Table of Contents

Chapter 1: Introduction

1.1 Home Automation: An Overview

Home automation, also known as domotics, refers to the use of technology to automate various tasks within the home. It involves controlling and monitoring home appliances and systems, such as lighting, heating, ventilation, air conditioning (HVAC), security, and entertainment, to improve convenience, comfort, energy efficiency, and security. Home automation systems typically consist of a central control unit, sensors, actuators, and a communication network that allows the user to interact with the system.

1.2 Project Overview

This project focuses on developing a basic home automation system using the Arduino Uno microcontroller, the HC05 Bluetooth module, and relay modules. The system allows users to wirelessly control electrical appliances in their homes using an Android mobile application. This system provides a cost-effective and flexible solution for home automation, which can be further expanded to include more advanced features.

1.3 Objectives

The main objectives of this project are:

- To design and implement a cost-effective home automation system.
- To enable wireless control of home appliances using a smartphone.
- To utilize Arduino Uno, HC05 Bluetooth module, and relay modules for system development.
- To develop an Android application for user interaction.
- To ensure the system is user-friendly, reliable, and safe.

1.4 Scope

The scope of this project includes:

- Controlling up to four electrical appliances (e.g., lights, fans) using a relay module.
- Wireless communication between the Android application and the Arduino Uno via Bluetooth.
- Development of a user-friendly Android application.
- Implementation of basic on/off control for the connected appliances.

1.5 Benefits of Home Automation

Home automation offers several benefits, including:

- **Convenience:** Control appliances from anywhere using a smartphone.
- **Comfort:** Automate settings for optimal living conditions.
- **Energy Efficiency:** Reduce energy consumption by controlling devices.
- **Security:** Enhance home security with automated systems.
- **Accessibility:** Provide easier control for elderly and disabled individuals.

Chapter 2: Literature Review

2.1 Existing Home Automation Systems

Existing home automation systems range from simple, stand-alone systems to complex, integrated networks. Some common types include:

- **Wired Systems:** These systems use physical cables to connect the control unit to the devices. They are reliable but can be difficult and expensive to install.
- **Wireless Systems:** These systems use wireless technologies like Wi-Fi, Zigbee, or Z-Wave to communicate between the control unit and the devices. They offer more flexibility and easier installation.
- **Smart Home Hubs:** These central control devices can integrate various smart devices and allow control through a single interface. Examples include Amazon Echo, Google Home, and Apple HomeKit.

2.2 Bluetooth Technology in Home Automation

Bluetooth is a wireless communication technology that allows devices to exchange data over short distances. It is a popular choice for home automation due to its:

- **Low Cost:** Bluetooth modules are relatively inexpensive.
- **Low Power Consumption:** Suitable for battery-powered devices.
- **Ease of Use:** Simple pairing and configuration process.
- **Wide Availability:** Most smartphones and tablets support Bluetooth.

2.3 Arduino in Home Automation

Arduino is an open-source electronics platform based on easy-to-use hardware and software. It is widely used in home automation projects because of its:

- **Low Cost:** Arduino boards are affordable.
- **Flexibility:** Can be used with a wide range of sensors and actuators.
- **Ease of Programming:** Simple programming language and IDE.
- **Large Community Support:** Plenty of online resources and libraries.

2.4 Relay Modules in Automation

A relay is an electrically operated switch. Relay modules are used in home automation systems to control high-voltage devices, such as lights and fans, with the low-voltage signals from a microcontroller like Arduino. They provide electrical isolation, ensuring safety and preventing damage to the control circuit.

