to low level, LED is
off
      break;//exit loop
    case 'r'://if val is character 'r', program will circulate
      digitalWrite(7, LOW);
      digitalWrite(6, HIGH); //fan rotates anticlockwise at the
fastest speed
      break;//exit loop
    case 's'://if val is character 's', program will circulate
      digitalWrite(7, LOW);
      digitalWrite(6, LOW); //fan stops rotating
      break;//exit loop
  }
}

//////////////////////set birthday
song//////////////////////////////////
void birthday()
{
  tone(3, 294); //digital 3 outputs 294HZ sound
  delay(250);//delay in 250ms
  tone(3, 440);
  delay(250);
  tone(3, 392);
  delay(250);
  tone(3, 532);
  delay(250);
  tone(3, 494);
  delay(500);
  tone(3, 392);
  delay(250);
  tone(3, 440);
  delay(250);
  tone(3, 392);
  delay(250);
  tone(3, 587);
  delay(250);
  tone(3, 532);
  delay(500);
  tone(3, 392);
  delay(250);
  tone(3, 784);
  delay(250);
```

```
  tone(3, 659);
  delay(250);
  tone(3, 532);
  delay(250);
  tone(3, 494);
  delay(250);
  tone(3, 440);
  delay(250);
  tone(3, 698);
  delay(375);
  tone(3, 659);
  delay(250);
  tone(3, 532);
  delay(250);
  tone(3, 587);
  delay(250);
  tone(3, 532);
  delay(500);
}

//detect gas
void auto_sensor() {
  gas = analogRead(A0);//assign the analog value of A0 to gas
  if (gas > 700) {
//if variable gas>700
    flag = 1;//set variable flag to 1
    while (flag == 1)
      //if flag is 1, program will circulate
    {
      Serial.println("danger");//output "danger" in new lines
      tone(3, 440);
      delay(125);
      delay(100);
      noTone(3);
      delay(100);
      tone(3, 440);
      delay(125);
      delay(100);
      noTone(3);
      delay(300);
      gas = analogRead(A0);//gas analog the value of A0 to gas
      if (gas < 100)  //if variable gas is less than 100
      {
        flag = 0;//set variable flag to 0
        break;//exit loop exist to loop
      }
    }

  } else
    //otherwise
  {
    noTone(3);// digital 3 stops playing music
```

```
  }
  light = analogRead(A1);////Assign the analog value of A1 to
light
  if (light < 300)//if variable light is less than 300
  {
    infrar = digitalRead(2);//assign the value of digital 2 to
infrar
    Serial.println(infrar);//output the value of variable infrar
in new lines
    if (infrar == 1)
      // if variable infra is 1
    {
      digitalWrite(13, HIGH); //set digital 13 to high level, LED
is on
    } else//Otherwise
    {
      digitalWrite(13, LOW); //set digital 13 to low level, LED is
off
    }
```

```
  }
  water = analogRead(A3);//assign the analog value of A3 to
variable water
  if (water > 800)
    // if variable water is larger than 800
  {
    flag2 = 1;//if variable flag 2 to 1
    while (flag2 == 1)
      // if flag2 is 1, program will circulate
    {
      Serial.println("rain");//output "rain" in new lines
      servo_10.write(180);// set the servo connected to digital 10
to 180°
      delay(300);//delay in 300ms
      delay(100);
      water = analogRead(A3);;//assign the analog value of A3 to
variable water
      if (water < 30)// if variable water is less than 30
      {
        flag2 = 0;// set flag2 to 0
        break;//exit loop
      }
    }
```

```
  } else//Otherwise
  {
    if (val!= 'u' && val!= 'n')
      //if val is not equivalent 'u' either 'n'
    {
      servo_10.write(0);//set servo connected to digital 10 to 0°
      delay(10);
```

