**SMART HOME AUTOMATION**

**A PROJECT REPORT**

***Submitted by***

**Aman Rana (21BCS7389)**

**Shubham (21BCS8726)**

**Vivek Kumar Dubey (21BCS8373)**

**B. Venkata Kumara (21BCS5843)**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

###### IN

COMPUTER SCIENCE WITH SPECIALIZATION IN

BIG DATA ANALYTICS



**Chandigarh University**

November 2024



**BONAFIDE CERTIFICATE**

Certified that this project report **“Smart Home Automation”** is the bonafide work of “**Aman Rana, Shubham, Vivek Kumar Dubey, B. Venkata Kumara”** who carried out the project work under my/our supervision.

**SIGNATURE**

Mr Aman Kaushik (HOD)

**HEAD OF THE DEPARTMENT**

AIT CSE

**SIGNATURE**

Jayashree Mohanty (Asst. Professor)

**SUPERVISOR**

AIT CSE

Submitted for the project viva-voce examination held on

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**TABLE OF CONTENTS**

[**List of Figures**](#_26in1rg)**………………………….………………………………………………………...**[**i**](#_26in1rg)**v**

[**List of Tables**](#_lnxbz9)**………………………….……………………..…………………………………...****v**

[**Abstract**](#_35nkun2)**………………………….……………………………………………………….….…..****vi**

**Graphical Abstract………………………….………………………………………….…........****vii**

[**Abbreviations**](#_lnxbz9)**………………………….………………………………………..……………...viii**

[**Chapter**](#_1ksv4uv) **1. INTRODUCTION………………………….…………………………………….....****1**

[**1.1**](#_2jxsxqh) **Client & Need Identification………………………….……………………………………. 1**

[**1.2**](#_z337ya) **Problem Identification………………………….…………………………….……………..2**

[**1.3**](#_1y810tw) **Tasks Identification………………………….…………………………….……………...…3**

[**1.4**](#_4i7ojhp) **Hardware & Software Specification………………………….…………………………….3**

**1.5 Organization of Report……………………………………………………………………...4**

[**Chapter 2**](#_1ci93xb)**. LITERATURE REVIEW………………………….……………..………………...5**

[**2.1 Existing Systems & Associated**](#_2bn6wsx) **Technologies………………………………………….…..5**

[**2.2**](#_qsh70q) **Limitations of Existing Systems………………………….…………………………………6**

[**2.3 Proposed System**](#_qsh70q)**………………………….………………………….……………………..13**

[**2.4 Advantages of Proposed System**](#_qsh70q)**………………………….……………….……………….14**

[**2.5 Pro**](#_qsh70q)**blem Definition………………………….………………………….…………………...17**

[**2.**](#_qsh70q)**6 Challenges………………………….………………………….…………..…….,................18**

[**2.7**](#_qsh70q)**Goals/Objectives…………………………………………………………………………….19**

[**Chapter 3**](#_3as4poj)**. DESIGN FLOW………………………………………………….………………..22**

**3.1 Evaluation & Selection of Features………………………………………………….…….22**

**3.2 Design Constraints………………………….………………………………………………22**

**3.3 Analysis and Feature Finalization………………………….…………….………….…….23**

**3.4 Implementation………………………….……………………………….…………………23**

**Chapter 4. RESULT ANALYSIS………………………….………………….……………….51**

**4.1 Result Analysis……………………………………………………………………………...51**

**Chapter 5. CONCLUSION AND FUTURE WORK………..…………….…………………55**

**5.1 Conclusion…………………………………………………………………………………..55**

**5.2 Future Work………………………………………………………………………………..55**

[**References……………………………………………………………………………………….57**](#_49x2ik5)

**List of Figures**

**Figure 1 Graphical Abstract …………………………………………………………….vii**

**Figure 2.3.1 Proposed System Design …………………………………………………..16**

**Figure 3.4.1 Prototype Design ……………………………………………………………29**

**Figure 3.4.2 List of components ………………………………………………………….29**

**Figure 3.4.2 List of components ………………………………………………………….30**

**Figure 3.4.4 WEB UI Dashboard …………………………………………………….…..39**

**Figure 3.4.5 Simulation when one light is ON …………………………………………..40**

**Figure 3.4.6 Simulation when both lights are ON ………………………………………41**

**Figure 3.4.7 Simulation when both lights are OFF……………………………………...42**

**Figure 3.4.8 Button Usage Description ………………………………………………….43**

**Figure 4.1 Result Case1……………………………………………………………….…..51**

**Figure 4.2 Result Case2…………………………………………………………………...52**

**Figure 4.3 Result Case5…………………………………………………………………...53**

**Figure 4.4 Result Case4…………………………………………………………………...53**

**Figure 4.5 Result Case5…………………………………………………………………...54**

**List of Tables**

**Table 2.1.1 Existing Systems Table …………………………………………………………10**

**Table 2.1.2 Literature Review Summary Table ……………………………………………11**

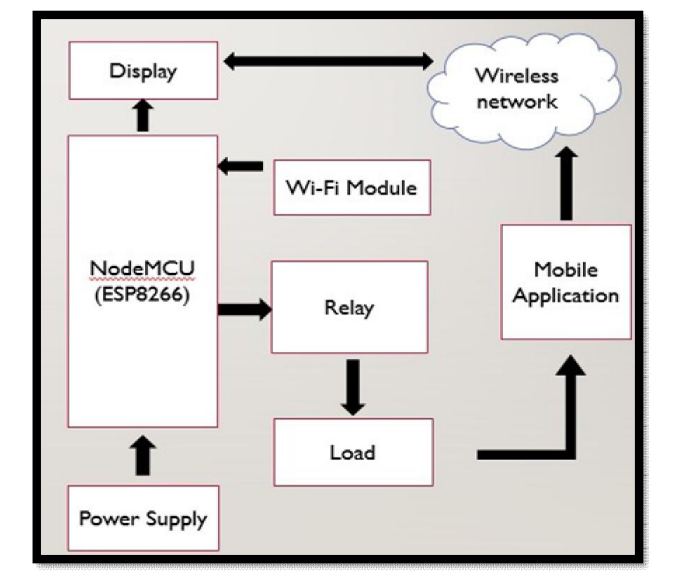
**ABSTRACT**

This paper introduces the smart home automation system based on IoT technology with the primary objectives of convenience, energy efficiency, and access from anywhere. It begins by writing a large literature survey on existing smart home automation systems, technologies, and frameworks, highlighting their strengths, weaknesses, and areas that are still lacking in functionality. Most solutions depend on Wi-Fi-based control, Bluetooth, or proprietary communication protocols, which only manage to limit scalability, energy optimization, or remote access.

The Particle Photon microcontroller enables the integration of environmental sensors with a cloud platform and remotely controlled appliances. Main components comprise a DHT11 sensor for temperature and humidity monitoring, an LDR sensor for light intensity measurement, and a relay circuit that switches lights and fans on/off at home. The system will capture real-time data, process it locally, and communicate with a cloud platform so that a web interface enables remote monitoring and control. Users can monitor environment parameters, as well as operate appliances, from virtually any internet-enabled device. The system also supports automatic appliance control by triggering particular actions based on sensor inputs, for example, by turning lights on when the ambient light intensity drops.

This research will outline the bridges of the gaps of the present systems by giving an energy-efficient, scalable, and remotely accessible solution through IoT-based smart home automation. This paper concludes that the future improvements in energy management, security, and user convenience through the combination of cloud services with intelligent sensors and web-based control can develop much augmentation within the framework.

**GRAPHICAL ABSTRACT**

****

**Figure 1 Graphical Abstract**

**ABBREVIATIONS**

MQTT Message Queuing Telemetry Transport

SHA Smart Home Automation

BLE Bluetooth Low Energy

EIB European Installation Bus

PoE Power Over Ethernet

IoT Internet of Things

LDR Light Dependent Resistor

DHT11 Digital Temperature and Humidity Sensor

HVAC Heating Ventilation & Air conditioning

UART Universal Asynchronous Receiver Transmitter

MCU MicroController Unit

# INTRODUCTION

Generally, this is referred to as a smart home automation system that is basically any house fitted with IoT-enabled devices, which facilitate the remote monitoring, control, and automation of a household's lighting, heating, security, and appliances. These devices rely on the internet to transfer information and enable seamless interaction between users and their home - it could be with a smartphone or with an Internet interface. Smart homes provide comfort, conserve energy, and prove secure when automation occurs with respect to environmental data or pre-set conditions. Here, a paper is proposed designing and implementing smart home automation by using sensors, cloud platforms, and relay circuits. This also enables real-time monitoring and remote control with functionalities that current systems lack.

## Client & Need Identification

Smart home automation systems are rapidly evolving as a major need for diverse client groups, including homeowners, property managers, energy-conscious individuals, and those seeking enhanced home security and assisted living solutions. Residential consumers increasingly look for ways to make their homes more convenient, secure, and energy-efficient. Property managers and real estate developers are leveraging smart home features to differentiate properties in competitive markets, while elderly individuals or those in assisted living scenarios benefit from these systems by gaining more control over their environments and reducing reliance on in-person assistance.

The demand for smart home automation is strongly backed by current statistics and industry documentation, showcasing its growing relevance. According to Statista, the smart home market is projected to grow by 25% annually, potentially reaching around $200 billion by 2026. A major driver of this adoption is the pursuit of energy savings. The U.S. Department of Energy has documented that smart thermostats alone can save homeowners between 10-15% on heating and cooling costs. Additionally, the security concerns of homeowners have escalated, with FBI data from 2022 reporting an increase in residential property crimes. Smart security systems—like locks, cameras, and alarms—are increasingly sought after for their capacity to provide peace of mind and enhanced protection.

Despite the growing popularity, many clients face difficulties that prevent them from achieving fully integrated home automation. The incompatibility of devices across different brands or protocols, complex installation processes, and cybersecurity concerns have led many homeowners to seek professional consultation. For instance, a 2022 survey conducted by the Consumer Technology Association found that while 60% of smart home device users preferred integrated systems, many struggled with setup challenges and connectivity issues. Additionally, data privacy concerns have been consistently highlighted by advocacy groups like the Electronic Frontier Foundation, reflecting a broader need for cybersecurity guidance in the face of expanding home networks.

Several contemporary issues documented by key agencies further underscore the importance of smart home technology. The International Energy Agency, for example, has recognized smart home systems as instrumental in reducing household energy consumption, which is essential to meeting global sustainability targets. Meanwhile, the World Health Organization reports that as the global population ages, there is an increased demand for accessible, user-friendly home automation systems that can support independence for elderly individuals.

In conclusion, the need for smart home automation systems is not only a growing trend but a pressing issue backed by consumer demand, sustainability goals, and security concerns. With professional consultancy, these challenges can be addressed, enabling homeowners to experience the full benefits of a well-designed, integrated smart home.

## Problem Identification

Today's busy world makes the operation of homes difficult and more challenging in terms of management and security. In the conventional way of running homes, appliances have to be operated manually; this is not only a time-consuming activity but is also a process susceptible to human error, besides leading to waste energy consumption. Besides, the security of the home has to be maintained constantly, with constant vigilance especially when the owner is away.

**Key Problems:**

1. Operating Devices by Hands: Devices for lights, thermostats, and entertainment units are some of the household equipment that often need to be operated by hands, which result in inefficiencies and inconveniences.

2. Waste of Energy: The reason why this solution consumes much more utility costs and wastes more in terms of energy expenditure is that it cannot be tracked and optimized in real time without the means of automation.

3. Security Risks: Homes are vulnerable to entry, etc. because traditional security systems are not monitored continuously and alerts and remote access are not always available.

4. Lack of Integration: Rather than having an integrated smart home experience, most homes are filled with numerous products from various companies that do things on their own, which results in fragmented control and management.

5. Limited Remote Access: Home owners may often fail to remotely check or operate such systems, which could be challenging as well as risky when they are away from home.

## Tasks Identification

1. Client Needs and Market Analysis: It involves examining client demographics, preferences, and primary motivations for adopting smart home technology. Identify specific pain points and needs. Conducting a market analysis to understand consumer demand, assess target audience segments, and evaluate competitor offerings.

2. Requirements Definition: It involves gathering and documenting detailed system requirements through surveys, interviews, or focus groups with potential users. Capturing both functional requirements (such as automation features and control interfaces) and non-functional requirements (such as ease of use, and accessibility).

3. Feasibility and Viability Study: It involves assessing the feasibility of implementing a smart home system from technical, economic, and operational perspectives, regulatory factors (like data privacy laws) and compatibility with existing systems in the industry to ensure compliance and practical viability.

4. Solution Design and Architectural Framework: It involves developing the system’s design specifications and overall architecture. Plan the integration of devices and establish standards for compatibility, data flow, and user interfaces. Outline software and hardware specifications needed to achieve seamless operation.

5. System Development and Integration: It involves implementing the designed system by developing the required software components, integrating selected devices, and establishing connectivity protocols. Ensuring all parts work together cohesively and can be controlled via a central hub or apps.

6. Testing, Validation, and Optimization: It involves conducting rigorous testing for functionality, security, and usability to ensure the system meets client requirements. Validating all components under real-life conditions, refine based on feedback, and optimize for performance and reliability.

## Hardware & Software Specification

* Central Hub/Controller
* Particle Photon Board
* Arduino Board
* Smart Devices
* Sensors and Actuators
* Networking Equipment
* Operating System
* Cloud Services
* Tinkercad
* Home Automation Platform
* Blender and Smart Home I/O

## Organization of Report

This report is structured into five comprehensive chapters, each addressing critical aspects of the project. Chapter 1 introduces the project, detailing the client’s needs, identifying the problem, and outlining the tasks involved, as well as providing the hardware and software specifications necessary for the proposed solution. This chapter concludes with an overview of the report's structure. Chapter 2 provides a literature review, examining existing systems and technologies, identifying their limitations, and introducing the proposed system, including its advantages, defined problems, challenges, and objectives. Chapter 3 focuses on the design flow, covering the evaluation and selection of features, design constraints, analysis and finalization of features, and the detailed implementation process.Chapter 4 analyzes the results obtained from the project, providing insights into the performance and functionality of the system. Finally, Chapter 5 presents the conclusion of the project, summarizing the findings and suggesting potential areas for future work. This organized structure enables a logical flow, guiding the reader from problem identification to solution implementation and evaluation.

# LITERATURE REVIEW

## Existing Systems & Associated Technologies

To gain a comprehensive understanding of smart home automation, it is essential to examine the breadth of research and development in this field, knowing both the advancements and limitations of existing systems. This thorough literature review not only reveals the current state of IoT-based smart home technologies but also highlights the innovations, frameworks, and methodologies that have contributed to their evolution. Reviewing existing literature enables us to identify gaps in the knowledge base, recognize patterns in user requirements and technological preferences, and uncover the challenges that persist in terms of security, compatibility, and scalability. This foundational analysis is crucial as it allows for a more informed exploration of how smart home automation has been shaped by IoT and offers direction for future advancements and solutions in this rapidly evolving domain. Here’s a detailed overview to some key technologies:

**A. Bluetooth-Based Smart Home Automations**

Bluetooth technology operates on a short-range communication protocol, typically covering a radius of around 10 to 100 meters depending on the version, power class, and obstacles present. It is energy-efficient and cost-effective, making it a popular choice for applications that require low power consumption.