2.5 Comparison with Other Technologies

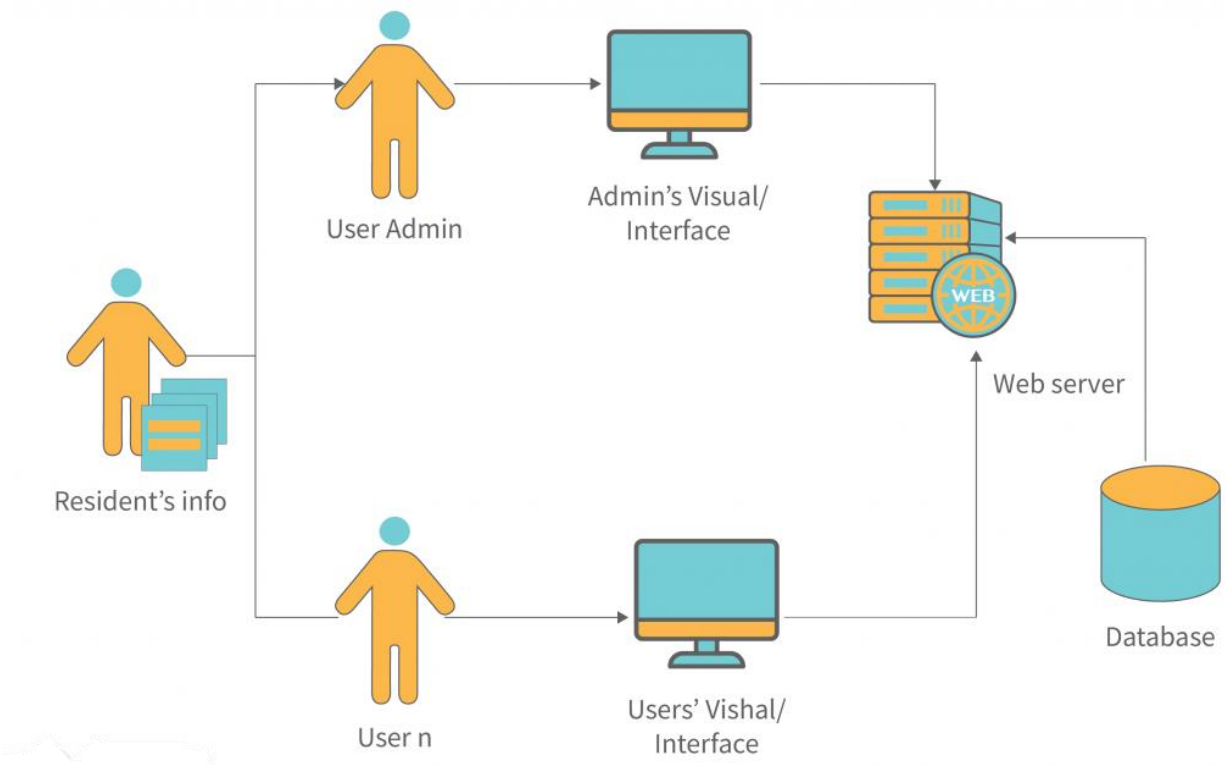
Technology	Advantages	Disadvantages
Bluetooth	Low cost, low power consumption, easy to use	Short range, limited bandwidth
Wi-Fi	Long range, high bandwidth	Higher power consumption, more complex setup
Zigbee	Low power consumption, mesh networking	Lower bandwidth, more complex than Bluetooth
Z-Wave	Low power consumption, reliable, mesh networking	Proprietary, limited device selection
Wired	High reliability, high speed	High cost, difficult installation, low flexibility

Chapter 3: System Design and Methodology

3.1 System Architecture

The system architecture consists of the following main components:

1. **Android Mobile Application:** Provides a user interface to control the home appliances. It communicates with the Arduino Uno via Bluetooth.
2. **HC05 Bluetooth Module:** Enables wireless communication between the Android application and the Arduino Uno.
3. **Arduino Uno:** The microcontroller that receives commands from the Bluetooth module and controls the relay module.
4. **Relay Module:** Switches the electrical appliances (lights, fans, etc.) on or off based on the signals from the Arduino Uno.
5. **Electrical Appliances:** The devices that are being controlled (e.g., lights, fans).
6. **Power Supply:** Provides power to the Arduino Uno, HC05, and relay module.

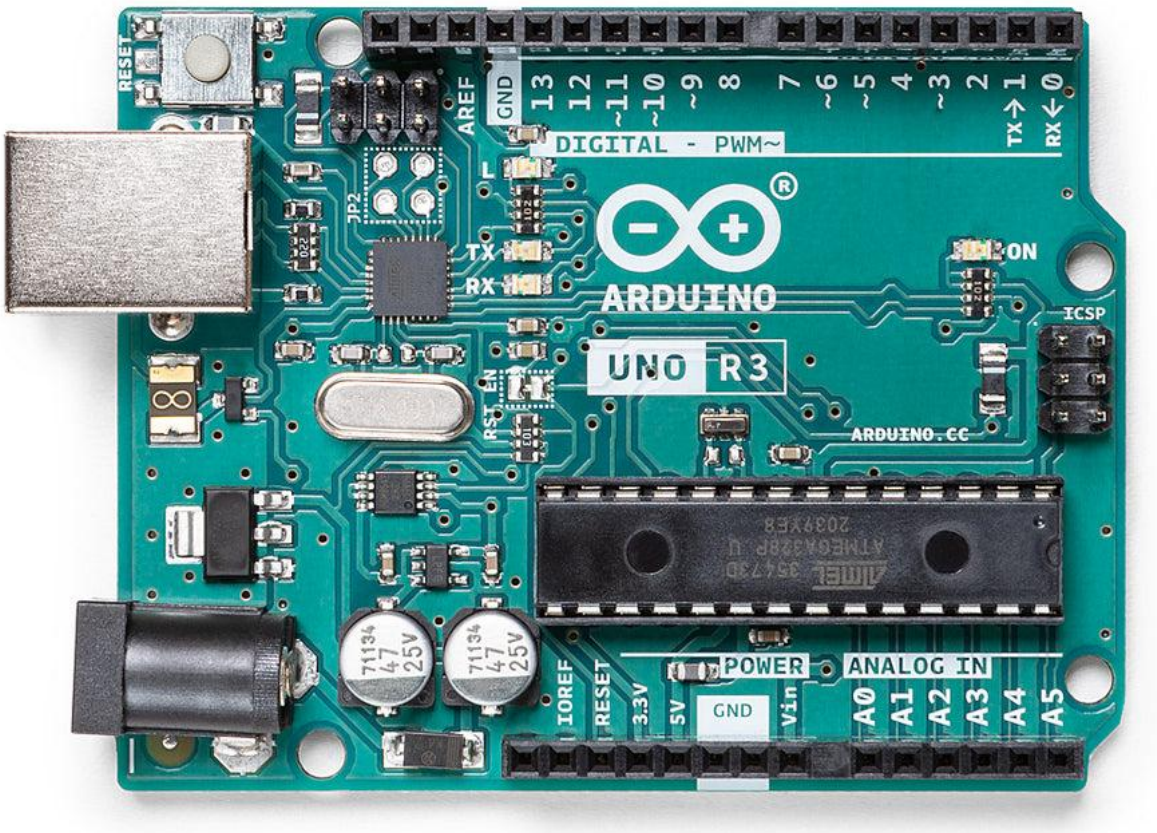


3.2 Hardware Components

- **Arduino Uno**
- **HC05 Bluetooth Module**
- **Relay Module**
- **Power Supply**
- **Jumper Wires**
- **Electrical Appliances**