```
    }

  }
  soil = analogRead(A2);//assign the analog value of A2 to
variable soil
  if (soil > 50)
    // if variable soil is greater than 50
  {
    flag3 = 1;//set flag3 to 1
    while (flag3 == 1)
      //If set flag3 to 1, program will circulate
    {
      Serial.println("hydropenia ");//output "hydropenia " in new
lines
      tone(3, 440);
      delay(125);
      delay(100);
      noTone(3);
      delay(100);
      tone(3, 440);
      delay(125);
      delay(100);
      noTone(3);//digital 3 stops playing sound
      delay(300);
      soil = analogRead(A2);//Assign the analog value of A2 to
variable soil
      if (soil < 10)//If variable soil<10
      {
        flag3 = 0;//set flag3 to 0
        break;//exit loop
      }
    }
  } else//Otherwise
  {
    noTone(3);//set digital 3 to stop playing music
  }
  door();//run subroutine
}
void door() {
  button1 = digitalRead(4);// assign the value of digital 4 to
button1
  button2 = digitalRead(8);//assign the value of digital 8 to
button2
  if (button1 == 0)//if variablebutton1 is 0
  {
    delay(10);//delay in 10ms
    while (button1 == 0) //if variablebutton1 is 0, program will
circulate
    {
      button1 = digitalRead(4);// assign the value of digital 4 to
button1
```

```
      btn1_num = btn1_num + 1;//variable btn1_num plus 1
      delay(100);// delay in 100ms
    }
  }
  if (btn1_num >= 1 && btn1_num < 5) //1≤if variablebtn1_num<5
  {
    Serial.print(".");
    Serial.print("");
    passwd = String(passwd) + String(".");//set passwd
pass = String(pass) + String(".");//set pass
    //LCD shows pass at the first row and column
    mylcd.setCursor(1 - 1, 2 - 1);
    mylcd.print(pass);
  }
  if (btn1_num >= 5)
    //if variablebtn1_num ≥5
  {
    Serial.print("-");
    passwd = String(passwd) + String("-");//Set passwd
    pass = String(pass) + String("-");//set pass
    //LCD shows pass at the first row and column
    mylcd.setCursor(1 - 1, 2 - 1);
    mylcd.print(pass);
  }
  if (button2 == 0) //if variablebutton2 is 0
  {
    delay(10);
    if (button2 == 0)//if variablebutton2 is 0
    {
      if (passwd == ".--.-.")//if passwd is ".--.-."
      {
        mylcd.clear();//clear LCD screen
        //LCD shows "open!" at first character on second row
        mylcd.setCursor(1 - 1, 2 - 1);
        mylcd.print("open!");
        servo_9.write(100);//set servo connected to digital 9 to
100°
        delay(300);
        delay(5000);
        passwd = "";
        pass = "";
        mylcd.clear();//clear LCD screen
        //LCD shows "password:"at first character on first row
        mylcd.setCursor(1 - 1, 1 - 1);
        mylcd.print("password:");
      } else //Otherwise
      {
        mylcd.clear();//clear LCD screen
        //LCD shows "error!"at first character on first row
        mylcd.setCursor(1 - 1, 1 - 1);
        mylcd.print("error!");
        passwd = "";
```

```
        pass = "";
        delay(2000);
        //LCD shows "again" at first character on first row
        mylcd.setCursor(1 - 1, 1 - 1);
        mylcd.print("again");
      }
    }
  }
  infrar = digitalRead(2);//assign the value of digital 2 to
infrar
  if (infrar == 0 && (val!= 'l' && val!= 't'))
    //if variable infrar is 0 and val is not 'l' either 't'
  {
    servo_9.write(0);//set servo connected to digital 9 to 0°
    delay(50);
  }
  if (button2 == 0)//if variablebutton2 is 0
  {
    delay(10);
    while (button2 == 0) //if variablebutton2 is 0, program will
circulate
    {
      button2 = digitalRead(8);//assign the value of digital 8 to
button2
      btn2_num = btn2_num + 1;//variable btn2_num plus 1
      delay(100);
      if (btn2_num >= 15)//if variablebtn2_num ≥15
      {
        tone(3, 532);
        delay(125);
        mylcd.clear();//clear LCD screen
        //LCD shows "password:" at the first character on first
row
        mylcd.setCursor(1 - 1, 1 - 1);
        mylcd.print("password:");
        //LCD shows "wait" at the first character on first row
        mylcd.setCursor(1 - 1, 1 - 1);
        mylcd.print("wait");
      } else//Otherwise
      {
        noTone(3);//digital 3 stops playing music
      }
    }