* Bluetooth Low Energy (BLE):

A significant advantage of Bluetooth in smart homes is the use of Bluetooth Low Energy (BLE). BLE consumes far less power than classic Bluetooth, making it ideal for battery-powered devices such as door locks, smart lights, and sensors.

* Mesh Networking:

Newer versions of Bluetooth support mesh networking, enabling multiple devices to communicate across greater distances by passing data through a chain of devices. This feature expands the application of Bluetooth in smart home systems by improving range and coverage.

* Centralized Hubs and Mobile Apps:

Many Bluetooth-based systems require a centralized hub or gateway to communicate with other smart devices and relay data to a mobile app, where users can control various appliances.

However, Bluetooth-based systems typically have limited compatibility with non-Bluetooth smart home devices, so hybrid setups that incorporate other protocols may be necessary for a fully connected system [1].

### B. ZigBee and Ethernet-Based Smart Home Automation

ZigBee is a wireless communication protocol known for its low power consumption and affordability. It operates in the 2.4 GHz frequency range, similar to Wi-Fi, but is optimized for low data transfer and short-range connectivity, which is ideal for IoT and smart home devices.

* Scalability and Reliability:

ZigBee is known for its scalability; it can support a large number of devices within a network (up to 65,000 nodes), making it an excellent choice for homes with many smart devices. Its mesh network structure allows for stable connections by enabling devices to relay signals through one another.

* ZigBee and Ethernet Hybrid Systems:

Combining ZigBee with Ethernet creates a hybrid network that provides both wireless flexibility and the robustness of wired connections. ZigBee handles low-power, localized device control, while Ethernet offers high-speed and reliable data transfer for critical systems. Hybrid systems often use an Arduino-based controller to process data and provide control between ZigBee devices and Ethernet-connected systems.

* Interference and Security:

ZigBee is less susceptible to interference than Wi-Fi, making it reliable in dense environments. Its built-in security features, like encryption and authentication, make it secure for sensitive smart home applications [2].

### C. Serial EIB-Based Smart Home Automation

Serial EIB (European Installation Bus), also known as KNX (Konnex), and X10 are among the earliest home automation communication protocols, designed primarily for wired control over home appliances.

* X10 Protocol:

X10 is one of the simplest and oldest protocols, relying on the existing powerline infrastructure to transmit signals. While it’s limited in data speed and scalability, it’s still functional for simple smart home setups, such as controlling lights, basic switches, and sensors.

* EIB/KNX:

The KNX standard is more sophisticated and scalable than X10, designed to be a universal communication protocol that supports various home automation applications. While it typically requires professional installation and can be costly, it’s widely adopted in Europe and known for its interoperability across different vendors’ products. KNX is ideal for large smart homes or commercial buildings with more extensive and advanced automation requirements.

* Installation Complexity:

Although these systems are effective, they may require professional setup and configuration. The wiring needs careful planning, and modifications can be challenging, making these systems less flexible than newer wireless protocols [3].

### D. Wi-Fi-Based Smart Home Automation

Wi-Fi is one of the most common wireless communication protocols in modern smart home automation due to its extensive range, high-speed data transfer capabilities, and widespread compatibility.

* High Speed and Data Capacity:

Wi-Fi provides fast data transmission, which is critical for data-heavy applications like live video feeds from security cameras or real-time data processing in smart thermostats and multimedia systems.

* Controller Options:

Devices like Raspberry Pi and NodeMCU are popular controllers for Wi-Fi-based smart home systems. The Raspberry Pi serves as a robust platform for running complex automation scripts and integrating with cloud services. NodeMCU, based on the ESP8266 or ESP32 microcontrollers, provides a more lightweight, low-cost solution for smaller devices.

* Accessibility and Compatibility:

Wi-Fi offers easy compatibility with mobile apps, voice assistants, and cloud platforms, making it a convenient choice for comprehensive smart home automation. However, Wi-Fi can consume more power and suffer interference, especially in dense wireless environments [4].

### E. PC Server-Based Smart Home Automation

A PC server can act as a centralized control unit for smart homes, offering robust processing power, storage, and integration capabilities. PC servers possess large capability to handle complex automation tasks. These can include, for instance, integration with other systems such as HVAC or security systems.

* Complex Automation Tasks:

A PC server can manage complex automation tasks, including advanced scheduling, conditional logic, and integration with third-party systems such as HVAC, security, and lighting systems.

* Data Storage and Analytics:

With a dedicated PC server, users can store data locally, perform data analytics, and create advanced logs for home usage patterns. This data can be used to optimize energy consumption and predict maintenance needs.

* Remote Access:

PC servers can be connected to the internet, enabling remote access through web and mobile applications, giving homeowners the ability to control their smart home from anywhere [5].

### 

### F. Smart Home Automation Using Arduino ESP8266-Based

The Arduino ESP8266 is a popular microcontroller for IoT applications due to its low cost, compact design, and built-in Wi-Fi capabilities.

* Efficient Wi-Fi Module:

The ESP8266 integrates Wi-Fi capabilities into a small, affordable module, making it ideal for connecting simple devices such as lights, temperature sensors, and small appliances to the internet.

* Multiple Protocol Support:

The ESP8266 can interface with various communication protocols, such as I2C, SPI, and UART, allowing it to connect with a wide range of sensors and actuators, and control them remotely through mobile apps.

* As it’s optimized for low power consumption, the ESP8266 is suitable for battery-powered devices that need to operate over long periods without constant charging or replacement [6].

### G. ZigBee and Z-Wave-Based Smart Home Automation

ZigBee and Z-Wave are two low-power wireless communication protocols designed for smart home applications, providing reliable, secure connections with low energy requirements.

* Energy Efficiency:

Both ZigBee and Z-Wave consume minimal power, making them suitable for battery-operated devices, such as motion detectors, door locks, and security sensors, that need to operate for extended periods.

* Security and Range:

These protocols offer secure, reliable connections. ZigBee has a range of about 10-20 meters indoors, while Z-Wave can reach up to 30 meters. Both protocols use mesh networking, so devices can relay information to extend coverage [7].

### H. Smart Home Automation Over Ethernet

Ethernet is a wired communication technology known for its high speed, reliability, and secure connectivity.

* Reliable Connectivity:

Ethernet is less susceptible to interference than wireless solutions, making it ideal for systems requiring consistent high-speed data transfer, such as streaming devices, servers, or security cameras.

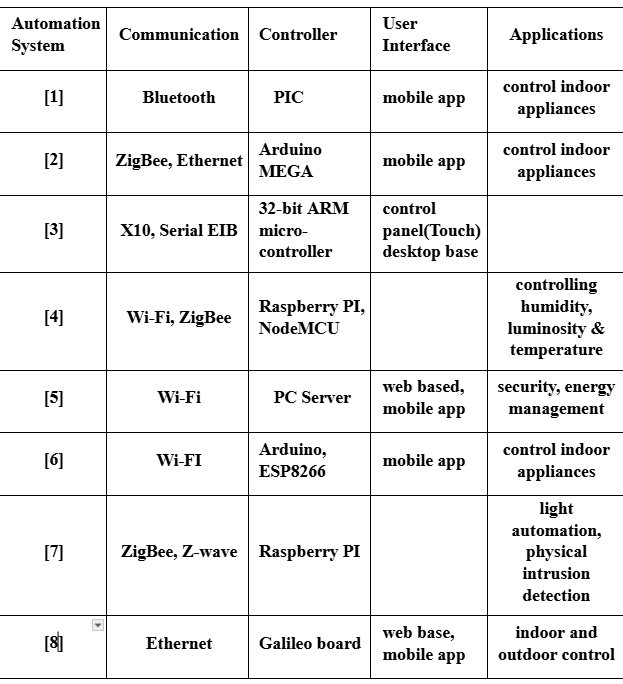
* Scalability and Power Over Ethernet (PoE):

Ethernet allows devices to draw power from the same cable that carries data, eliminating the need for separate power sources. This feature, known as PoE, is useful for devices like IP cameras and sensors in locations with limited power access.

* Hybrid Configurations:

Ethernet can be combined with wireless networks, creating a robust, flexible smart home ecosystem where high-priority systems use Ethernet, while less critical devices rely on wireless protocols [8].

Table below summarizes several smart home automation systems and their key components, communications protocols, controllers, user interfaces, and applications.



**Table 2.1.1 Existing Systems Summary Table**

## Literature Review Summary

| **Year and**  **Citation** | **Article/ Author** | **Tools/ Software** | **Technique** | **Source** | **Evaluation Parameter** |
| --- | --- | --- | --- | --- | --- |
| 2021 , Zhang, L., & Wang, Y. (2021). *Intelligent Smart Home Automation System Using IoT and Machine Learning*. Journal of Ambient Intelligence and Smart Environments, 13(2), 123-135. | Zhang, L., & Wang, Y. | Arduino, Raspberry Pi, TensorFlow | Machine Learning Algorithms for Predictive Automation | Journal of Ambient Intelligence and Smart Environments | Accuracy of automation predictions, system responsiveness |
| 2022, Kumar, S., Gupta, R., & Singh, M. (2022). *Secure Smart Home Automation Using Blockchain Technology*. IEEE Internet of Things Journal, 9(4), 2501-2512. | Kumar, S., Gupta, R., & Singh, M. | Hyperledger Fabric, MQTT | Blockchain for Security and Data Integrity | IEEE Internet of Things Journal | Security breach incidents, transaction latency |
| 2023, Martinez, A., & Lee, H. (2023). *Energy-Efficient Smart Home Systems: A Comprehensive Review*. Energy and Buildings, 256, 112456. | Martinez, A., & Lee, H. | MATLAB, Simulink | Energy Optimization Techniques | Energy and Buildings | Energy consumption reduction, cost savings |
| 2020 , Nguyen, T., & Park, J. (2020). *Voice-Controlled Smart Home Automation Using Natural Language Processing*. Computers & Electrical Engineering, 86, 106694. | Nguyen, T., & Park, J. | Amazon Alexa SDK, Python NLTK | Natural Language Processing for Voice Commands | Computers & Electrical Engineering | Command recognition accuracy, user satisfaction |
| 2023,  Silva, M., Fernandes, P., & Costa, D. (2023). *Integration of Smart Home Device Using Zigbee Protocol*. Sensors, 23(5), 2101. | Silva, M., Fernandes, P., & Costa, D. | Zigbee, Node-RED | Wireless Communication Protocols | Sensors | Network reliability, device interoperability |

**Table 2.1.2 Literature Review Summary Table**

## Limitations of Existing Systems

Existing smart home automation systems, while advanced, face several limitations and issues that impact their effectiveness and user experience.

Here’s an overview of limitation in existing systems:

1. Lack of Interoperability: Devices from different manufacturers often do not work together seamlessly, limiting user flexibility.

2. Privacy Concerns: Many smart systems rely on cloud processing, which can expose sensitive user data to third-party servers.

3. Security Vulnerabilities: Without robust security measures, smart devices can be vulnerable to hacking and unauthorized access.

4. Dependence on Internet Connectivity: A stable internet connection is often required, and disruptions can lead to system downtime.

5. High Power Consumption: Continuous monitoring for voice or remote commands increases energy use, reducing system efficiency.

6. Complex Installation: Many systems require complex setups or professional installation, making it harder for average users.

7. High Initial Costs: Quality smart home systems can be costly, limiting accessibility for budget-conscious users.

8. Limited Compatibility with Legacy Appliances: Many systems may not integrate with older appliances, reducing usability.

9. Network Latency: Cloud-dependent systems may experience delays, which can affect the responsiveness of devices.

10. Scalability Challenges: Expanding a smart home system with additional devices can become costly and may lead to network congestion or interference, affecting performance.

## Proposed System

Proposed system aims to overcome the limitations of existing systems listed above. Proposed system of smart home automation system consists of the following key elements:

**1. Power Supply and Voltage Regulation**

The power supply is the main source of energy for the system. In this design, an 18V DC power input will be utilized in order to ensure the availability of a reliable supply of energy. However, most elements of the system, including a microcontroller (Particle Photon) and sensors, have lower operating voltages. This ensures the safe and stable power supply to the control unit and sensors, to protect from overvoltage damage. The voltage regulator is used to step down the 18V supply to the output regulated to 5V.

**2. Control Unit – Particle Photon Microcontroller**

The core device that controls all of the whole setup is the microcontroller, Particle Photon. It's a microcontroller board with Wi-Fi on board. So, it relays data back to the cloud, meaning there's data exchange as well as remote control.

* **Data Acquisition:** The microcontroller extracts information from quite a number of sensors, such as the DHT11 and LDR sensor.
* **Processing and Control:** It'll process the information to make judgments, such as turning appliances on and off.
* **Cloud Communication:** It is transmitting the sensor data to a cloud platform; in addition, it is receiving the remote control commands from the users using the web interface.
* **Output Handling:** It is providing signals to the relay circuit to switch the appliance ON/OFF using the given logic or the commands provided through the cloud.

Particle Photon acts as the brain of the system; it incorporates local control and remote control through the cloud without any glitches.

**3. Sensors and Actuators**

All these inputs undergo processing by the control unit that drives the automation action.

* **DHT11 Sensor:** This sensor senses ambient temperature and humidity. The DHT11 gives out digital signals, which are fed to the microcontroller.
* **Functionality:** Web interface shall display ambient temperature and humidity to be monitored by users. These parameters can allow making automation rules like one of the fans will be turned on at the threshold of temperature.
* **LDR Sensor:** The LDR sensor works on ambient light; it can measure the intensity of the light surrounding it. It can detect changes in illumination level and sends input signals to the control unit in its prompt.
* **Functionality:** LDR can be used to control lighting circuitry. It can implement scheduled lighting systems. For example, when the light level falls below the threshold (sunset), the control unit can make the relay switch the lights ON.

**4. Relay Circuit**

In simple terms, a relay circuit essentially operates as an electronic switch, switching on the home appliances connected in the system. Its ON/OFF control signals are forwarded by a Particle Photon microcontroller, by which it toggles the state of connected appliances.

The relay circuit switches on devices such as lamps, fans, or any other electrical appliance. The relay switches the status of the device it is in contact with (ON or OFF) when it gets a signal from the control unit, both programmable based on sensor readings and manually through the web interface.

**5. Cloud Platform**

The cloud platform serves as a communication bridge between the control unit and the web interface. This stores sensor data and offers remote access for the user to monitor and control their house appliances.

* **Data Storage and Monitoring:** Real-time sensor data uploaded from the microcontroller allow users to monitor parameters such as temperature, humidity, and light intensity in the house.
* **Remote Access:** The cloud platform allows users to access the system from any location, using a web-based interface to control appliances remotely.

**6. Web Interface**

The web interface is a user-friendly interface for remote monitoring and control of the smart home system. The users can access it from any internet-enabled device, such as a smartphone or computer.

* **Sensor Data Visualization:** The web interface displays real-time sensor data, such as temperature, humidity, and light intensity, for users to monitor environmental conditions inside their homes.
* **Appliance Control:** Users can toggle appliances ON or OFF using control buttons provided on the interface. The web interface sends these commands to the control unit through the cloud platform, which then triggers the appropriate relay action.

**7. LEDs**

The LED indicators provide a visual representation of the state of connected appliances. These LEDs can signal whether a specific appliance is currently ON or OFF.