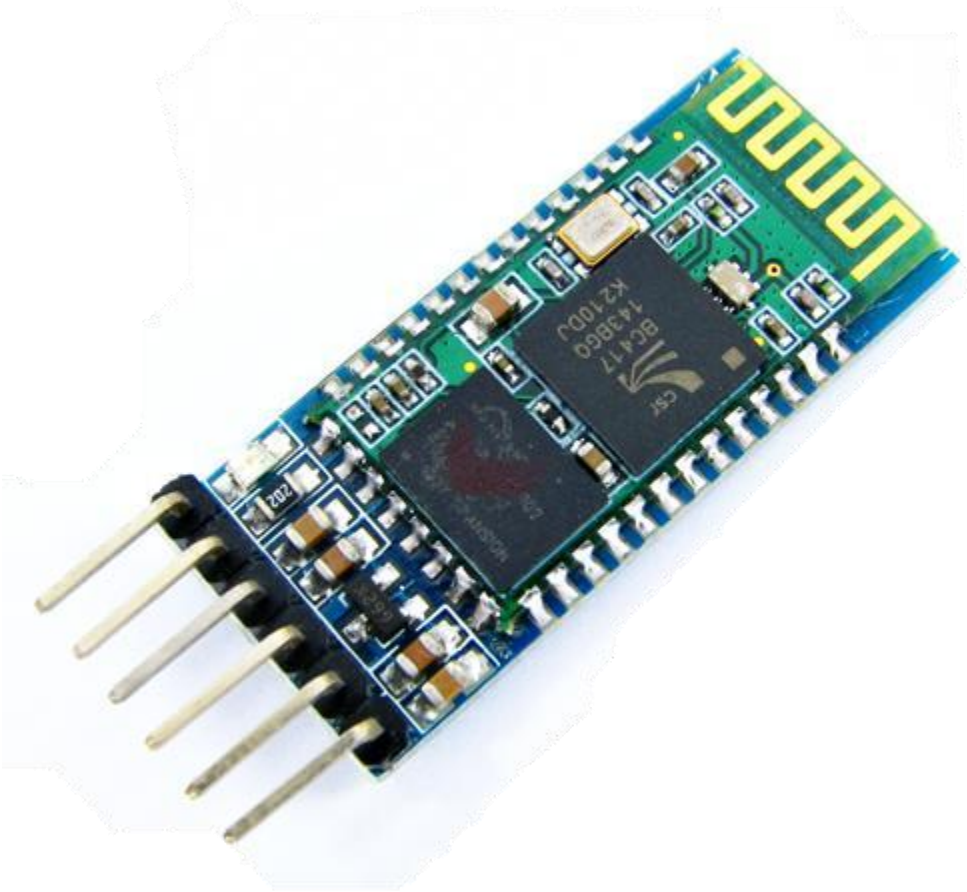
3.2.1 Arduino Uno

The Arduino Uno is a popular open-source microcontroller board based on the ATmega328P microcontroller. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 Analog inputs, a USB connection, a power jack, an ICSP header, and a reset button. In this project, the Arduino Uno is used to receive commands from the HC05 Bluetooth module and control the relay module.