  }
  btn1_num = 0;//set btn1_num to 0
  btn2_num = 0;//set btn2_num to 0
}

// Birthday song
void music1() {
```

```
  birthday();
}
//Ode to joy
void music2() {
  Ode_to_Joy();
}
void Ode_to_Joy()//play Ode to joy song
{
  for (int x = 0; x < length; x++)
  {
    tone(tonepin, tune[x]);
    delay(300 * durt[x]);
  }
}

//PWM control
void pwm_control() {
  switch (val)
  {
    case 't'://if val is 't', program will circulate
      servo1 = Serial.readStringUntil('#');
      servo1_angle = String(servo1).toInt();
      servo_9.write(servo1_angle);//set the angle of servo
connected to digital 9 to servo1_angle
      delay(300);
      break;//exit loop
    case 'u'://if val is 'u', program will circulate
      servo2 = Serial.readStringUntil('#');
      servo2_angle = String(servo2).toInt();
      servo_10.write(servo2_angle);//set the angle of servo
connected to digital 10 to servo2_angle
      delay(300);
      break;//exit loop
    case 'v'://if val is 'v', program will circulate
      led2 = Serial.readStringUntil('#');
      value_led2 = String(led2).toInt();
      analogWrite(5, value_led2); //PWM value of digital 5 is
value_led2
      break;//exit loop
    case 'w'://if val is 'w', program will circulate
      fans_char = Serial.readStringUntil('#');
      fans_val = String(fans_char).toInt();
      digitalWrite(7, LOW);
      analogWrite(6, fans_val); //set PWM value of digital 6 to
fans_val, the larger the value, the faster the fan
break;//exit loop
  }
}
```

2. System Functionality:

- **Sensor Data Acquisition:** The Arduino continuously reads data from all connected sensors, monitoring various aspects of the home environment.
- **Real-time Monitoring:** Sensor data is transmitted securely via BLE to the mobile application for real-time visualization, allowing users to remotely monitor their home.
- **Remote Control:** Users can control various aspects of their smart home through the familiar interface of the existing mobile application, including:
  - Turning lights and appliances on/off remotely.
  - Adjusting thermostat settings for temperature control.
- **Automated Routines:** The system can be programmed for automated routines based on time, user presence, or sensor data. Examples include:
  - Automatically adjusting light levels based on the photocell sensor readings.
  - Activating watering schedules for plants based on soil moisture readings.
- **Security Alerts:** The system triggers alerts within the mobile application for potential security breaches, gas leaks, smoke detection, and other emergencies, allowing for a quick response.
- **User Authentication:** The existing mobile application may already have a user authentication mechanism (password, pin code) to restrict unauthorized access. If not, this will be implemented within the application.

3. System Benefits:

- **Enhanced Security:** Real-time monitoring and alerts deter unauthorized entry and notify you of potential security breaches.
- **Improved Comfort and Convenience:** Automated climate control, lighting adjustments, and remote control of appliances offer a more comfortable and convenient living experience.
- **Increased Energy Efficiency:** Sensor-based automation optimizes device usage, leading to reduced energy consumption.
- **Peace of Mind:** Real-time monitoring and alerts provide peace of mind when you're away from home.
- **User-Friendly Interface:** The existing mobile application offers a familiar interface for easy interaction and control.

4. Security Considerations:

- Secure communication protocols like BLE encrypt data transmission between the Arduino and the mobile application.