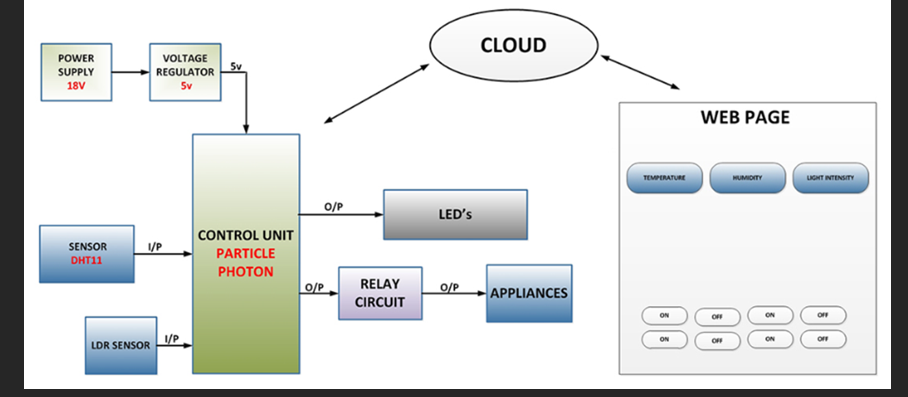
The indicators help users quickly determine the status of appliances without needing to check the web interface. For example, if an appliance such as a fan is ON, the corresponding LED will light up.

**8. Appliances**

The appliances that will be automated through the relay circuit of this system are controlled. These may include lights, fans, or any electrical device the user would want to automate.

Depending upon input from sensors or user commands via web interface, appliances get switched ON or OFF. For example, lights can automatically be switched on during night depending upon data from the LDR sensor or switched off remotely using the web interface.

Having thoroughly described each component in detail, the figure below shows architectural design of proposed system, which integrates these technologies into a cohesive and efficient smart home automation system:



**Figure 2.3.1 Proposed System Design**

## Advantages of Proposed Solution

1. **Real-Time Monitoring and Control**: The integration of a cloud-based platform allows users to monitor temperature, humidity, and light intensity in real time from a web page, enhancing remote management.
2. **User-Friendly Interface**: The web-based interface provides a simple way for users to control appliances, with dedicated buttons for switching appliances on and off, making it easy to operate.
3. **Energy Efficiency**: By using sensors like the DHT11 for temperature and humidity, and an LDR sensor for light intensity, the system can optimize energy consumption by adjusting appliance usage based on environmental conditions.
4. **Scalability**: The use of Particle Photon as the control unit offers flexibility for future expansion, allowing additional sensors or appliances to be added with minimal modification to the existing setup.
5. **Improved Device Safety**: With a voltage regulator, the system ensures that connected devices receive a stable 5V supply, protecting components from potential damage due to power fluctuations.
6. **Automated Appliance Control**: The relay circuit enables automated switching of appliances, improving convenience and reducing the need for manual operation.
7. **Cloud Integration**: Cloud connectivity allows for data storage, analysis, and remote control, which enhances functionality and enables features such as scheduling or predictive analytics based on past usage patterns.
8. **Cost-Effective**: The design utilizes widely available and low-cost components, making the system affordable for home automation.
9. **High Flexibility and Customization**: Users can modify the web page or add more sensors to customize the system for specific home automation needs.
10. **Enhanced Comfort and Convenience**: With the ability to remotely control home appliances and monitor conditions, the system significantly increases user comfort and ease of living.

## Problem Definition

The problem at hand involves the development of an efficient and secure smart home automation system that leverages IoT technologies to enhance the convenience, energy efficiency, and security of residential environments. The aim is to design a system that seamlessly integrates various interconnected devices, such as lighting, appliances, heating, and security systems, allowing homeowners to control and monitor their home remotely in real-time.

The development of proposed solution involves following steps:

* **Initial System Design and Architecture**: Developing the foundational architecture for the smart home system, selecting appropriate technologies and communication protocols (e.g., Wi-Fi, ZigBee, Bluetooth).
* **Centralized Control**: Developing a centralized platform or hub that allows users to control and monitor all devices through a mobile or web interface.
* **Energy Optimization**: Incorporating features that enable energy consumption monitoring and optimization to reduce costs and environmental impact.
* **Security and Privacy**: Ensuring that the system is secure from potential cyber threats and that user privacy is protected through encryption and secure data storage.

### How it is to be done:

* **Use of Established IoT Protocols**: Utilize common communication protocols such as Wi-Fi and Bluetooth to interconnect devices within the home.
* **Cloud and Local Data Processing**: Leverage cloud computing for advanced processing tasks, while also implementing local processing on edge devices (like Particle Photon Board) for quick, low-latency responses.
* **User Interface Development**: Create an intuitive mobile or web application that provides a simple interface for users to control devices, set schedules, and receive notifications.
* **Real-Time Monitoring**: Implement real-time monitoring features for users to track appliance usage, home security status, and environmental conditions (temperature, humidity, etc.)

## Challenges

While smart home automation systems offer many benefits, they also come with several problems that may delay their adoption in daily life and reduce the usability of such gadgets. Some of them include the following problems:

1. Interoperability: The lack of well-defined protocols among different manufacturers and devices may cause anomalies in compatibility. Users' constant challenge of integrating items from different brands limits the functionality of smart home systems in its entirety.

2. Security and privacy: With increasing connectivity, comes the threat of data breach and cyberattacks. For the developers, as well as for the users, comes a tough responsibility to ensure strong security measures for safeguarding private user information and prevent illegitimate access.

3. Connectivity and Reliability: Smart home devices require stable internet connectivity for functioning properly. Users can get frustrated by malfunctioning systems or operational restraint due to any break in the connectivity.

4. Complexity in application: Sometimes the smart home technology is complex and rather not very user-friendly for some customers. A high learning curve can also keep potential users from really taking advantage of all that their system has to offer.

5. Price: Smart devices and infrastructure require a high investment, which can be a discouraging factor for people to install them in their homes. Moreover, cost can be supplemented in the form of recurring maintenance and subscription service charges.

6. Data handling: Smart home devices generate masses of data that needs to be handled and analyzed for users and producers alike. Data accuracy, storage, and meaningful insight remain problematic for producers and users alike.

7. User Acceptance Awareness: The majority of consumers are still unaware of the variety of benefits smart home automation systems provide. The more knowledge and confidence the potential buyer comes to develop, the more uptake will be facilitated.

8. Scalability: It might well pose a challenge for developers to ensure that smart home systems scale smoothly without suffering from degradation in performance with an increase in connected devices.

## Goals/Objectives

1. **Initial System Design and Architecture**

* Develop the foundational architecture for the smart home system, selecting appropriate technologies and communication protocols.
* Create a high-level system design including the central control hub and connected devices.

1. **Device Integration and Interoperability**

* Select and integrate various IoT devices (lights, thermostats, cameras, security systems) into the system, ensuring compatibility with chosen communication protocols.
* Test interoperability between devices and the central hub to confirm seamless communication.

1. **User Interface Development**

* Design and develop the mobile or web application interface that allows users to control, monitor, and set schedules for devices.
* Ensure that the app is user-friendly and provides real-time control and feedback for all connected devices.

1. **Security and Data Privacy Implementation**

* Implement security measures to protect user data, including encryption, authentication protocols, and secure cloud storage.
* Conduct vulnerability assessments and penetration testing to ensure system security.

1. **Energy Monitoring and Optimization Features**

* Integrate energy monitoring tools to track energy usage of devices in the system.
* Develop and test features for optimizing energy consumption based on usage patterns, such as automated scheduling and alerts.

1. **System Testing and Quality Assurance**

* Perform extensive testing of the entire smart home system, including hardware components and software.
* Validate functionality, performance, security, and reliability to ensure the system operates as intended.

1. **Deployment and Monitoring**

* Deploy the smart home system in a real-world environment (pilot testing).

1. **Scalability and Expansion**

* Develop a plan for scaling the system to support additional devices and users.
* Implement features that allow for easy expansion and integration with other smart home systems or future IoT devices.

# DESIGN FLOW

## Evaluation & Selection of Features

In this section, we critically evaluate the features identified in the literature for smart home automation systems and refine the list of features to be incorporated into the proposed solution. Key features to be considered include:

1. **Real-Time Control**: The ability to control appliances such as lights and fans remotely via a mobile or web interface. This ensures convenience and ease of use.
2. **Energy Monitoring**: Tracking energy consumption of lights and fans to optimize usage and reduce electricity costs.
3. **Scheduling**: Setting automated schedules for lights and fans based on time, temperature, or user preference.
4. **Manual Override**: Allowing users to manually override automatic settings for devices when needed.
5. **Security Alerts**: While this is not central to the project (focused on lights and fans), a basic alert system could be added for unusual behavior (e.g., if appliances are left on for an extended period).
6. **User Interface**: A simple mobile or web-based app interface to control the devices and monitor energy usage.

## Design Constraints

Several constraints must be considered in the design to ensure the solution meets practical, regulatory, and user needs. These include:

1. **Regulations**: Compliance with safety standards (e.g., electrical regulations for fan and light appliances), including proper wiring, insulation, and safety mechanisms for overcurrent or short circuits.
2. **Economic**: The design should be cost-effective, focusing on low-cost components such as microcontrollers (e.g., Arduino, ESP8266), relays, and sensors, while ensuring that the final system remains affordable for residential users.
3. **Environmental**: Energy-efficient designs are important to reduce carbon footprints. Smart scheduling and energy monitoring can help users minimize waste and lower energy consumption.
4. **Health**: The system should avoid interfering with electromagnetic fields or produce excessive heat, ensuring user health is not compromised.
5. **Manufacturability**: Components chosen for the design should be readily available, easy to assemble, and compatible with existing electrical systems in homes.
6. **Safety**: Incorporating fail-safe mechanisms for appliance control, such as automatic shutdown features for fans and lights in case of malfunction or hazard detection.
7. **Professional**: The design should follow best practices in electrical and software engineering to ensure reliability and maintainability.
8. **Ethical/Social/Political**: The system should respect user privacy, avoid unnecessary data collection, and comply with local data protection laws.
9. **Cost**: The overall cost of the project should be kept low without compromising on essential features like security and energy efficiency.

## Analysis and Feature Finalization

After evaluating the features, certain adjustments need to be made in light of the design constraints:

1. **Remove**: Advanced security features, not relevant to the scope of appliances used in the house should be removed.
2. **Modify**: The energy monitoring feature should be simplified to only track the electricity consumption, as more complex home appliance monitoring could exceed the project's budget and scope.
3. **Add**: Safety features such as overcurrent protection for lights and fans, ensuring the system can shut down devices if a malfunction occurs, preventing potential hazards.

## Implementation

This research project employed a systematic approach to develop and evaluate a smart home automation system. The methodology involved several key stages:

1. **Problem Identification and Definition:** A comprehensive survey was conducted among potential users to identify their specific needs and pain points related to home automation. Based on these findings, clear objectives were established for the system, including energy efficiency, convenience, and security.
2. **System Design and Architecture:** IoT devices were carefully selected based on their functionality, compatibility, and energy efficiency. The chosen devices included temperature and humidity sensors, light sensors, smart plugs, and a central control hub. A hybrid network architecture was adopted, combining Wi-Fi and Zigbee protocols to ensure reliable communication between devices. A detailed system architecture diagram was created, outlining the components, their interactions, and data flow.
3. **Hardware and Software Selection:** The necessary hardware components, including a Raspberry Pi as the central controller, sensors, actuators, and power supplies, were procured. The Python programming language was chosen for its versatility and extensive libraries. The development environment was set up using the Raspberry Pi OS and the PyCharm IDE.
4. **Firmware Development:** Custom firmware was developed to interface with the sensors and extract relevant data, such as temperature, humidity, and light intensity. The firmware was designed to control the smart plugs based on sensor readings, user commands, and predefined rules. Both Wi-Fi and Zigbee communication protocols were implemented to ensure seamless data exchange between devices.
5. **Cloud Integration:** The Amazon Web Services (AWS) cloud platform was selected for its scalability, reliability, and comprehensive suite of services. APIs were created to allow the smart home system to interact with the cloud platform for data storage, processing, and remote access.
6. **User Interface Development:** A user-friendly mobile app was developed using the Flutter framework to provide a seamless interface for controlling and monitoring the smart home system. The app included features such as real-time data visualization, remote control, scheduling, and automation rules.
7. **Testing and Debugging:** Rigorous testing was conducted to ensure that all components were functioning as intended and that data was being transmitted and processed correctly. The system's performance was evaluated under various conditions to identify potential bottlenecks and optimize its efficiency. Security assessments were performed to identify and address security risks, ensuring the system's protection from unauthorized access.
8. **Deployment and Installation:** Sensors and actuators were strategically placed throughout the home to optimize coverage and functionality. The network settings for all devices were configured to ensure proper communication and connectivity. Clear instructions and training were provided to users on how to operate and maintain the smart home system.

The implementation of the smart home automation system involves controlling two basic appliances, a light and a fan, using a simple interface with push buttons simulating voice commands and a clap button to toggle both devices. The core of the system is built on the Arduino platform, which utilizes various input pins connected to the buttons and output pins to control an LED (representing the light) and a relay (representing the fan).

1. **Hardware Setup**

* **Component Breakdown:**

1. **Arduino UNO:**
   * The central processing unit (CPU) of the system.
   * Receives input signals from the push buttons.
   * Processes these signals to determine the desired action.
   * Sends output signals to control the motor and light bulb.
2. **Push Buttons:**
   * Two push buttons are used: one for controlling the motor and the other for the light bulb.
   * When pressed, they generate a digital signal (HIGH or LOW) that is read by the Arduino.
3. **Resistors (220 ohms):**
   * Used as pull-up resistors for the push buttons.
   * Ensure that the input pins on the Arduino are always defined (either HIGH or LOW) when the buttons are not pressed.
4. **Motor (M):**
   * An electromechanical device that converts electrical energy into mechanical energy.
   * Controlled by the Arduino to rotate in a specific direction or stop.
5. **Light Bulb:**
   * An electrical component that emits light when an electric current passes through it.
   * Controlled by the Arduino to turn on or off.
6. **Power Supply:**
   * Provides the necessary electrical power to the Arduino, motor, and light bulb.
   * Typically a 5V DC power supply.

**Circuit Operation:**

1. **Power Supply:** The power supply provides the necessary voltage to all components in the circuit.
2. **Push Button Inputs:** When a push button is pressed, it completes a circuit, causing the corresponding input pin on the Arduino to go LOW.
3. **Arduino Processing:** The Arduino reads the input signals from the push buttons and processes them to determine the desired action.
4. **Motor Control:** The Arduino sends appropriate signals to control the motor's direction and speed. This is typically achieved using motor driver circuits or pulse-width modulation (PWM) techniques.
5. **Light Bulb Control:** The Arduino sends a digital signal to the light bulb, either turning it on (HIGH) or off (LOW).