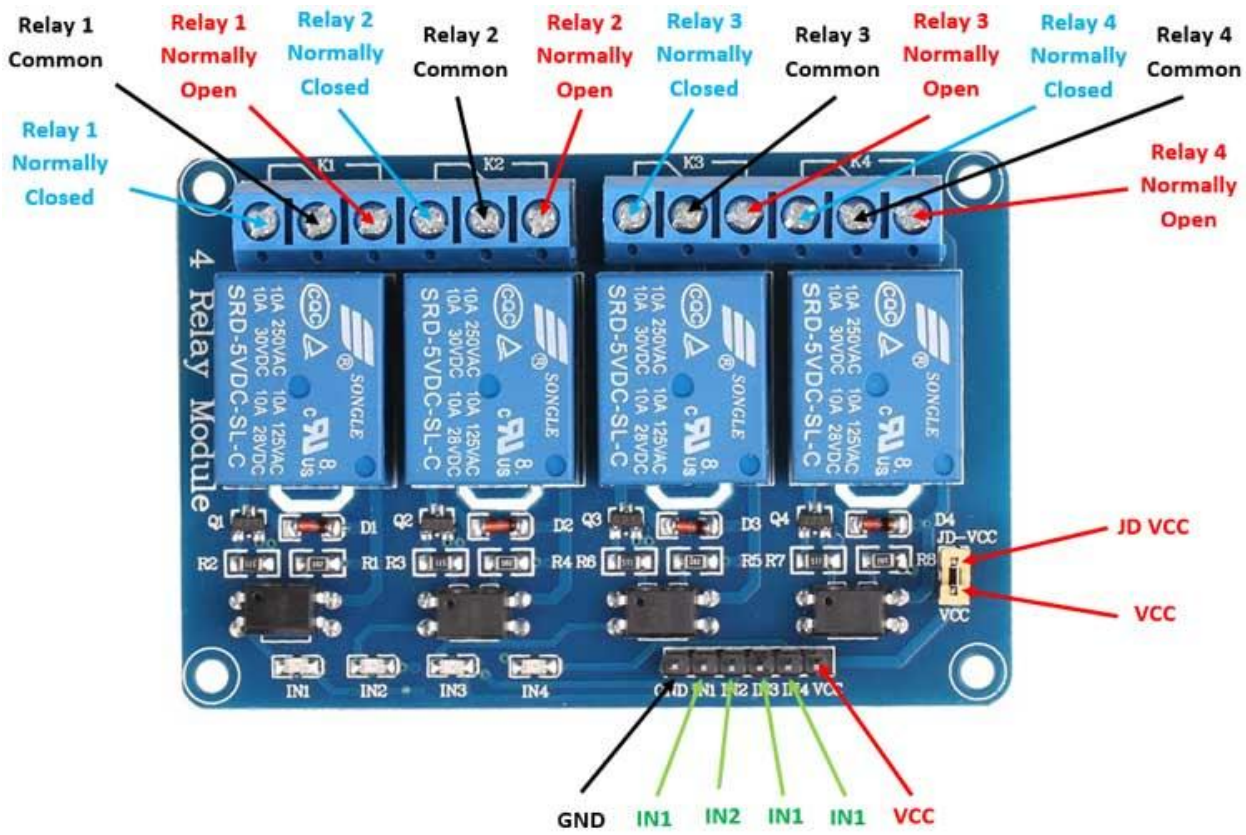
3.2.2 HC05 Bluetooth Module

The HC05 is a Bluetooth module that allows wireless serial communication between devices. It operates in the 2.4GHz frequency band and uses the Serial Port Profile (SPP) to communicate with other devices. In this project, the HC05 module receives control signals from the Android application and transmits them to the Arduino Uno.



3.2.3 Relay Module

A relay module is an electronic switch that allows low-voltage circuits to control high-voltage circuits. It consists of relays, transistors, diodes, and connectors. The relay module used in this project has four relays, which can be used to control four different electrical appliances.



3.2.4 Power Supply

A power supply is needed to provide power to the Arduino Uno, HC05 Bluetooth module, and the relay module. A 5V power supply is typically used for the Arduino and HC05, while the relay module may require 5V or 12V depending on the type of relays used. A USB power adapter or a dedicated power supply can be used.

3.2.5 Jumper Wires

Jumper wires are essential components in electronics, used to connect different parts of a circuit without soldering. Jumper wires also vary in colour, but the colours don't affect functionality—they simply help users organize connections.

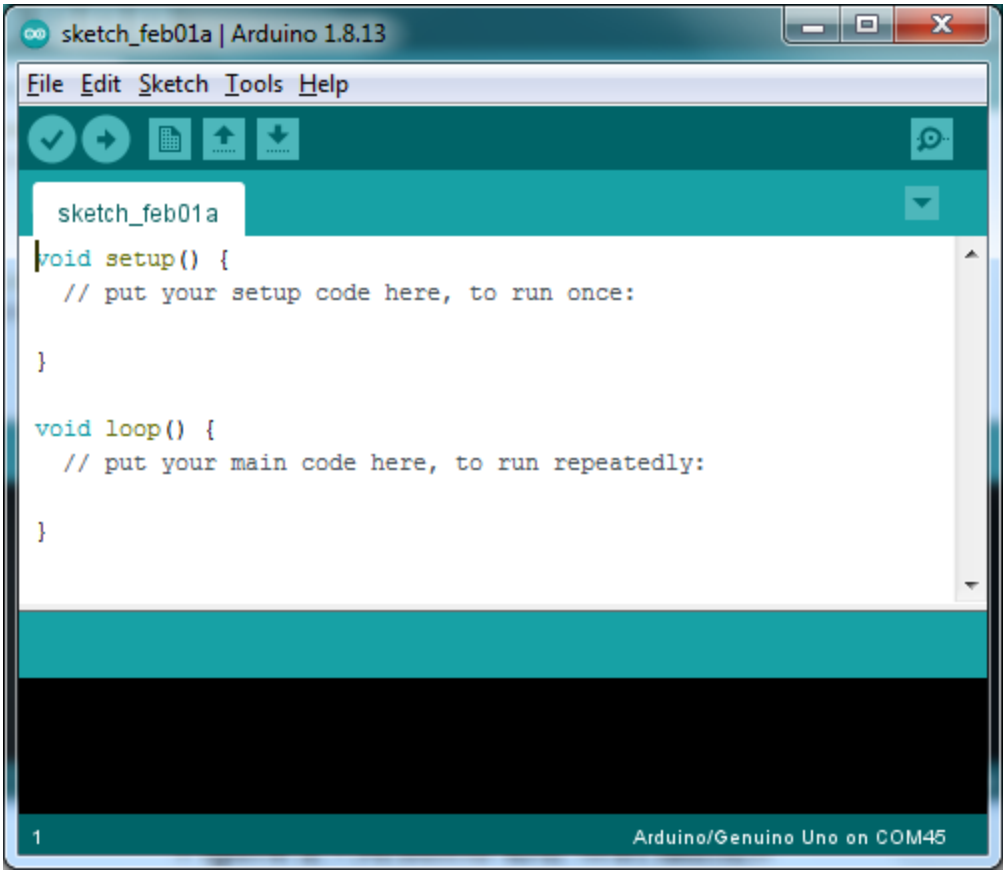
3.2.6 Electrical Appliances

The electrical appliances to be controlled in this project can be any device that operates on AC power, such as lights, fans, and other household appliances.