- User authentication within the existing mobile application restricts unauthorized access to the system.
- Consider implementing data encryption for sensitive information, especially if user data is stored within the application.

5. Conclusion:

This system description outlines a secure and user-friendly smart home automation system that leverages existing technologies like Arduino and a mobile application. The system provides a practical and user-friendly solution for creating a more secure, comfortable, and energy-efficient home environment.

System Integration

Objective

The objective of system integration is to connect the Arduino project with other external systems to create a more comprehensive and automated solution. This can enhance functionality, enable remote monitoring and control, and facilitate data analysis.

Integration Options

The choice of integration approach depends on the target system and desired level of complexity. Here are some potential options:

- **Home Automation Hubs (SmartThings, Hubitat):** Seamlessly integrate the Arduino project into your smart home environment, allowing control through a hub app, voice assistants, and automation routines.
- **Cloud Platforms (ThingSpeak, Adafruit IO):** Enable remote monitoring of sensor data (e.g., temperature, humidity) over the internet. You can view sensor readings in real-time through a web dashboard or mobile app, set up alerts for critical values, and even potentially store and analyze historical data.
- **Smartphone App (Direct):** Develop a dedicated mobile app that communicates directly with the Arduino project via Bluetooth. This app can provide a user-friendly interface for control (turning on lights, adjusting settings) and real-time sensor data visualization.
- **Additional Sensors:** Expand the project's capabilities by integrating new sensors like temperature sensors for climate control, motion sensors for security, or soil moisture sensors for smart gardens. Combining data from multiple sensors allows for more intelligent decision-making and automation.

Benefits of System Integration

- **Enhanced Functionality:** Connects your Arduino system with other platforms, offering more control options and expanding its capabilities.
- **Remote Monitoring and Control:** Allows you to monitor sensor data and control functionalities remotely through web dashboards, mobile apps, or voice assistants.
- **Improved Automation:** Enables the creation of more complex automations based on combined sensor data from various sources. This can lead to a more convenient and efficient system.
- **Scalability:** The system can be easily expanded by integrating additional sensors and devices as needed, making it future-proof.

Considerations

- **Communication Protocols:** Choose appropriate communication protocols like Bluetooth, WiFi, or Ethernet depending on the desired range and complexity of the integration.
- **Security:** If connecting to the internet, implement security measures like secure communication protocols, authentication, and encryption to protect your data and prevent unauthorized access to your system.
- **Power Requirements:** Ensure a reliable power source with sufficient capacity for the Arduino and any additional integrated devices.

Simulation and experimental results.

We conducted 15 experiments before starting the project to ensure that everything was functioning properly and capable of performing its intended tasks in our smart home.

Project 1: LED Blink

Description: This project blinks an LED on and off every second.
Code :

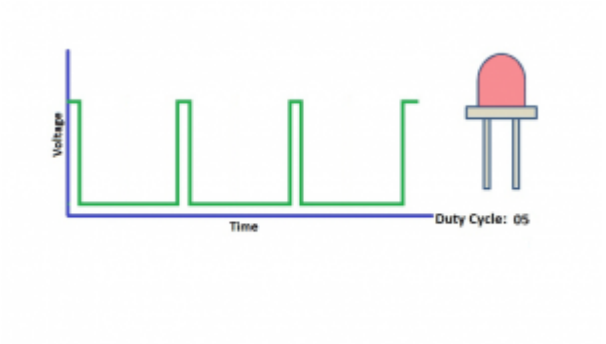
```
void setup() {
  pinMode(13, OUTPUT); // Initialize digital pin 13 as an output
}