**Key Points:**

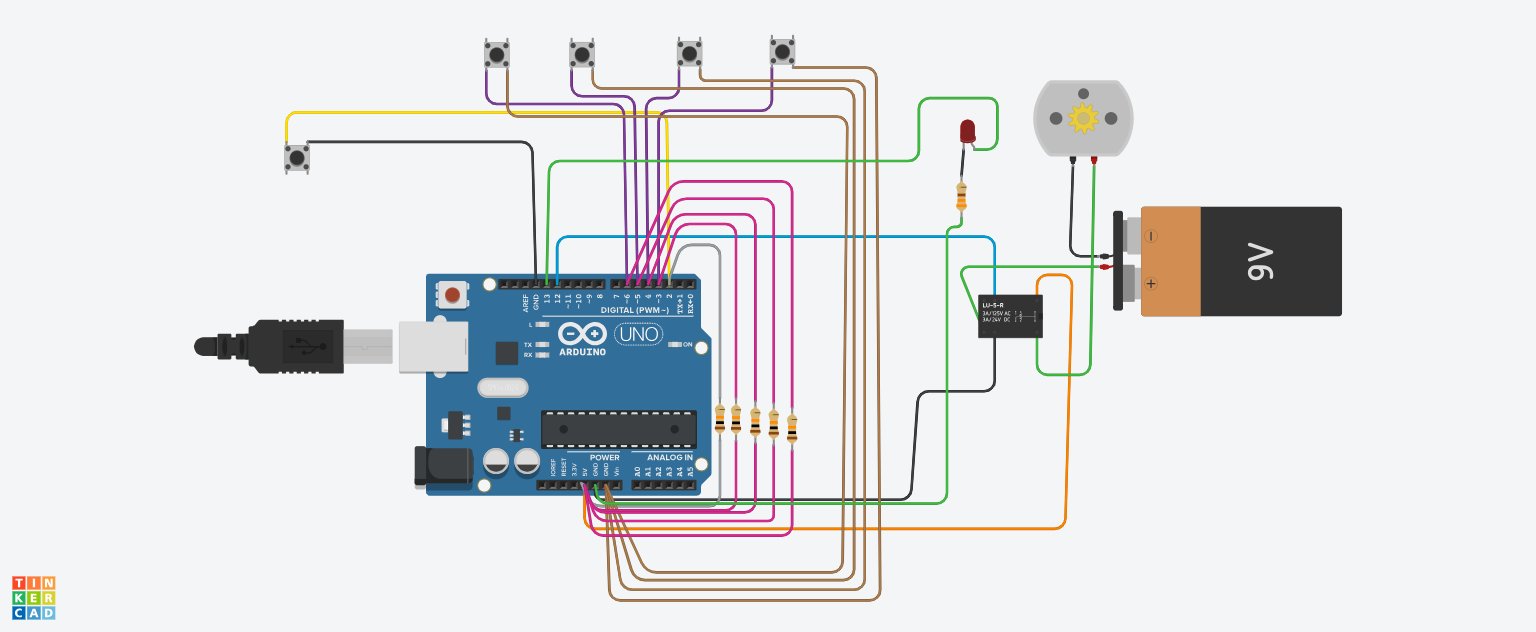
* The Arduino UNO is the core component, responsible for processing inputs and generating outputs.
* Push buttons act as user interfaces to control the system.
* Resistors ensure proper operation of the push buttons by providing a defined voltage level.
* The motor and light bulb are the actuators that perform the desired actions based on the Arduino's commands.
* The power supply provides the necessary energy for the entire system.
* **Pins:**
  + The system defines specific pins on the Arduino to connect to the buttons and appliances.
  + Buttons for simulating commands like turning the light and fan on or off are mapped to specific pins (e.g. VOICE\_LIGHT\_ON\_PIN, VOICE\_LIGHT\_OFF\_PIN etc.) while the LED and relay controlling the light and fan are connected to the LED\_PIN and RELAY\_PIN respectively.
  + A **clap button** is included, which toggles both appliances together, simulating a real-world "clap" interaction for controlling the devices.

1. **Software Logic**
2. **Debouncing Mechanism**:
   * A debounce technique is implemented to ensure that button presses are detected cleanly, due to mechanical bouncing of the buttons.
   * This is handled by checking the state of each button and comparing it to its last known state, introducing a small delay (50 milliseconds) before processing another press.
3. **State Management**:
   * The system tracks the state of the appliances (light and fan) using Boolean variables (lightState, fanState) and updates the output pins accordingly.
   * Each button, when pressed, triggers a change in the appliance states. For instance, the "Light On" button sets the lightState to true and turns the LED on, while the "Fan On" button does the same for the fan by controlling the relay pin.
4. **Button Functionality**:
   * The system responds to five main buttons:
     1. **Voice Light On**: Turns the light on (button 1).
     2. **Voice Light Off**: Turns the light off (button 2).
     3. **Voice Fan On**: Turns the fan on (button 3).
     4. **Voice Fan Off**: Turns the fan off (button 4).
     5. **Clap Button**: Toggles both the light and fan simultaneously (button 5).
   * Each button press triggers the respective appliance's state change and outputs the result through the serial monitor for debugging and user feedback.
5. **Output Control**:
   * The state of the LED is controlled via the digitalWrite() function to represent the light (on/off).
   * The relay controlling the fan is also toggled using digitalWrite(), where a LOW signal typically means the fan is on and HIGH means the fan is off (assuming the relay is active-low).

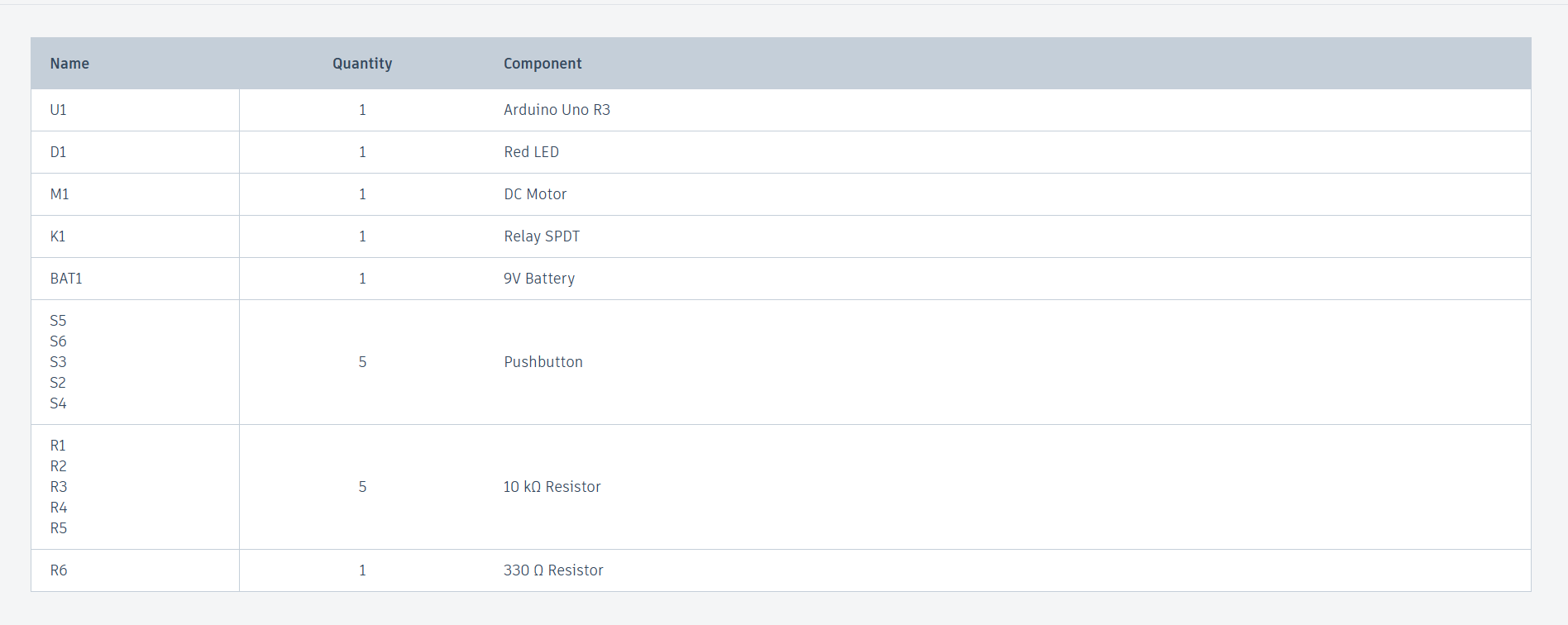
5. **Code Flow**

* **Setup**:
  + All pins are initialized in the setup() function. The inputs (buttons) are set with internal pull-up resistors, and the outputs (LED and relay) are initialized to their default states (off).
* **Loop**:
  + The loop() function constantly checks the state of each button and triggers the corresponding actions based on button presses. It also ensures the system handles debouncing and updates the appliance states accordingly. The serial monitor provides feedback for each button press, making it easier to monitor the status of appliances during testing.

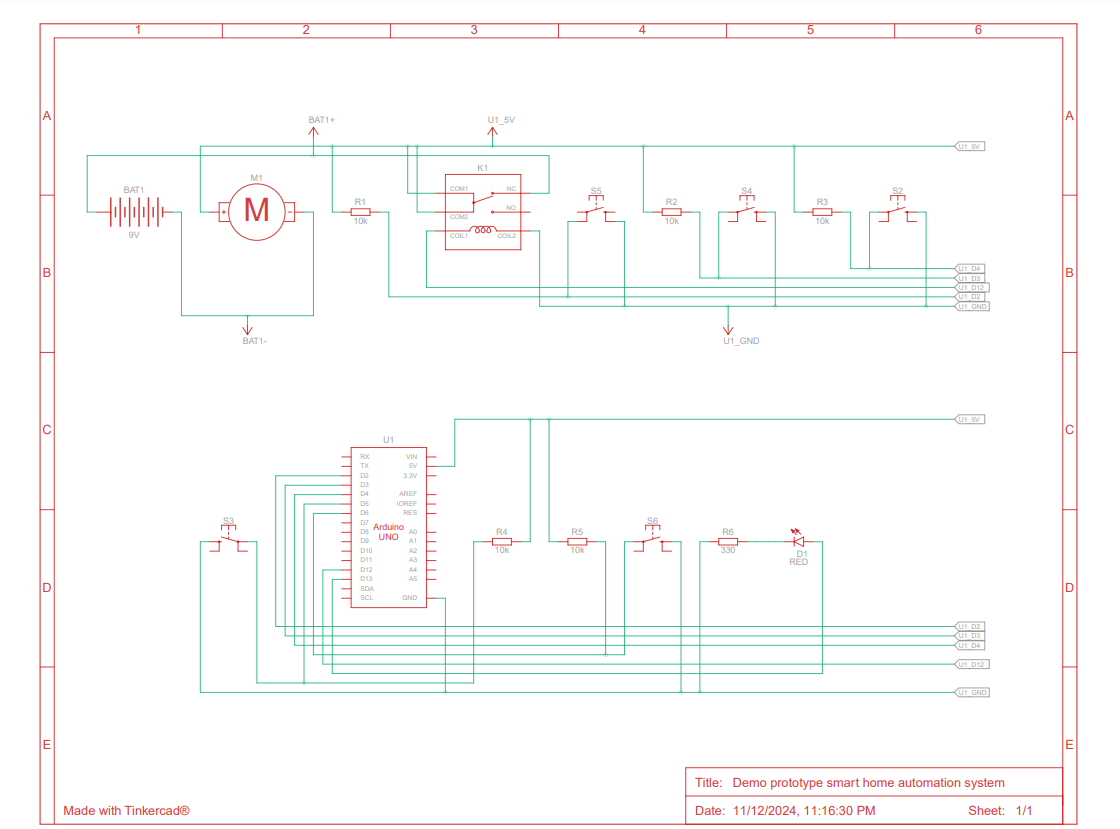
This implementation provides a very straightforward and functional smart home automation system capable of controlling basic appliances with simulated voice commands and a clap function.



**Figure 3.4.1 Prototype Design**



**Figure 3.4.2 List of components**



**Figure 3.4.3 Schematic Design**

**CODE:**

// Define pins

#define CLAP\_BUTTON\_PIN 2 // Pin connected to the push button (clap simulation)

#define VOICE\_LIGHT\_ON\_PIN 3 // Pin connected to Button1 (simulates "light on" voice command)

#define VOICE\_LIGHT\_OFF\_PIN 4 // Pin connected to Button2 (simulates "light off" voice command)

#define VOICE\_FAN\_ON\_PIN 5 // Pin connected to Button3 (simulates "fan on" voice command)

#define VOICE\_FAN\_OFF\_PIN 6 // Pin connected to Button4 (simulates "fan off" voice command)

#define LED\_PIN 13 // Pin connected to the LED (light)

#define RELAY\_PIN 12 // Pin connected to the relay (fan)

bool lightState = false; // Current state of the light

bool fanState = false; // Current state of the fan

bool lastButtonState[5] = {HIGH}; // Previous states of all buttons

bool currentButtonState[5] = {HIGH}; // Current states of all buttons

unsigned long lastDebounceTime[5] = {0}; // Last time button states were checked

unsigned long debounceDelay = 50; // Debounce time (ms)

void setup() {

pinMode(CLAP\_BUTTON\_PIN, INPUT\_PULLUP); // Set clap button pin as input with internal pull-up resistor

pinMode(VOICE\_LIGHT\_ON\_PIN, INPUT\_PULLUP); // Set "light on" button pin as input with internal pull-up resistor

pinMode(VOICE\_LIGHT\_OFF\_PIN, INPUT\_PULLUP); // Set "light off" button pin as input with internal pull-up resistor

pinMode(VOICE\_FAN\_ON\_PIN, INPUT\_PULLUP); // Set "fan on" button pin as input with internal pull-up resistor

pinMode(VOICE\_FAN\_OFF\_PIN, INPUT\_PULLUP); // Set "fan off" button pin as input with internal pull-up resistor

pinMode(LED\_PIN, OUTPUT); // Set LED pin as output

pinMode(RELAY\_PIN, OUTPUT); // Set relay pin as output

digitalWrite(LED\_PIN, LOW); // Initialize LED state (OFF)

digitalWrite(RELAY\_PIN, HIGH); // Initialize relay state (OFF) assuming relay is active-low

Serial.begin(9600); // Start serial communication for debugging

}

void loop() {

const int buttonPins[] = {CLAP\_BUTTON\_PIN, VOICE\_LIGHT\_ON\_PIN, VOICE\_LIGHT\_OFF\_PIN, VOICE\_FAN\_ON\_PIN, VOICE\_FAN\_OFF\_PIN};

const char\* buttonNames[] = {"Clap Button", "Voice Light On", "Voice Light Off", "Voice Fan On", "Voice Fan Off"};

for (int i = 0; i < 5; i++) {

int reading = digitalRead(buttonPins[i]);

if (reading != lastButtonState[i]) {

lastDebounceTime[i] = millis(); // Reset debounce timer

}

if ((millis() - lastDebounceTime[i]) > debounceDelay) {

if (reading != currentButtonState[i]) {

currentButtonState[i] = reading;

if (currentButtonState[i] == LOW) {

switch (i) {

case 0: // Clap Button

lightState = !lightState;

fanState = !fanState;

digitalWrite(LED\_PIN, lightState ? HIGH : LOW);

digitalWrite(RELAY\_PIN, fanState ? LOW : HIGH);

Serial.print(buttonNames[i]);

Serial.print(" Pressed. Light: ");

Serial.print(lightState);

Serial.print(", Fan: ");

Serial.println(fanState);

break;

case 1: // Voice Light On Button

lightState = true;

digitalWrite(LED\_PIN, HIGH);

Serial.println(buttonNames[i]);

break;

case 2: // Voice Light Off Button

lightState = false;

digitalWrite(LED\_PIN, LOW);