3.3 Software

3.3.1 Arduino IDE

The Arduino IDE (Integrated Development Environment) is a software application used to write and upload code to Arduino boards. It provides a simple and user-friendly interface for writing code in the Arduino programming language, which is based on C++.

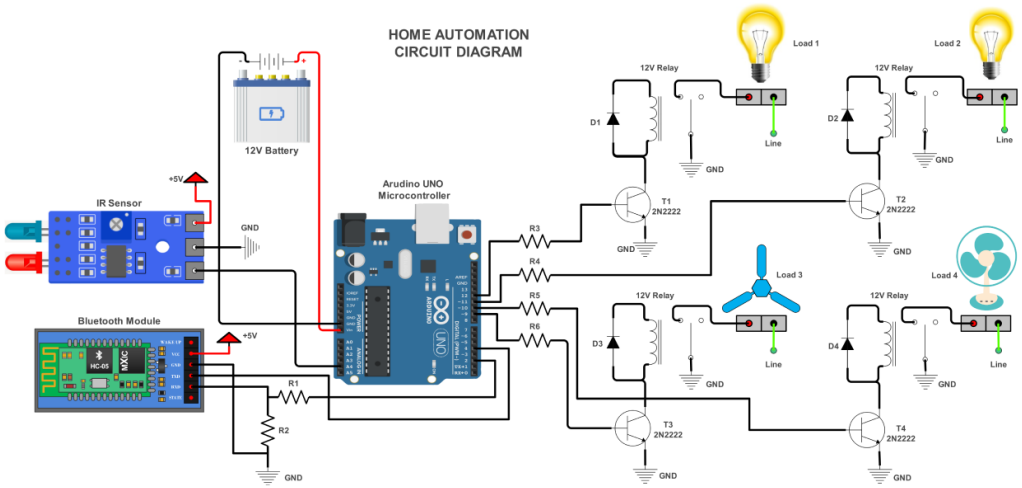


3.3.2 Android Application

An Android application is developed to provide a user interface for controlling the home automation system. The name of application is Arduino Bluetooth Controller which can be easily found on Play Store, App Store etc. The application will include features for:

- Establishing a Bluetooth connection with the HC05 module.
- Sending commands to the Arduino Uno to control the relays.

3.4 Circuit Diagram and Explanation



Circuit Explanation:

1. **Power Supply:** The 5V power supply is connected to the VCC and GND pins of the Arduino Uno, HC05 Bluetooth module, and relay module.
2. **Arduino Uno and HC05:**
   - The TX pin of the HC05 is connected to the RX pin of the Arduino Uno.
   - The RX pin of the HC05 is connected to the TX pin of the Arduino Uno.
   - The VCC and GND pins of the HC05 are connected to the 5V and GND pins of the Arduino Uno.
3. **Arduino Uno and Relay Module:**
   - Digital output pins from the Arduino Uno (e.g., pins 2, 3, 4, and 5) are connected to the input pins (IN1, IN2, IN3, and IN4) of the relay module.
   - The VCC and GND pins of the relay module are connected to the 5V and GND pins of the Arduino Uno (or an external power supply if needed).
4. **Relay Module and Electrical Appliances:**
   - The common (COM), normally open (NO), and normally closed (NC) terminals of the relay module are connected to the electrical appliances and the AC power supply. The specific wiring depends on whether you want the appliance to be normally on or normally off. For controlling home appliances, connect the COM terminal to the AC power source, and connect the NO

terminal to the appliance.

3.5 Working Principle

The home automation system works as follows:

1. The user sends a command (e.g., "turn on light 1") from the Android application on their smartphone.
2. The Android application transmits the command wirelessly to the HC05 Bluetooth module.
3. The HC05 Bluetooth module receives the command and sends it to the Arduino Uno via serial communication.
4. The Arduino Uno receives the command and processes it.
5. The Arduino Uno sends a control signal to the corresponding pin of the relay module.
6. The relay module activates the appropriate relay, which switches the connected electrical appliance on or off.
7. The electrical appliance (e.g., light) turns on or off accordingly.

3.6 Flowchart

Flowchart for Arduino Home Automation System

Using "Arduino Bluetooth Controller" App

[Flowchart of the system's working principle, showing the sequence of steps from user command to appliance control using the "Arduino Bluetooth Controller" app]

Explanation of the Flowchart:

1. **Start:** The system begins when the user interacts with the "Arduino Bluetooth Controller" app.

2. **User Sends Command from App:** The user presses a button in the "Arduino Bluetooth Controller" app, configured to send a specific command (e.g., '1', '2', '3', '4', '0', or '5').