void loop() {
  digitalWrite(13, HIGH); // Turn the LED on
  delay(1000); // Wait for a second
  digitalWrite(13, LOW); // Turn the LED off
  delay(1000); // Wait for a second
}
```

Project 2: Breathing Light

Description: This project uses Pulse Width Modulation (PWM) to gradually increase and decrease the brightness of an LED, mimicking a breathing pattern.

Our objective is to manipulate the brightness of an LED, mimicking a natural breathing pattern. Imagine the LED gradually intensifying in brightness, then smoothly fading, emulating the rhythm of respiration.



Beyond On/Off: The Magic of PWM
Traditional Arduino pins operate digitally, offering only two distinct states: HIGH (typically 5V) and LOW (usually 0V). This might seem limiting for controlling brightness, but here's where PWM steps in.
PWM offers a clever way to achieve an analog-like effect using digital signals. It works by rapidly switching the pin between HIGH and LOW at a specific frequency, creating a square wave. The critical factor here is the duty cycle, which represents the portion of time the signal spends at HIGH voltage compared to the total cycle time.

Code:

```
int ledPin = 5; // Define the LED pin at D5

void setup () {
  pinMode(ledPin, OUTPUT); // Initialize ledPin as an output
}

void loop () {
  for (int value = 0; value < 255; value++) {
    analogWrite(ledPin, value); // Gradually increase LED brightness
    delay(5); // Delay 5ms
  }
  for (int value = 255; value > 0; value--) {
    analogWrite(ledPin, value); // Gradually decrease LED brightness
    delay(5); // Delay 5ms
  }
}
```

Project 3: Passive Buzzer

Description: This project generates a tone using a passive buzzer connected to pin D3.

```
int tonePin = 3; // Set the Pin of the buzzer to digital D3
```

Code:

```
void setup () {
  pinMode(tonePin, OUTPUT); // Set the digital IO pin mode to output
}

void loop () {
  for (unsigned char i = 0; i < 80; i++) {
    digitalWrite(tonePin, HIGH); // Sound
    delay(1); // Delay 1ms
    digitalWrite(tonePin, LOW); // No sound
    delay(1); // Delay 1ms
  }
  for (unsigned char i = 0; i < 100; i++) {
    digitalWrite(tonePin, HIGH); // Sound
    delay(2); // Delay 2ms
    digitalWrite(tonePin, LOW); // No sound
    delay(2); // Delay 2ms
  }
}
```

Project 4：Controlling LED By Button Module

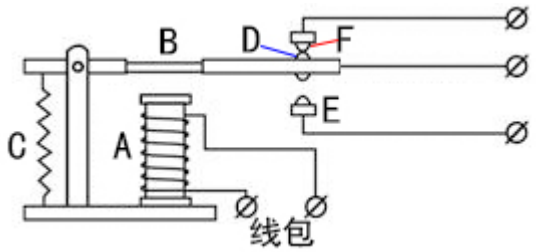
Description: This project turns an LED on when a button is pressed.

Code :

```
int ledpin = 5; // Define the led light in D5
int inpin = 4; // Define the button in D4
int val; // Define variable val
void setup ()
{
pinMode (ledpin, OUTPUT); // The LED light interface is defined as output
pinMode (inpin, INPUT); // Define the button interface as input
}
void loop ()
{
val = digitalRead (inpin); // Read the digital 4 level value and assign it to val
if (val == LOW) // Whether the key is pressed, the light will be on when pressed
{digitalWrite (ledpin, HIGH);}
else
{digitalWrite (ledpin, LOW);}
}
```

Project 5：1-channel Relay Module

Description: This project demonstrates how to control a relay module to turn a device on and off.



Code :

```
int Relay = 12; // Define the relay pin at D12
void setup ()
{
pinMode (13, OUTPUT); // Set Pin13 as output
digitalWrite (13, HIGH); // Set Pin13 High
pinMode (Relay, OUTPUT); // Set Pin12 as output
}
void loop ()
{
digitalWrite (Relay, HIGH); // Turn off relay
delay (2000);
digitalWrite (Relay, LOW); // Turn on relay
delay (2000);
```

```
}
Project 6：Photocell Sensor
int LED = 5; // Set LED pin at D5
int val = 0; // Read the voltage value of the photodiode
void setup () {
      pinMode (LED, OUTPUT); // LED is output
      Serial.begin (9600); // The serial port baud rate is set to 9600
}
void loop () {
      val = analogRead (A1); // Read the voltage value of A1 Pin
      Serial.println (val); // Serial port to view the change of voltage value
      if (val <900)
      {// Less than 1000, LED light is off
      digitalWrite (LED, LOW);
      }
      else
      {// Otherwise, the LED lights up
      digitalWrite (LED, HIGH);
      }
      delay (10); // Delay 10ms
}
```

Project 6: Photocell Sensor

Description: This project reads the analog value from a photocell sensor and turns an LED on or off based on the light intensity.

Code