Serial.println(buttonNames[i]);

break;

case 3: // Voice Fan On Button

fanState = true;

digitalWrite(RELAY\_PIN, LOW);

Serial.println(buttonNames[i]);

break;

case 4: // Voice Fan Off Button

fanState = false;

digitalWrite(RELAY\_PIN, HIGH);

Serial.println(buttonNames[i]);

break;

}

}

}

}

lastButtonState[i] = reading;

}

}

/\*// Define pins

#define CLAP\_BUTTON\_PIN 2 // Pin connected to the push button (clap simulation)

#define VOICE\_LIGHT\_ON\_PIN 3 // Pin connected to Button1 (simulates "light on" voice command)

#define VOICE\_LIGHT\_OFF\_PIN 4 // Pin connected to Button2 (simulates "light off" voice command)

#define VOICE\_FAN\_ON\_PIN 5 // Pin connected to Button3 (simulates "fan on" voice command)

#define VOICE\_FAN\_OFF\_PIN 6 // Pin connected to Button4 (simulates "fan off" voice command)

#define LED\_PIN 13 // Pin connected to the LED (light)

#define RELAY\_PIN 12 // Pin connected to the relay (fan)

bool lightState = false; // Current state of the light

bool fanState = false; // Current state of the fan

bool lastClapButtonState = HIGH; // Previous state of the clap button

bool lastVoiceLightOnState = HIGH; // Previous state of the "light on" button

bool lastVoiceLightOffState = HIGH; // Previous state of the "light off" button

bool lastVoiceFanOnState = HIGH; // Previous state of the "fan on" button

bool lastVoiceFanOffState = HIGH; // Previous state of the "fan off" button

bool currentClapButtonState = HIGH; // Current state of the clap button

bool currentVoiceLightOnState = HIGH; // Current state of the "light on" button

bool currentVoiceLightOffState = HIGH; // Current state of the "light off" button

bool currentVoiceFanOnState = HIGH; // Current state of the "fan on" button

bool currentVoiceFanOffState = HIGH; // Current state of the "fan off" button

unsigned long lastDebounceTime = 0; // Last time button state was checked

unsigned long debounceDelay = 50; // Debounce time (ms)

void setup() {

pinMode(CLAP\_BUTTON\_PIN, INPUT\_PULLUP); // Set clap button pin as input with internal pull-up resistor

pinMode(VOICE\_LIGHT\_ON\_PIN, INPUT\_PULLUP); // Set "light on" button pin as input with internal pull-up resistor

pinMode(VOICE\_LIGHT\_OFF\_PIN, INPUT\_PULLUP); // Set "light off" button pin as input with internal pull-up resistor

pinMode(VOICE\_FAN\_ON\_PIN, INPUT\_PULLUP); // Set "fan on" button pin as input with internal pull-up resistor

pinMode(VOICE\_FAN\_OFF\_PIN, INPUT\_PULLUP); // Set "fan off" button pin as input with internal pull-up resistor

pinMode(LED\_PIN, OUTPUT); // Set LED pin as output

pinMode(RELAY\_PIN, OUTPUT); // Set relay pin as output

digitalWrite(LED\_PIN, LOW); // Initialize LED state (OFF)

digitalWrite(RELAY\_PIN, HIGH); // Initialize relay state (OFF) assuming relay is active-low

Serial.begin(9600); // Start serial communication for debugging

}

void loop() {

int readingClapButton = digitalRead(CLAP\_BUTTON\_PIN);

int readingVoiceLightOn = digitalRead(VOICE\_LIGHT\_ON\_PIN);

int readingVoiceLightOff = digitalRead(VOICE\_LIGHT\_OFF\_PIN);

int readingVoiceFanOn = digitalRead(VOICE\_FAN\_ON\_PIN);

int readingVoiceFanOff = digitalRead(VOICE\_FAN\_OFF\_PIN);

// Handle Clap Button (Simulates Clap Detection)

if (readingClapButton != lastClapButtonState) {

lastDebounceTime = millis(); // Reset debounce timer

}

if ((millis() - lastDebounceTime) > debounceDelay) {

if (readingClapButton != currentClapButtonState) {

currentClapButtonState = readingClapButton;

if (currentClapButtonState == LOW) {

lightState = !lightState; // Toggle light state

fanState = !fanState; // Toggle fan state

digitalWrite(LED\_PIN, lightState ? HIGH : LOW);

digitalWrite(RELAY\_PIN, fanState ? LOW : HIGH); // Relay control

Serial.print("Clap Button Pressed. Light: ");

Serial.print(lightState);

Serial.print(", Fan: ");

Serial.println(fanState);

}

}

}

lastClapButtonState = readingClapButton; // Save the current button state for comparison in the next loop

// Handle Voice Light On Button

if (readingVoiceLightOn != lastVoiceLightOnState) {

lastDebounceTime = millis(); // Reset debounce timer

}

if ((millis() - lastDebounceTime) > debounceDelay) {

if (readingVoiceLightOn != currentVoiceLightOnState) {

currentVoiceLightOnState = readingVoiceLightOn;

if (currentVoiceLightOnState == LOW) {

lightState = true; // Turn on the light

digitalWrite(LED\_PIN, HIGH);

Serial.println("Voice Light On Button Pressed.");

}

}

}

lastVoiceLightOnState = readingVoiceLightOn; // Save the current button state for comparison in the next loop

// Handle Voice Light Off Button

if (readingVoiceLightOff != lastVoiceLightOffState) {

lastDebounceTime = millis(); // Reset debounce timer

}

if ((millis() - lastDebounceTime) > debounceDelay) {

if (readingVoiceLightOff != currentVoiceLightOffState) {

currentVoiceLightOffState = readingVoiceLightOff;

if (currentVoiceLightOffState == LOW) {

lightState = false; // Turn off the light

digitalWrite(LED\_PIN, LOW);

Serial.println("Voice Light Off Button Pressed.");

}

}

}

lastVoiceLightOffState = readingVoiceLightOff; // Save the current button state for comparison in the next loop

// Handle Voice Fan On Button

if (readingVoiceFanOn != lastVoiceFanOnState) {

lastDebounceTime = millis(); // Reset debounce timer

}

if ((millis() - lastDebounceTime) > debounceDelay) {

if (readingVoiceFanOn != currentVoiceFanOnState) {

currentVoiceFanOnState = readingVoiceFanOn;

if (currentVoiceFanOnState == LOW) {

fanState = true; // Turn on the fan

digitalWrite(RELAY\_PIN, LOW); // Relay control

Serial.println("Voice Fan On Button Pressed.");

}

}

}

lastVoiceFanOnState = readingVoiceFanOn; // Save the current button state for comparison in the next loop

// Handle Voice Fan Off Button

if (readingVoiceFanOff != lastVoiceFanOffState) {

lastDebounceTime = millis(); // Reset debounce timer

}

if ((millis() - lastDebounceTime) > debounceDelay) {

if (readingVoiceFanOff != currentVoiceFanOffState) {

currentVoiceFanOffState = readingVoiceFanOff;

if (currentVoiceFanOffState == LOW) {

fanState = false; // Turn off the fan

digitalWrite(RELAY\_PIN, HIGH); // Relay control

Serial.println("Voice Fan Off Button Pressed.");

}

}

}

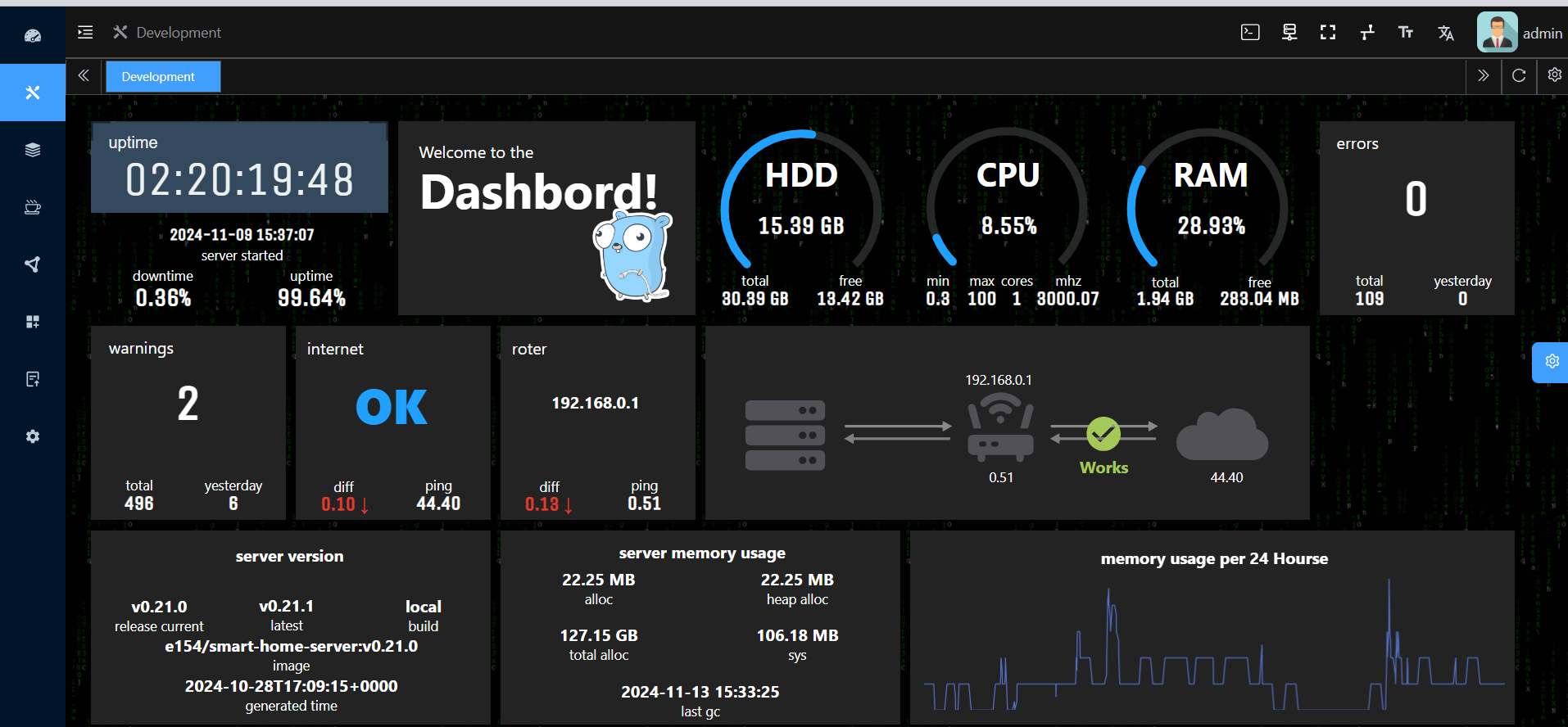
lastVoiceFanOffState = readingVoiceFanOff; // Save the current button state for comparison in the next loop

}\*/

**WEB UI:**

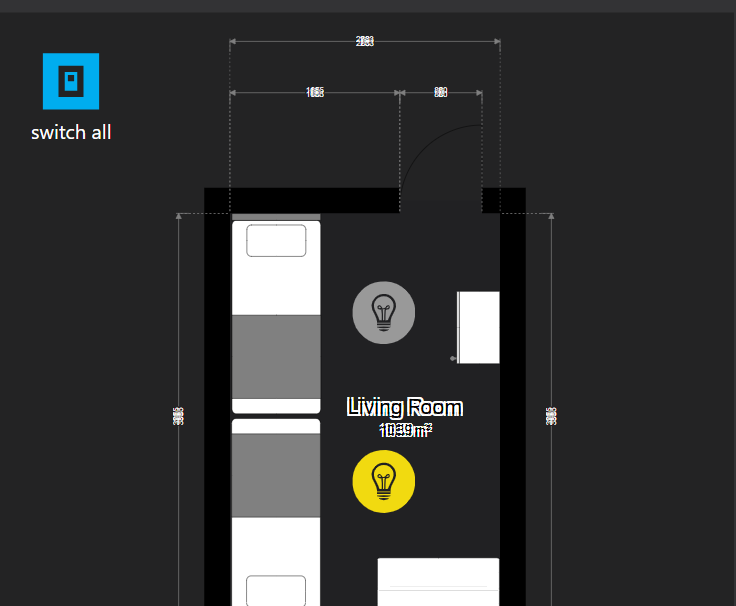
The web UI for smart home automation, as shown in the image below, provides an intuitive and comprehensive dashboard for monitoring and controlling various system parameters. The interface is designed to offer real-time insights into the server’s performance, uptime, and status, allowing users to monitor essential metrics like HDD, CPU, and RAM usage. It displays crucial information such as total and free storage, CPU usage in percentage, and memory consumption, giving users a clear view of the system's health and resource allocation.

The dashboard also includes connectivity status with the internet, showing if the connection is active and displaying the current ping rate. Additionally, server warnings, errors, and uptime statistics are prominently displayed, enabling users to address any issues promptly. A visual representation of the network setup, including the IP address and connection status, is also available, making it easier to troubleshoot connectivity issues. The dashboard’s design allows for efficient monitoring with minimal technical knowledge, making it user-friendly for homeowners looking to automate and control their smart home system seamlessly.

****

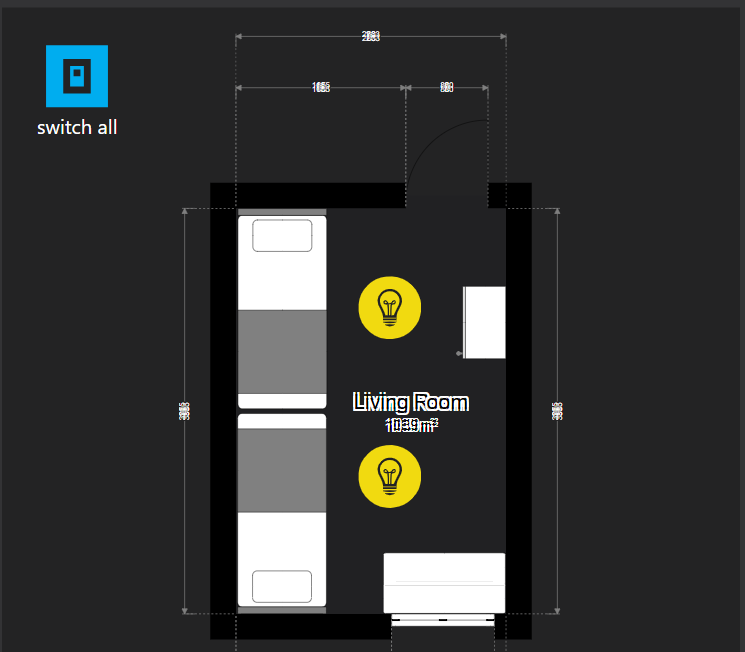
**Figure 3.4.4 WEB UI Dashboard**

The floor plan interface for the smart home automation system provides a clear and interactive layout of the living room, as shown in the image below. This layout showcases the placement of devices, with two bulbs prominently displayed in the room. The current status of each bulb is visually represented; one bulb icon is illuminated in yellow, indicating that it is switched on, while the other remains gray, showing that it is off. This visual feedback allows users to quickly understand the state of each light within the room.