3. **Transmit via Bluetooth:** The Android device transmits the command wirelessly to the HC05 Bluetooth module.

4. **Receive Command:** The HC05 Bluetooth module receives the command.

5. **Send to Arduino:** The HC05 module sends the command to the Arduino Uno via serial communication.
6. **Arduino Receives Command:** The Arduino Uno receives the command.
7. **Process Command:** The Arduino Uno processes the command to determine which relay to control and what action to take (ON/OFF).
8. **Send Signal to Relay:** The Arduino Uno sends a control signal (HIGH or LOW) to the corresponding pin of the relay module.
9. **Relay Activates/Deactivates:** The relay module activates or deactivates the appropriate relay.
10. **Switch Appliance:** The relay switches the connected electrical appliance ON or OFF.
11. **Appliance Changes State:** The electrical appliance's state changes (turns ON or OFF).
12. **End:** The process completes. The system waits for the next command from the user via the "Arduino Bluetooth Controller" app.

Chapter 4: Hardware Implementation

4.1 Component Interfacing

4.1.1 Interfacing Arduino with HC05

1. Connect the VCC pin of the HC05 to the 3.3V pin on the Arduino Uno.
2. Connect the GND pin of the HC05 to the GND pin on the Arduino Uno.
3. Connect the TX pin of the HC05 to the RX pin on the Arduino Uno (through a voltage divider if the HC05 operates on 3.3V).
4. Connect the RX pin of the HC05 to the TX pin on the Arduino Uno.

4.1.2 Interfacing Arduino with Relay Module

1. Connect the VCC pin of the relay module to the 5V pin on the Arduino Uno.
2. Connect the GND pin of the relay module to the GND pin on the Arduino Uno.
3. Connect the input pins of the relay module (IN1, IN2, IN3, IN4) to digital output pins on the Arduino Uno (e.g., pins 2, 3, 4, and 5).

4.1.3 Connecting the Power Supply

1. Connect the positive terminal of the 5V power supply to the VCC pins of the Arduino Uno, HC05 Bluetooth module, and relay module.
2. Connect the negative terminal of the power supply to the GND pins of the Arduino Uno, HC05 Bluetooth module, and relay module.
3. Ensure that the power supply has sufficient current rating to power all the components.

4.1.4 Connecting the Electrical Appliances

- **Caution:** Working with AC power can be dangerous. Ensure that all power is disconnected before making any connections. If you are not comfortable working with AC power, seek the help of a qualified electrician.

1. Connect the common (COM) terminal of the relay to the AC power source.
2. Connect the normally open (NO) terminal of the relay to one terminal of the electrical appliance.
3. Connect the other terminal of the electrical appliance to the other terminal of the AC power source.
4. Repeat this process for each appliance that you want to control.

4.2 PCB Design (Optional)

For a more permanent and robust solution, you can design a printed circuit board (PCB) to mount the components. PCB design software such as Eagle, KiCad, or Fritzing can be used. The PCB should be designed to accommodate the Arduino Uno, HC05 Bluetooth module, relay module, and power supply connections.

4.3 Enclosure Design

To protect the electronic components and provide a professional look, you can design an enclosure for the system. The enclosure can be made from plastic, wood, or metal. It should have openings for the power cord, appliance connections, and any other necessary interfaces. Ensure that the enclosure is properly ventilated to prevent overheating.

Chapter 5: Software Implementation

5.1 Arduino Code

The Arduino code is responsible for receiving commands from the HC05 Bluetooth module and controlling the relay module.

```
const int relayPins[] = {12, 13, 7, 8};  // Relay pin slots
bool relayStates[] = {false, false, false, false};  // Track relay states

void setup() {
  Serial.begin(9600);  // Start serial communication
  for (int i = 0; i < 4; i++) {
    pinMode(relayPins[i], OUTPUT);
    digitalWrite(relayPins[i], LOW);  // Ensure all relays start OFF
  }
}

void loop() {
  if (Serial.available()) {
    char command = Serial.read();  // Read user input

    if (command >= '1' && command <= '4') {
      int index = command - '1';  // Convert char to index (0-3)
      relayStates[index] = !relayStates[index];  // Toggle state
      digitalWrite(relayPins[index], relayStates[index] ? HIGH : LOW);
      Serial.print("Relay ");
      Serial.print(relayPins[index]);
      Serial.println(relayStates[index] ? " ON" : " OFF");
    }
```

```
    else if (command == '0') {
      for (int i = 0; i < 4; i++) {
        relayStates[i] = true;
        digitalWrite(relayPins[i], HIGH);
      }
      Serial.println("All relays ON");
    }
    else if (command == '5') {
      for (int i = 0; i < 4; i++) {
        relayStates[i] = false;
        digitalWrite(relayPins[i], LOW);
      }
      Serial.println("All relays OFF");
    }
  }
}
```