```
int LED = 5; // Set LED pin at D5

void setup () {
  pinMode(LED, OUTPUT); // LED is output
  Serial.begin(9600); // Set the serial port baud rate to 9600
}

void loop () {
  int val = analogRead(A1); // Read the voltage value of A1 Pin
  Serial.println(val); // Serial port to view the change of voltage value
  if (val < 900) {
    digitalWrite(LED, LOW); // LED off if light is low
  } else {
    digitalWrite(LED, HIGH); // LED on if light is high
  }
  delay(10); // Delay 10ms
}
```

Project 7：Adjusting Motor Servo Angle

Description: This project uses a servo motor to rotate an object back and forth.

Code

```
#include <Servo.h> // Servo function library
Servo myservo;
int pos = 0; // Start angle of servo
void setup ()
{
myservo.attach (9); // Define the position of the servo on D9
}
void loop ()
{
for(pos = 0; pos < 180; pos += 1)// angle from 0 to 180 degrees
{
myservo.write (pos); // The servo angle is pos
delay (15); // Delay 15ms
}
for(pos = 180; pos>=1; pos-=1) // Angle from 180 to 0 degrees
{
myservo.write (pos); // The angle of the servo is pos
delay (15); // Delay 15ms
}
}
```

Project 8: Fan Module

Description: This project demonstrates controlling a fan's direction using two digital pins.

Code

```
void setup () {
  pinMode (7, OUTPUT); //define D7 pin as output
  pinMode (6, OUTPUT); //define  D6 pin as output
}
void loop () {
  digitalWrite (7, LOW);
  digitalWrite (6, HIGH); // Reverse rotation of the motor
  delay (3000); // delay 3S
  digitalWrite (7, LOW);
  digitalWrite (6, LOW); // The motor stops rotating
  delay (1000); //delay 1S
  digitalWrite (7, HIGH);
  digitalWrite (6, LOW); // The motor rotates in the forward direction
  delay (3000); // delay 3S
}
```

Project 9：Steam Sensor

Description: This project reads the analog value from a steam sensor and prints the moisture level via serial communication.

Code :

```
void setup()
{
Serial.begin(9600); //open serial port, and set baud rate at 9600bps
}
void loop()
{
int val;
val=analogRead(3); //plug vapor sensor into analog port 3
Serial.print("Moisture is ");
Serial.println(val,DEC); //read analog value through serial port printed
delay(100); //delay 100ms
}
```

Project 10: PIR Motion Sensor

Description: This project uses a PIR motion sensor to detect movement and control an LED and a fan.

Code :

```
void setup () {
  Serial.begin (9600); // open serial port, and set baud rate at 9600bps
  pinMode (2, INPUT); // Define PIR as input in D2
  Serial.begin (9600);
  pinMode (13, OUTPUT); // Define LED as output in D13
  pinMode (7, OUTPUT); // Define D7 as output
  pinMode (6, OUTPUT); // Define D6 as output
}

void loop () {
  Serial.println (digitalRead (2));
  delay (500); // Delay 500ms
  if (digitalRead (2) == 1) // If someone is detected walking
 {
    digitalWrite (13, HIGH); // LED light is on
    digitalWrite (7, HIGH);
    analogWrite (6,150); // Fan rotates

  } else // If no person is detected walking
{
    digitalWrite (13, LOW); // LED light is not on
    digitalWrite (7, LOW);
    analogWrite (6,0); // The fan does not rotate
  }}
```

Project 11: Analog (MQ-2) Sensor

Description: This project reads the analog value from an MQ-2 gas sensor and triggers a buzzer if the gas concentration exceeds a threshold.

Code :