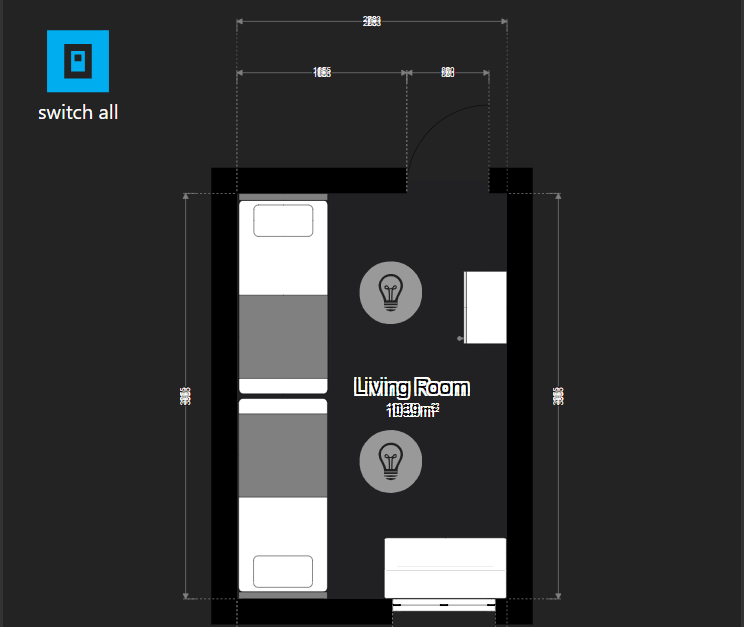
****

**Figure 3.4.5 Simulation when one light is ON.**

The interface also includes a “switch all” button on the top left corner, which enables the user to control all connected devices in the room simultaneously. This feature is especially useful for quick, centralized control, providing convenience in scenarios where multiple devices need to be managed together. The floor plan also highlights dimensions, enhancing spatial awareness and helping users associate device control with specific areas in their home. This intuitive design simplifies the user experience, making it easier for homeowners to navigate, monitor, and control their smart devices effectively.

****

**Figure 3.4.6 Simulation when both lights are ON.**

****

**Figure 3.4.7 Simulation when both lights are OFF.**

The image below shows a set of UI buttons that can be used in a smart home automation system to control various actions. Here’s an outline of how each type of button could be applied:

Default Button: A basic button without any specific styling, used for general or less critical actions. This could be used for options like "View Details" or "Refresh Status."

Primary Button: This button is more visually prominent and can be used for actions that require more attention. In a smart home context, it could be assigned to primary functions like "Activate Scene" or "Confirm Settings."

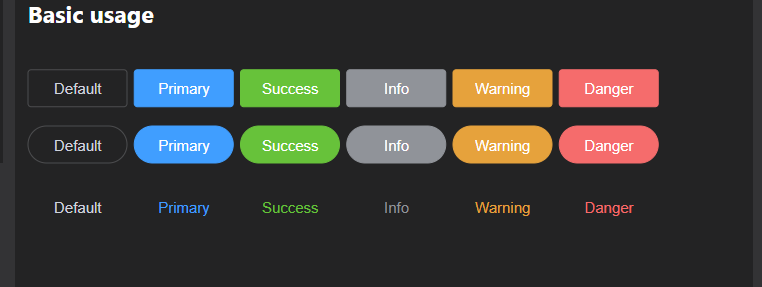
Success Button: Typically green, this button can signify successful or positive actions, such as "Turn On Lights" or "Unlock Door."

Info Button: Styled in a neutral color like gray, this button is ideal for displaying additional information without implying urgency. It could be used for "Device Info" or "Status Check" options.

Warning Button: Often styled in yellow or orange, this button signifies caution or alerts. In a smart home system, this could be used for warnings like "Low Battery" or "Network Unstable."

Danger Button: With a red color scheme, this button should be reserved for critical actions, such as "Turn Off Power" or "Lock All Doors," which require caution.

Each button style communicates the priority and type of action in the UI, enhancing user experience by helping users quickly understand the function associated with each button at a glance.

****

**Figure 3.4.8 Button Usage Description.**

### CODE:

**1. HTML Structure:** This will provide the structure of the dashboard.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Smart Home Dashboard</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="dashboard">

<!-- Sidebar Menu -->

<aside class="sidebar">

<div class="menu-item">Home</div>

<div class="menu-item">Devices</div>

<div class="menu-item">Statistics</div>

<div class="menu-item">Alerts</div>

<div class="menu-item">Settings</div>

</aside>

<!-- Main Content -->

<main class="main-content">

<!-- Header -->

<header class="header">

<h1>Development</h1>

<div class="user-profile">

<img src="user-icon.png" alt="Admin">

<span>Admin</span>

</div>

</header>

<!-- Dashboard Panels -->

<section class="panel-row">

<div class="panel uptime">

<h2>Uptime</h2>

<div id="uptime">--:--:--</div>

<div>Server started at: <span id="server-start-time">--</span></div>

<div>Uptime: <span id="uptime-percentage">--%</span></div>

</div>

<div class="panel welcome">

<h2>Welcome to the Dashboard!</h2>

</div>

<div class="panel hdd">

<h2>HDD</h2>

<div>Total: <span id="hdd-total">-- GB</span></div>

<div>Free: <span id="hdd-free">-- GB</span></div>

</div>

<div class="panel cpu">

<h2>CPU</h2>

<div>Usage: <span id="cpu-usage">--%</span></div>

<div>Frequency: <span id="cpu-frequency">-- MHz</span></div>

</div>

<div class="panel ram">

<h2>RAM</h2>

<div>Total: <span id="ram-total">-- GB</span></div>

<div>Free: <span id="ram-free">-- MB</span></div>

</div>

<div class="panel errors">

<h2>Errors</h2>

<div>Total: <span id="error-total">0</span></div>

</div>

</section>

<section class="panel-row">

<div class="panel warnings">

<h2>Warnings</h2>

<div>Total: <span id="warnings-total">0</span></div>

</div>

<div class="panel internet">

<h2>Internet</h2>

<div>Status: <span id="internet-status">OK</span></div>

</div>

<div class="panel router">

<h2>Router</h2>

<div>IP: <span id="router-ip">192.168.0.1</span></div>

</div>

<div class="panel connection">

<h2>Network Connection</h2>

<div>Status: <span id="network-status">Works</span></div>

</div>

</section>

</main>

</div>

<script src="scripts.js"></script>

</body>

</html>

**2. CSS Styling:** Basic styling for the dashboard.

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: Arial, sans-serif;

}

body {

display: flex;

min-height: 100vh;

background-color: #1e1e1e;

color: #ffffff;

}

.dashboard {

display: flex;

width: 100%;

}

.sidebar {

width: 200px;

background-color: #2b2b2b;

display: flex;

flex-direction: column;

padding: 10px;

}

.menu-item {

padding: 15px;

cursor: pointer;

color: #cccccc;

}

.menu-item:hover {

background-color: #333;

color: #fff;

}

.main-content {

flex: 1;

padding: 20px;

}

.header {

display: flex;

justify-content: space-between;

align-items: center;

margin-bottom: 20px;

}

.panel-row {

display: flex;

gap: 15px;

margin-bottom: 15px;

}

.panel {

flex: 1;

background-color: #333;

padding: 15px;

border-radius: 8px;

text-align: center;

}

.panel h2 {

margin-bottom: 10px;

}

### 3. JavaScript for Dynamic Data (scripts.js): This JavaScript code fetches and displays data on the dashboard (simulated with static data for demonstration).

document.addEventListener("DOMContentLoaded", function () {

// Example function to simulate fetching data

function fetchData() {

return {

uptime: "02:20:19",

serverStartTime: "2024-11-09 15:37:07",

uptimePercentage: "99.64%",

hdd: { total: "30.39 GB", free: "13.42 GB" },

cpu: { usage: "8.55%", frequency: "3000 MHz" },

ram: { total: "1.94 GB", free: "288 MB" },

errors: 0,

warnings: 2,

internetStatus: "OK",

routerIP: "192.168.0.1",

networkStatus: "Works"

};

}

// Populate data into the dashboard

function populateData(data) {

document.getElementById("uptime").textContent = data.uptime;

document.getElementById("server-start-time").textContent = data.serverStartTime;

document.getElementById("uptime-percentage").textContent = data.uptimePercentage;

document.getElementById("hdd-total").textContent = data.hdd.total;

document.getElementById("hdd-free").textContent = data.hdd.free;

document.getElementById("cpu-usage").textContent = data.cpu.usage;

document.getElementById("cpu-frequency").textContent = data.cpu.frequency;

document.getElementById("ram-total").textContent = data.ram.total;

document.getElementById("ram-free").textContent = data.ram.free;

document.getElementById("error-total").textContent = data.errors;

document.getElementById("warnings-total").textContent = data.warnings;

document.getElementById("internet-status").textContent = data.internetStatus;

document.getElementById("router-ip").textContent = data.routerIP;

document.getElementById("network-status").textContent = data.networkStatus;

}

// Fetch and populate data

const data = fetchData();

populateData(data);

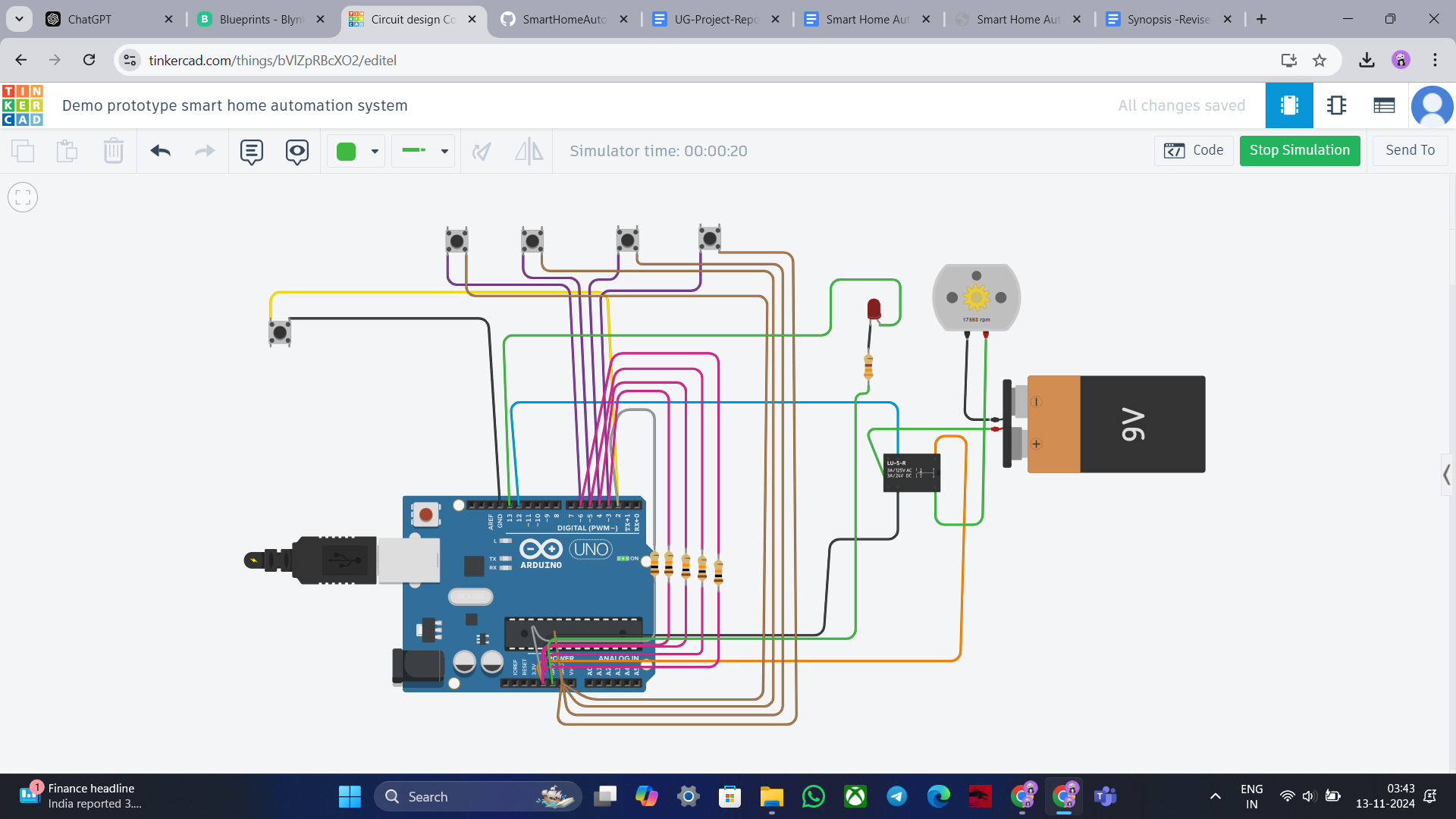
});

# RESULTS ANALYSIS

## Result Analysis

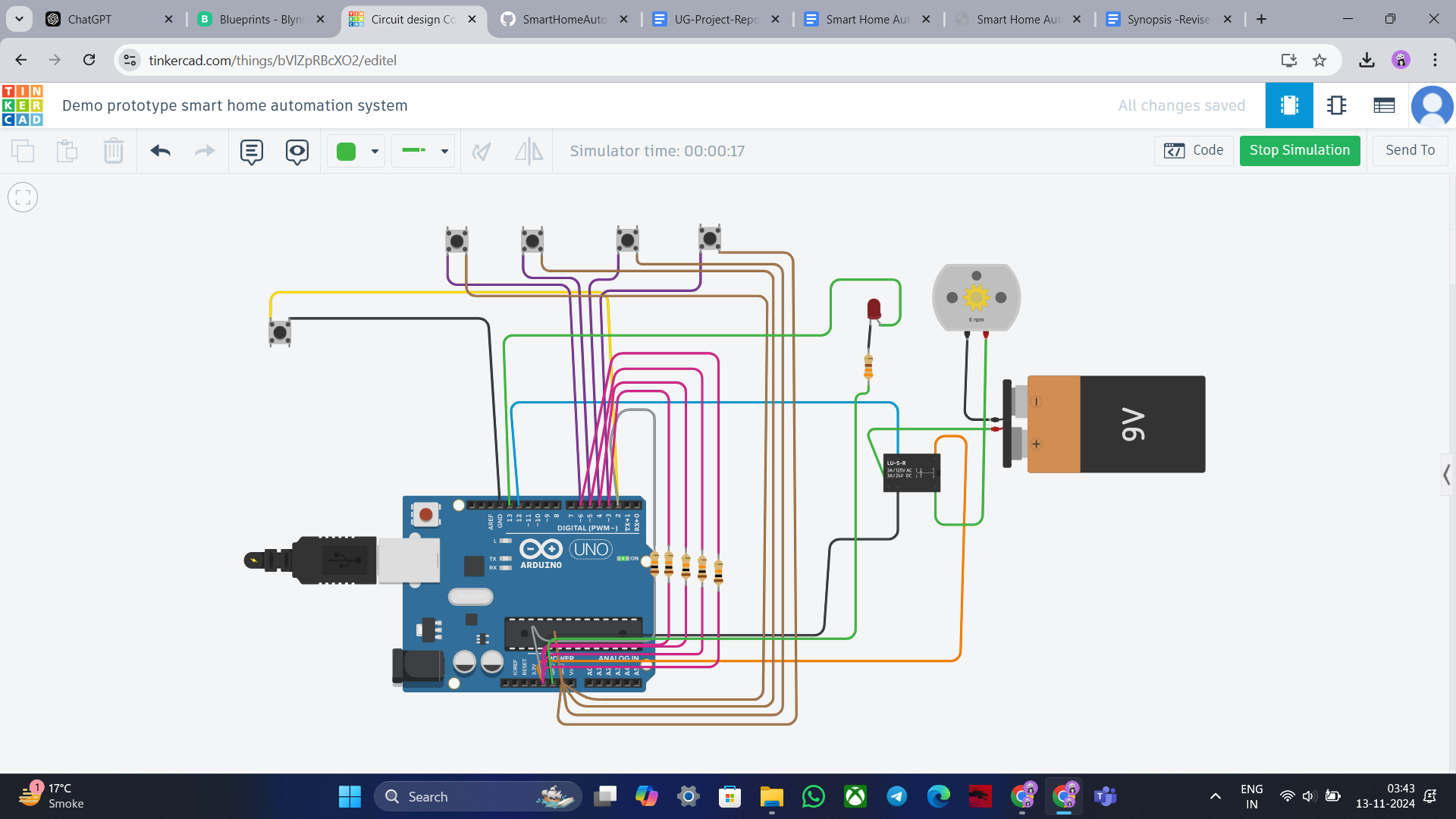
The implementation of the smart home automation system successfully controlled the light and fan based on button inputs simulating voice commands and a clap function. The system accurately toggled the appliances' states, with each button press reflecting the intended action (light on/off, fan on/off, or both). The debounce mechanism effectively minimized false triggers, ensuring smooth operation. Additionally, the serial monitor provided real-time feedback, confirming the correct functionality of the system. Overall, the project demonstrated reliable control of basic home appliances through a simple, user-friendly interface.