5.1.1 Code Description and Explanation:

- **Constants and Variables:**
    - const int relayPins[] = {12, 13, 7, 8};: This line declares a constant array named relayPins and initializes it with the pin numbers to which the relays are connected on the Arduino. const means that these values cannot be changed during the program's execution.
    - bool relayStates[] = {false, false, false, false};: This line declares an array named relayStates of type bool (boolean) to store the current state of each relay (either true for ON or false for OFF). It is initialized with all relays set to false (OFF).
- **void setup() Function:**
    - Serial.begin(9600);: This line initializes serial communication at a baud rate of 9600 bits per second. Serial communication is used to receive commands from the Bluetooth module (or a computer via the serial monitor).
    - The for loop iterates through the relayPins array:
        - pinMode(relayPins[i], OUTPUT);: This sets the pin mode of each relay pin to OUTPUT, meaning the Arduino will send signals to these pins to control the relays.
        - digitalWrite(relayPins[i], LOW);: This line sets the initial state of each relay to OFF. It assumes that the relay module is active-LOW, which is a common configuration where a LOW signal turns the relay ON, and a HIGH signal turns it OFF.
- **void loop() Function:**
    - if (Serial.available()): This checks if there is any data available to be read from the serial port.
    - char command = Serial.read();: If data is available, this line reads the first character from the serial input and stores it in the command variable.
    - if (command >= '1' && command <= '4'): This if statement checks if the received character is between '1' and '4'. These characters are used to control individual relays.
        - int index = command - '1';: This line converts the character command ('1', '2', '3', or '4') to a corresponding array index (0, 1, 2,

or 3). For example, if command is '1', then index will be 0.

- relayStates[index] = !relayStates[index];: This line toggles the state of the selected relay. The ! operator inverts the current state. If the relay was OFF (false), it becomes ON (true), and vice versa.
- digitalWrite(relayPins[index], relayStates[index] ? HIGH : LOW);: This line sets the output of the relay pin to either HIGH or LOW based on the new relayStates[index]. This controls the actual relay. The ? : is the ternary operator, a shorthand for an if-else statement.
- Serial.print("Relay "); Serial.print(relayPins[index]); Serial.println(relayStates[index] ? " ON" : " OFF");: These lines print the relay pin number and its new state to the serial monitor for debugging and feedback.

- else if (command == '0'): This else if block checks if the received character is '0'. This command is used to turn all relays ON.
  - The for loop iterates through all four relays.
  - relayStates[i] = true;: Sets the state of all relays to ON.
  - digitalWrite(relayPins[i], HIGH);: Turns all relays ON.
  - Serial.println("All relays ON");: Prints a message to the serial monitor.
- else if (command == '5'): This else if block checks if the received character is '5'. This command is used to turn all relays OFF.
  - The for loop iterates through all four relays.
  - relayStates[i] = false;: Sets the state of all relays to OFF.
  - digitalWrite(relayPins[i], LOW);: Turns all relays OFF.
  - Serial.println("All relays OFF");: Prints a message.
- The code within the loop() function continuously checks for serial input, reads commands, and updates the relay states accordingly

5.1.2 Code Optimization

- The code can be optimized by using arrays to store the relay pins and commands, making it more concise and easier to expand.
- Error handling can be improved by adding more robust checks for invalid commands.
- The code can be made more modular by using functions to control each relay.

5.2 Android Application Development

The Android application provides the user interface for controlling the home automation system. It handles Bluetooth connectivity, sends commands to the Arduino Uno, and displays the status of the connected appliances.

5.2.1 User Interface Design

The user interface should be simple and intuitive. It can include:

- Buttons for each appliance (e.g., "Light 1 On," "Light 1 Off").
- A status display to show whether each appliance is on or off.
- A Bluetooth connection status indicator.
- A settings menu to configure the Bluetooth connection.

5.2.2 Bluetooth Connectivity

The application needs to include functionality to:

1. Request Bluetooth permissions.
2. Discover and list available Bluetooth devices.
3. Connect to the HC05 Bluetooth module.
4. Send data to the HC05 module.
5. Handle Bluetooth connection errors.

5.2.3 Controlling the Relays

When a user presses a button in the Android application, the application should send a corresponding command to the Arduino Uno via Bluetooth. For example, pressing the "Light 1 On" button might send the character 'a' to the Arduino. The Arduino code then interprets this character and turns on the corresponding relay.

5.2.4 Using "Arduino Bluetooth Controller" App

For this project, instead of developing a custom Android application, you can use a readily available app from the Google Play Store: "Arduino Bluetooth Controller." This app simplifies the process of sending commands to the Arduino via Bluetooth.

To use this app:

1. Download and install "Arduino Bluetooth Controller" from the Google Play Store.
2. Pair your Android device with the HC05 Bluetooth module.
3. Open the app and connect to the paired HC05 module.
4. Configure the app's buttons to send specific commands (e.g., 'a', 'b', 'c', 'd') that correspond to the commands expected by the Arduino code. The app typically allows you to assign a character or string to each button.
5. Use the buttons in the app to send the commands to the Arduino, which will then control the relays.

This approach eliminates the need for developing a custom Android application, making the project setup faster and easier. However, it requires the user to configure the "Arduino Bluetooth Controller" app according to the specific commands defined in the Arduino code.

Chapter 6: Testing and Results

6.1 Hardware Testing

6.1.1 Testing the Arduino Uno

- Verify that the Arduino Uno is receiving power and that the power LED is lit.
- Upload a simple "Hello World" sketch to the Arduino and check if it is displayed on the serial monitor.
- Test the digital output pins by writing a simple sketch to turn an LED on and off.
- Test the digital input pins.