```
void setup() {
  pinMode(A0, INPUT); // Define MQ2 gas sensor pin at A0
  pinMode(3, OUTPUT); // Define the buzzer pin at D3
}

void loop() {
  int val = analogRead(A0); // Read the voltage value of A0 port
  if (val > 240) {
    tone(3, 589); // Turn on buzzer
    delay(300);
  } else {
    noTone(3); // Turn off buzzer
  }
}
```

Project 12: 1602 LCD Display

Description: This project initializes and prints messages to a 1602 LCD display.

Code :

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2); // Set the LCD address to 0x27 for a 16x2 display

void setup() {
  lcd.init(); // Initialize the LCD
  lcd.backlight(); // Turn on the backlight
  lcd.setCursor(3, 0);
  lcd.print("Hello, world!"); // Print "Hello, world!" to the LCD
  lcd.setCursor(2, 1);
  lcd.print("zhinengchuanke"); // Print "zhinengchuanke" to the LCD
}

void loop() {
}
```

Project 13: Soil Humidity Sensor

Description: This project reads the analog value from a soil humidity sensor and displays the soil condition on an LCD display.

Code:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2); // Set the LCD address to 0x27 for a 16x2 display

void setup() {
  lcd.init(); // Initialize the LCD
  lcd.backlight(); // Turn on the backlight
  lcd.clear(); // Clear the screen
  pinMode(A2, INPUT); // Set the soil sensor pin as input
}

void loop() {
  int value = analogRead(A2); // Read the value of the soil sensor
  if (value < 100) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Value:");
    lcd.setCursor(6, 0);
    lcd.print(value);
    lcd.setCursor(0, 1);
    lcd.print("Dry soil");
    delay(300);
  } else if (value >= 100 && value <= 200) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Value:");
    lcd.setCursor(6, 0);
    lcd.print(value);
    lcd.setCursor(0, 1);
    lcd.print("Humid soil");
    delay(300);
  } else if (value > 300) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Value:");
    lcd.setCursor(6, 0);
    lcd.print(value);
    lcd.setCursor(0, 1);
    lcd.print("In water");
    delay(300);
```

}}

Project 14: Bluetooth Test

Description: This project reads and prints values sent via Bluetooth.

Code :

```
void setup() {
  Serial.begin(9600); // Set the serial port baud rate to 9600
}

void loop() {
  while (Serial.available() > 0) {
    char val = Serial.read(); // Read value sent by Bluetooth
    Serial.print(val); // Print read value
  }
}
```

Conclusion

After completing the smart home project, several conclusions can be drawn. Firstly, the integration of various sensors, actuators, and communication modules within the Arduino framework proved to be effective in creating a functional smart home automation system. Each component, from the LED control to the gas sensor monitoring, contributed to the overall functionality and usability of the system.
Secondly, the use of mobile applications for remote control and monitoring enhanced the accessibility and convenience of the smart home system. The integration of Bluetooth modules allowed for seamless communication between the Arduino board and the mobile app, providing users with an intuitive interface for managing their home devices.
Lastly, the project highlighted the importance of energy efficiency and security in smart home systems. The implementation of sensors such as the photocell sensor for automatic lighting control and the PIR motion sensor for intruder detection demonstrated how smart technologies can enhance energy savings and home security.

Recommendations for Future Work:

1. **Enhanced Security Features:** Incorporating more advanced security features, such as facial recognition or voice authentication, can further enhance the security of the smart home system.
2. **Energy Optimization:** Implementing more advanced algorithms for energy optimization, such as machine learning algorithms for predicting user behavior, can help further reduce energy consumption in the smart home.
3. **Expandable System:** Designing the system to be easily expandable, allowing for the addition of more sensors and actuators in the future, will ensure that the smart home system remains adaptable to changing needs.
4. **Remote Monitoring:** Integrating sensors for monitoring environmental factors such as air quality and temperature, and providing real-time feedback to users, can enhance the overall comfort and health benefits of the smart home.
5. **User-Friendly Interface:** Continuously improving the mobile application interface to be more user-friendly and intuitive will enhance the overall user experience of the smart home system.

In conclusion, the smart home project successfully demonstrated the capabilities of an Arduino-based smart home automation system. By implementing the above recommendations, future iterations of the project can further enhance the functionality, efficiency, and usability of the smart home system.