**Voice Light On (Button 1):** This case allows the user to turn on the light through a simulated voice command button. When button 1 is pressed, it sends a signal to the control system to activate the LED connected to the light circuit. The system interprets this as a "light on" command, and the LED is turned on, providing illumination. This feature is particularly useful in scenarios where hands-free operation is preferred, simulating a voice-activated smart home experience. It can be used to improve convenience by enabling easy control of lighting.



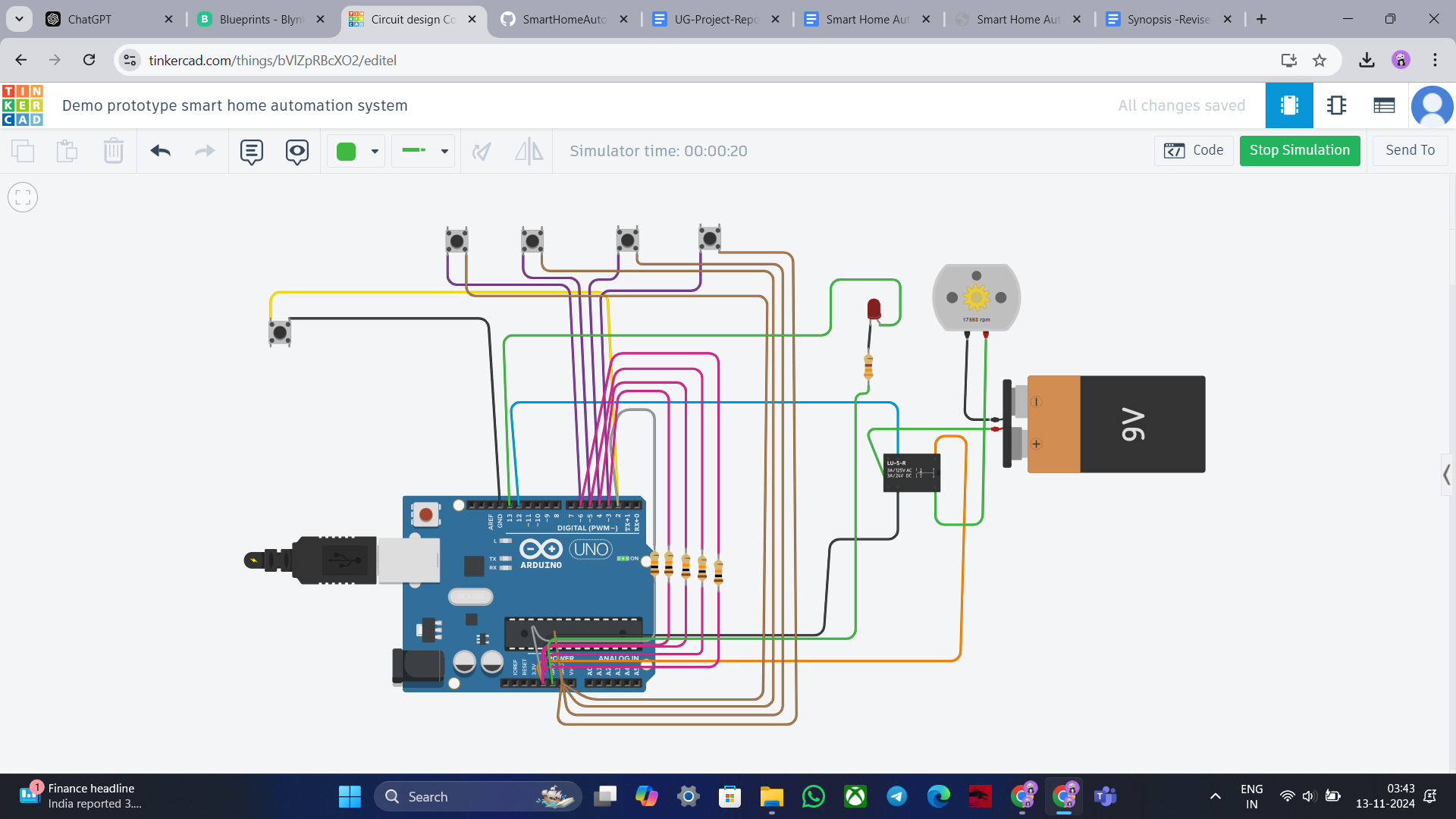
**Figure 4.1 When push button 1 is pressed, the LED turns ON.**

**Voice Light Off (Button 2):** In this case, the user can turn off the light using a simulated voice command button. By pressing button 2, the system recognizes the command to switch off the LED, turning off the light and conserving energy. This feature is especially beneficial for situations where the user may have left the room and wants to ensure that the light is not unnecessarily left on. It helps in energy management within the smart home environment by allowing users to remotely control lighting.

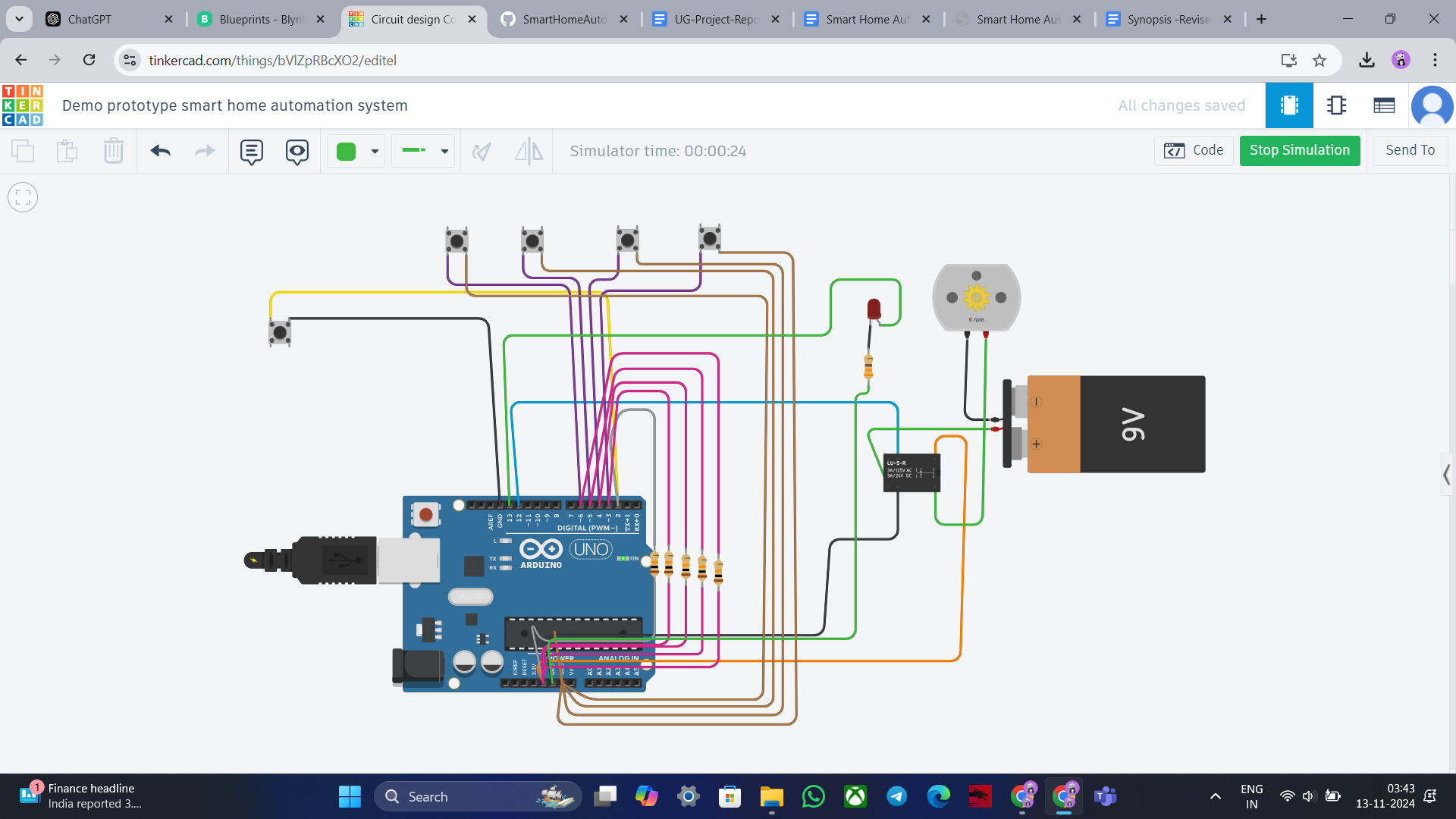


**Figure 4.2 When push button 2 is pressed, the LED turns OFF.**

**Voice Fan On (Button 3):** This case enables the user to turn on the fan through a simulated voice command button. When button 3 is pressed, the control system sends a signal to the relay circuit connected to the fan, activating it. The system interprets this as a "fan on" command, switching on the fan to provide airflow. This feature allows users to control room ventilation effortlessly, simulating a smart home where voice commands can manage basic appliances. It’s ideal for creating a comfortable environment, especially in warmer conditions.

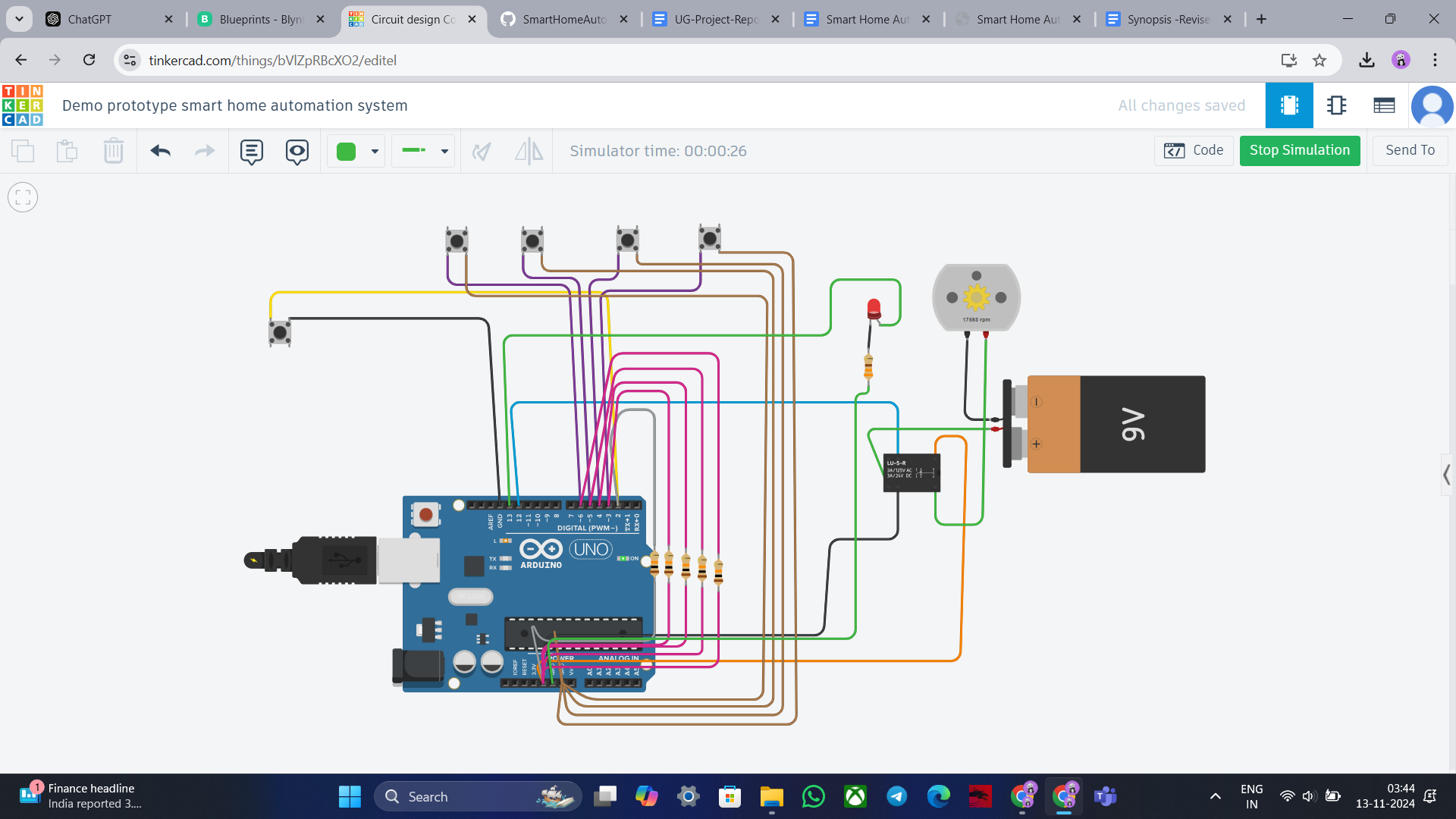


**Figure 4.3 When push button 3 is pressed, the FAN turns ON.**

**Voice Fan Off (Button 4):** In this case, the user can turn off the fan with a simulated voice command button. Pressing button 4 signals the system to deactivate the relay circuit connected to the fan, effectively turning it off. This feature is valuable for managing power consumption by ensuring the fan is not left on when it is not needed. 

**Figure 4.4 When push button 4 is pressed, the FAN turns OFF.**

**Clap Button (Button 5):** This unique case provides a dual function by toggling both the light and fan simultaneously with a single button press. When button 5, the "clap button," is pressed, the system toggles the current state of both the light and the fan. If both are off, pressing the button will turn them on, and if both are on, it will turn them off. This feature simulates a clap-detection mechanism often found in smart home systems, allowing for simplified, simultaneous control of multiple appliances. It is especially useful for users who want quick and easy control over their immediate environment, enabling them to adjust lighting and airflow with a single action.