6.1.2 Testing the HC05 Bluetooth Module

- Connect the HC05 module to the Arduino Uno and power it on.
- Use a smartphone to search for and pair with the HC05 module.
- Use a serial communication app on the smartphone to send data to the HC05 and verify that it is received by the Arduino Uno.
- Test the communication between the HC-05 and Android phone.

6.1.3 Testing the Relay Module

- Connect the relay module to the Arduino Uno.
- Write a simple sketch to turn the relays on and off and verify that the relays are working correctly (you should hear a clicking sound).
- **Caution:** When testing with AC power, exercise extreme caution. Ensure that all connections are correct and that you are not touching any exposed wires.
- Connect low-voltage DC loads (like LEDs) to the relays before connecting any AC devices.
- Use a Multimeter to check the voltage across the relay contacts when they are open and closed.

6.1.4 System Integration Testing

- Connect all the hardware components together as per the circuit diagram.
- Upload the Arduino code to the Arduino Uno.
- Install the Android application on a smartphone and connect to the HC05 module.
- Test all the functions of the system, including turning the appliances on and off, and verify that everything is working as expected.

6.2 Software Testing

6.2.1 Arduino Code Testing

- Use the serial monitor in the Arduino IDE to debug the code and check for any errors.
- Test different input commands and verify that the Arduino is controlling the relays correctly.
- Check for any logical errors in the code.

6.2.2 Android Application Testing

- Test the user interface to ensure that it is user-friendly and easy to navigate.
- Test the Bluetooth connectivity to ensure that the application can connect to the HC05 module reliably.
- Test the command transmission to ensure that the application is sending the correct commands to the Arduino Uno.
- Test the application on different Android devices to ensure compatibility.
- Perform user testing to get feedback on the application's usability.

6.3 Results and Discussion

- Summarize the results of the testing.
- Discuss the performance of the system, including its reliability, response time, and range.
- Analyze any problems encountered during the development and testing process and how they were resolved.
- Compare the system's performance with the objectives outlined in Chapter 1.

6.4 Error Handling

- Describe the error handling mechanisms implemented in the system.
- For example, how the system handles invalid commands from the Android application or a lost Bluetooth connection.
- Discuss any potential issues that could arise and how they can be prevented or mitigated.

Chapter 7: Conclusion and Future Scope

7.1 Conclusion

- Summarize the project and its main achievements.
- State whether the project objectives were met.
- Highlight the key features and benefits of the developed home automation system.
- Emphasize the potential of the system for future expansion and customization.

7.2 Future Scope

- Discuss potential improvements and future developments for the system, such as:
    - Adding support for more appliances and devices.
    - Implementing more advanced control features, such as scheduling, dimming, and fan speed control.
    - Integrating sensors for temperature, light, and motion detection.
    - Adding support for voice control using a voice assistant like Google Assistant or Amazon Alexa.
    - Developing a web interface for remote access and control over the internet.
    - Implementing energy monitoring and management features.
    - Enhancing the security of the system with encryption and authentication.
    - Using a more robust communication protocol like Wi-Fi or Zigbee for increased range and reliability.

7.3 Limitations

- Acknowledge any limitations of the current system, such as:
    - Limited range due to the use of Bluetooth technology.
    - Limited number of controllable devices.
    - Lackof advanced features.
    - Potential security vulnerabilities.

Chapter 8: Appendices

8.1 Arduino Code

```
const int relayPins[] = {12, 13, 7, 8};  // Relay pin slots
bool relayStates[] = {false, false, false, false};  // Track relay states

void setup() {
    Serial.begin(9600);  // Start serial communication
    for (int i = 0; i < 4; i++) {
        pinMode(relayPins[i], OUTPUT);
        digitalWrite(relayPins[i], LOW);  // Ensure all relays start OFF
    }
}

void loop() {
    if (Serial.available()) {
        char command = Serial.read();  // Read user input

        if (command >= '1' && command <= '4') {
            int index = command - '1';  // Convert char to index (0-3)
            relayStates[index] = !relayStates[index];  // Toggle state
            digitalWrite(relayPins[index], relayStates[index] ? HIGH : LOW);
            Serial.print("Relay ");
```

```
            Serial.print(relayPins[index]);
            Serial.println(relayStates[index] ? " ON" : " OFF");
        }
        else if (command == '0') {
            for (int i = 0; i < 4; i++) {
                relayStates[i] = true;
                digitalWrite(relayPins[i], HIGH);
            }
            Serial.println("All relays ON");
        }
        else if (command == '5') {
            for (int i = 0; i < 4; i++) {
                relayStates[i] = false;
                digitalWrite(relayPins[i], LOW);
            }
            Serial.println("All relays OFF");
        }
    }
}
```

8.2 Android Code (Partial)

[Key snippets of the Android code, focusing on Bluetooth connectivity and command transmission]

8.2 Bill of Materials

Component	Quantity	Description
Arduino Uno	1	Microcontroller board
HC05 Bluetooth Module	1	Bluetooth serial communication module
4-Channel Relay Module	1	Module with four relays to control electrical appliances
5V Power Supply	1	Provides power to the Arduino, HC05, and relay module
Jumper Wires	20	Wires for connecting components
Electrical Appliances	4	Devices to be controlled (e.g., lights, fans)
PCB (Optional)	1	Printed circuit board for mounting components
Enclosure (Optional)	1	Housing for the system

Chapter 9: References

-