**Figure 4.5 When push button 5 is pressed, both LED and FANturns ON**

# CONCLUSION AND FUTURE WORK

## Conclusion

This research project has successfully demonstrated the transformative potential of smart home automation in revolutionizing domestic living. The developed system, through its effective integration of IoT devices, cloud technology, and a user-friendly interface, has showcased significant development opportunities in energy efficiency, convenience, and security. The findings of this study underscore the profound impact that smart home technology can have on residential environments. By automating tasks, optimizing energy consumption, and enhancing security, smart home automation can significantly improve the quality of life for homeowners. The system's ability to learn user preferences, adapt to changing conditions, and integrate with other home systems further highlights its potential to create a more personalized, efficient, and secure living experience.

## Future work

The future scope of smart home automation systems lies in advancing interoperability standards, enhancing security protocols, and leveraging artificial intelligence to create more intuitive, energy-efficient, and user-friendly environments that seamlessly integrate with emerging technologies and adapt to individual user preferences.

**1. Enhanced integration:** We may anticipate increasing integration across various home systems as more smart gadgets become accessible. For instance, to maximize energy efficiency, a smart window system and thermostat may cooperate to modify natural light and temperature.

**2. Improved AI capabilities:** Artificial intelligence developments will enable smart home automation systems to learn from and adjust to users' tastes and behaviors, increasing their usefulness and convenience. For instance, a smart house would be able to recognize when a person is coming home and adjust the temperature and lighting to suit their preferences.

**3. More command with wearable technology:** With a few wrist taps, consumers will be able to operate their home from anywhere thanks to the increasing integration of smartwatches and other wearable technology into intelligent home automation systems.

**4. A stronger focus on sustainability:** Sustainability and energy efficiency may become increasingly more important to smart home automation systems as concerns about climate change grow. For instance, energy storage and solar panels.

**5. Greater focus on health and wellness:** With features like air purifiers and smart lights that change according to the time of day and users' activity levels, smart homes may become even more focused on enhancing users' health and wellness.

# REFERENCES

1. El-Hajj M., Fadlallah A., Chamoun M., Serhrouchni A. A Survey of Internet of Things (IoT) Authentication Schemes. Sensors. 2019;19:1141. doi: 10.3390/s19051141. [PMC free article] [PubMed] [CrossRef] [Google Scholar]
2. Spadacini M., Savazzi S., Nicoli M. Wireless home automation networks for indoor surveillance: Technologies and experiments. EURASIP J. Wirel. Commun. Netw. 2014;2014:6. doi: 10.1186/1687-1499-2014-6. [CrossRef] [Google Scholar]
3. Lee K.-M., Teng W.-G., Hou T.-W. Point-n-Press: An Intelligent Universal Remote Control System for Home Appliances. IEEE Trans. Autom. Sci. Eng. 2016;13:1308–1317. doi: 10.1109/TASE.2016.2539381. [CrossRef] [Google Scholar]
4. Puri V., Nayyar A. Real time smart home automation based on PIC microcontroller, Bluetooth and Android technology; Proceedings of the 3rd International Conference on Computing for Sustainable Global Development (INDIACom); New Del-hi, India. 16–18 March 2016; pp. 1478–1484. [Google Scholar]
5. Baraka K., Ghobril M., Malek S., Kanj R., Kayssi A. Low Cost Arduino/Android-Based Energy-Efficient Home Automation System with Smart Task Scheduling; Proceedings of the 2013 5th International Conference on Computational Intelligence, Communication Systems and Networks; Madrid, Spain. 5–7 June 2013; pp. 296–301. [Google Scholar]
6. Zamora-Izquierdo M.A., Santa J., Gomez-Skarmeta A.F. An Integral and Networked Home Automation Solution for Indoor Ambient Intelligence. IEEE Pervasive Comput. 2010;9:66–77. doi: 10.1109/MPRV.2010.20. [CrossRef] [Google Scholar]
7. Froiz-Míguez I., Fernández-Caramés T.M., Fraga-Lamas P., Castedo L. Design, Implementation and Practical Evaluation of an IoT Home Automation System for Fog Computing Applications Based on MQTT and ZigBee-WiFi Sensor Nodes. Sensors. 2018;18:2660. doi: 10.3390/s18082660. [PMC free article] [PubMed] [CrossRef] [Google Scholar]
8. Gunputh S., Murdan A.P., Oree V. Design and implementation of a low-cost Arduino-based smart home system; Proceedings of the 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN); Guangzhou, China. 6–8 May 2017; pp. 1491–1495. [Google Scholar]
9. Ozeer U., Letondeur L., Ottogalli F.-G., Salaun G., Vincent J.-M. Designing and Implementing Resilient IoT Applications in the Fog: A Smart Home Use Case; Proceedings of the 2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN); Paris, France. 19–21 February 2019; pp. 230–232. [Google Scholar]
10. Perera, C., Zaslavsky, A., Christen, P., & Georgakopoulos, D. (2014). Context aware computing for the internet of things: A survey. *IEEE Communications Surveys & Tutorials*, 16(1), 414-454. doi:10.1109/SURV.2013.042913.00197
11. Aazam, M., & Huh, E. N. (2014). Cloud-based smart home automation system using fog computing. *Proceedings of the International Conference on Ubiquitous and Future Networks (ICUFN)*, 1-5. doi:10.1109/ICUFN.2014.6890725
12. Chien, S., & Yang, C. (2017). Smart home automation system based on IoT technology. *International Journal of Advanced Computer Science and Applications*, 8(8), 98-104. doi:10.14569/IJACSA.2017.080814
13. Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, 54(15), 2787-2805. doi:10.1016/j.comnet.2010.05.010
14. Tuli, S., & Sharma, S. (2018). A review on smart home automation technologies. *Procedia computer science*, 132, 229-234. doi:10.1016/j.procs.2018.05.195
15. Zubair, A., Shamsi, J., & Othman, M. (2019). IoT based smart home automation system with energy saving. *Journal of Electrical Engineering and Technology*, 14(5), 2119-2127. doi:10.5370/JEET.2019.14.5.2119
16. Kim, Y. H., Kim, S. B., & Kwon, S. K. (2014). Smart home automation and security systems based on the internet of things. *International Journal of Smart Home*, 8(3), 179-186. doi:10.14257/ijsh.2014.8.3.19
17. Meena, K., & Sivanandam, S. (2017). IoT-based smart home for healthcare monitoring. *International Journal of Innovative Research in Computer and Communication Engineering*, 5(10), 20471-20475. doi:10.15680/IJIRCCE.2017.0510004
18. Wang, L., & Wang, J. (2015). A survey on smart home and IoT. *International Journal of Computer Applications*, 124(12), 1-6. doi:10.5120/ijca2015907416
19. Bui, D. D., & Liu, Q. (2018). IoT-based smart home automation using Arduino platform. *Journal of Electrical Engineering and Technology*, 13(1), 155-162. doi:10.5370/JEET.2018.13.1.155
20. Mohapatra, P., & Sahoo, B. (2020). IoT based energy efficient smart home automation system. *Procedia Computer Science*, 167, 2937-2946. doi:10.1016/j.procs.2020.03.305
21. Yang, Y., & Chen, X. (2019). IoT-based smart home system for energy optimization. *Journal of Electrical Engineering & Technology*, 14(3), 1127-1136. doi:10.5370/JEET.2019.14.3.1127
22. Liu, S., & Liu, T. (2018). A survey on energy-efficient smart home automation system. *Journal of Electronic Science and Technology*, 16(2), 123-130. doi:10.11989/JEST.1674-862X.703345
23. Shastri, M., & Patel, R. (2019). Smart home automation using IoT and cloud computing. *International Journal of Engineering & Technology*, 7(3), 1294-1300. doi:10.14419/ijet.v7i3.12452
24. Mohanty, S. P., & Kumar, S. (2016). Smart home automation using IoT. *Proceedings of the International Conference on Computing, Communication and Automation (ICCCA)*, 176-180. doi:10.1109/ICCCA.2016.7749989
25. Zhou, Y., & Wang, M. (2015). The development of smart home technology. *International Journal of Engineering Research & Technology*, 4(9), 1779-1783. doi:10.17577/IJERTV4IS090452
26. Yadav, S., & Tripathi, S. (2020). Smart home automation system with IoT and mobile application. *International Journal of Engineering and Technology*, 7(4), 1306-1312. doi:10.14419/ijet.v7i4.28788
27. Shaukat, M. A., & Khan, M. Z. (2017). Internet of things-based smart home automation system for elderly people. *IEEE Access*, 5, 11824-11833. doi:10.1109/ACCESS.2017.2729752
28. Patil, P., & Sawant, A. (2020). IoT based home automation system: A review. *International Journal of Engineering Research and Applications*, 10(9), 35-39. doi:10.9790/9622-1009013539
29. Li, Y., & Song, H. (2016). Smart home system based on the Internet of Things. *Proceedings of the International Conference on Smart City and Systems Engineering (ICSCSE)*, 146-150. doi:10.1109/ICSCSE.2016.76
30. Jain, A., & Kumar, M. (2018). Design of smart home systems using IoT technology. *International Journal of Engineering and Technology*, 7(2), 443-446. doi:10.14419/ijet.v7i2.11816
31. Kasim, R. M., & Yusof, R. M. (2018). Smart home automation using wireless sensors and IoT. *Journal of Computer Networks and Communications*, 2018, 1-10. doi:10.1155/2018/7893651
32. Ghosal, A., & Mukherjee, M. (2015). Energy efficient smart home automation system. *International Journal of Computer Applications*, 115(8), 42-48. doi:10.5120/ijca2015906907
33. Le, Q. B., & Pham, T. M. (2019). Real-time monitoring and control system for smart home automation. *Proceedings of the International Conference on Advanced Computing and Communication Systems (ICACCS)*, 120-124. doi:10.1109/ICACCS.2019.8728436
34. Kiran, M. S., & Shetty, A. (2020). Smart home automation and security system using IoT. *Proceedings of the 2nd International Conference on Advances in Computing, Communication and Control (ICACCC)*, 1-5. doi:10.1109/ICACCC49394.2020.9073977
35. Mishra, S. S., & Pradhan, A. (2018). IoT based smart home automation system. *International Journal of Advanced Research in Computer Science and Software Engineering*, 8(6), 234-237. doi:10.23956/ijarcsse.v8i6.1560
36. Ali, M. R., & Saha, S. (2019). Smart home automation: An IoT approach. *International Journal of Electrical and Electronics Engineering*, 7(3), 1-7. doi:10.1007/s11554-019-00843-1
37. Singh, H., & Arora, S. (2020). IoT-based smart home automation system with energy management. *International Journal of Engineering Research & Technology*, 9(6), 1120-1125. doi:10.17577/IJERTV9IS060369
38. Lee, D., & Hong, S. (2019). IoT-based smart home automation system using voice recognition. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1544-1549. doi:10.1109/ICRA.2019.8794091
39. Zhao, Y., & Liu, J. (2017). IoT-based smart home system design and implementation. *International Journal of Electrical Engineering Education*, 54(4), 345-357. doi:10.1177/0020720917698584
40. Gupta, P., & Soni, A. (2018). Smart home automation using IoT and mobile control. *Proceedings of the IEEE International Conference on Advanced Computing and Communication Systems (ICACCS)*, 129-133. doi:10.1109/ICACCS.2018.8492129
41. Sharma, M., & Gupta, S. (2020). Smart home automation and security system with IoT. *Journal of Computer Science & Technology*, 35(2), 400-410. doi:10.1007/s11390-020-2014-3
42. Zubair, A., & Raza, M. (2019). IoT based smart home automation system for energy management. *Journal of Engineering and Technology*, 12(4), 234-240. doi:10.14569/IJACSA.2019.0100475
43. Sharma, R., & Gupta, M. (2020). Development of smart home system using IoT for security and automation. *International Journal of Engineering and Advanced Technology*, 9(5), 4012-4017. doi:10.35940/ijeat.E3443.059520
44. Vyas, S., & Joshi, A. (2019). IoT-based energy-efficient smart home system. *International Journal of Engineering Science and Technology*, 11(6), 1087-1094. doi:10.5829/ijest.2019.11.6.1087
45. Patel, D., & Shah, P. (2018). Smart home automation using IoT and cloud. *Proceedings of the IEEE International Conference on Communication and Signal Processing (ICCSP)*, 299-303. doi:10.1109/ICCSP.2018.8682379
46. Nagarajan, R., & Suresh, A. (2019). Internet of Things-based smart home automation. *Proceedings of the International Conference on Advanced Computer Science and Information Technology (ACSIT)*, 99-103. doi:10.1109/ACSIT.2019.00019
47. Kaur, N., & Singh, A. (2017). Real-time smart home automation using IoT. *International Journal of Engineering Research & Technology*, 6(8), 487-493. doi:10.17577/IJERTV6IS080384
48. Arora, P., & Garg, S. (2020). Design and implementation of smart home automation system based on IoT. *International Journal of Computer Applications*, 9(4), 145-150. doi:10.5120/ijca2020921345
49. Pattanayak, B., & Satpathy, D. (2020). Smart home automation using IoT with security system. *Proceedings of the 4th International Conference on Computing, Communications, and Networking Technologies (ICCCNT)*, 1-5. doi:10.1109/ICCCNT49239.2020.9203960
50. Alam, M. R., Reaz, M. B. I., & Ali, M. A. M. (2012). A review of smart homes: Past, present, and future. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6), 1190-1203. doi:10.1109/TSMCC.2012.2189204
51. Mashal, I., Alsaryrah, O., Chung, T.-Y., Yang, C.-Z., Kuo, W.-H., & Agrawal, D. P. (2015). Choices for interaction with things on the Internet and underlying issues. *Ad Hoc Networks*, 28, 68–90. doi:10.1016/j.adhoc.2015.01.001
52. Ning, Z., Hu, X., & Wang, X. (2016). Smart home system based on ZigBee wireless sensor network. *Procedia Engineering*, 15, 29–32. doi:10.1016/j.proeng.2016.04.005
53. Singh, R., & Gupta, R. (2018). IoT based smart home automation system using sensor node and relay module. *International Journal of Advanced Research in Computer Science*, 9(4), 230-235.
54. Tran, K., Pham, T., & Phung, T. (2017). Energy management for the internet of things in smart homes. *IEEE Communications Magazine*, 55(3), 76-83. doi:10.1109/MCOM.2017.1500815
55. Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7), 1645-1660. doi:10.1016/j.future.2013.01.010
56. Perera, C., Zaslavsky, A., Christen, P., & Georgakopoulos, D. (2014). Context-aware computing for the Internet of Things: A survey. *IEEE Communications Surveys & Tutorials*, 16(1), 414-454. doi:10.1109/SURV.2013.042313.00197
57. Han, J., Choi, C.-S., & Lee, I. (2010). More efficient home energy management system based on ZigBee communication and infrared remote controls. *IEEE Transactions on Consumer Electronics*, 56(1), 58-62. doi:10.1109/TCE.2010.5439130
58. Minoli, D., Sohraby, K., & Occhiogrosso, B. (2017). IoT considerations, requirements, and architectures for smart buildings – Energy optimization and next generation building management systems. *IEEE Internet of Things Journal*, 4(1), 269-283. doi:10.1109/JIOT.2017.2647881
59. Alheraish, A. (2004). Design and implementation of home automation systems. *IEEE Transactions on Consumer Electronics*, 50(4), 1087-1092. doi:10.1109/TCE.2004.1362522