Design and Implementation of an IoT-Based Home Automation System with Security and Remote Monitoring Using NodeMCU and ESP32-CAM.

SHOPON KANTO DAS	**2132079007**
MD. RAHATUL ISLAM	**1912079003**
MD. JILLUR RAHMAN	**2132079003**

A Project Submitted to the

Department of Electrical and Electronic Engineering

Leading University, Sylhet.

In Partial Fulfillment of the Requirements for the Degree of

BACHELOR OF SCIENCE IN ELECTRICAL AND ELECTRONIC ENGINEERING

LEADING UNIVERSITY

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

LEADING UNIVERSITY

SYLHET-3112, BANGLADESH

MARCH 05, 2025

Design and Implementation of an IoT-Based Home Automation System with Security and Remote Monitoring Using NodeMCU and ESP32-CAM.

SHOPON KANTO DAS	**2132079007**
MD. RAHATUL ISLAM	**1912079003**
MD. JILLUR RAHMAN	**2132079003**

A Project Submitted to the

Department of Electrical and Electronic Engineering

Leading University, Sylhet.

In Partial Fulfillment of the Requirements for the Degree of

BACHELOR OF SCIENCE IN ELECTRICAL AND ELECTRONIC ENGINEERING

LEADING UNIVERSITY

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

LEADING UNIVERSITY

SYLHET-3112, BANGLADESH

MARCH 05, 2025

CERTIFICATE OF SUBMISSION

This is to certify that the title entitled **"Design and Implementation of an IoT-Based Home Automation System with Security and Remote Monitoring Using NodeMCU and ESP32-CAM"** submitted by the following students have been accepted as satisfactory in partial fulfilment of the requirement for the degree of Bachelor of Science in Electrical and Electronic Engineering on January 22, 2024.

SHOPON KANTO DAS	**2132079007**
MD. RAHATUL ISLAM	**1912079003**
MD. JILLUR RAHMAN	**2132079003**

Supervisor

S.M Tanbinul Hoque
Lecturer
Department of EEE
Leading University, Sylhet.

Co-Supervisor

Mirza Md. Mahbubur Rahman
Lecturer
Department of EEE
Leading University, Sylhet

Md. Niaz Morshedul Hoque
Assistant Professor and Head of Department
Department of Electrical and Electronic Engineering
Leading University, Sylhet.

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING (EEE)

LEADING UNIVERSITY
SYLHET-3112, BANGLADESH

DECLARATION

This is to clarify that work is the outcome of the accomplishment of the project on "Design and Implementation of an IoT-Based Home Automation System with Security and Remote Monitoring Using NodeMCU and ESP32-CAM" carried out by the student of the Department of Electrical and Electronic Engineering. Leading University, Sylhet, has not been submitted anywhere for any award or degree or published in any technical journal.

Signature of Candidates

Shopon Kanto Das
Student ID: 2132079007

Md. Rahatul Islam
Student ID: 1912079003

Md. Jillur Rahman
Student ID: 2132079003

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

LEADING UNIVERSITY
SYLHET-3112, BANGLADESH

ACKNOWLEDGEMENT

First and foremost, we appreciate the Most Gracious and Merciful Almighty. He showered us with blessings, providing us with courage and support during the most challenging parts of completing this dissertation.

For guidance, patience, and endless support, we sincerely thank the supervisor who motivated us. We sincerely thank our Supervisor, S.M Tanbinul Hoque, Lecturer, Department of Electrical & Electronic Engineering, Leading University.

Lastly, we thank our parents and dearest siblings for their unconditional love and continuous support. Without the mental strength you provided, surviving the journey full of ups and downs would not have been possible.

ABSTRACT

This project presents a study on the Design and Implementation of an IoT-Based Home Automation System with Security and Remote Monitoring Using NodeMCU and ESP32-CAM**.** The system aims to improve home security, automation, and real-time monitoring by integrating ESP8266 NodeMCU and ESP32-CAM with various IoT-based sensors. The study includes an in-depth analysis of existing home automation systems, their strengths and weaknesses, and the requirements for an effective IoT-based security and automation solution**.**

This system utilizes Wi-Fi communication to provide real-time remote control and monitoring of appliances such as lights, fans, and water pumps, ensuring improved user convenience. The ESP32-CAM-based security system, supported by dual PIR motion sensors, detects movement and captures images, which are instantly sent to the mobile app for real-time intrusion alerts. Additionally, the system includes gas leakage detection using the MQ-5 sensor, which triggers a buzzer and app notifications when hazardous gases are detected. The ADXL345 accelerometer provides seismic safety by detecting abnormal vibrations (earthquakes) and automatically shutting down all appliances to prevent potential hazards. Furthermore, the XKC-Y26-V water level sensor enables automatic water pump control, stopping the pump when the tank reaches a predefined level.

The system has been developed using state-of-the-art IoT technologies, including Wi-Fi-enabled ESP8266 NodeMCU, ESP32-CAM, and cloud-based remote monitoring via the Blynk app. The results of the implementation have been thoroughly evaluated, showing that the system is reliable and efficient in providing enhanced security, energy efficiency, and user convenience. The study concludes with a discussion on the potential impact of IoT-based home automation systems on improving safety and automation in modern households.

Keywords: IoT-based home automation, security monitoring, ESP8266 NodeMCU, ESP32-CAM, Blynk app, PIR motion sensor, ADXL345 accelerometer, MQ-5 gas sensor, XKC-Y26-V water level sensor, seismic detection, anti-theft system, remote monitoring, smart home, real-time alerts, automation, energy efficiency.

TABLE OF CONTENT

CHAPTER -4:

CHAPTER -5:

CHAPTER -6:

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Introduction.

1.2 Statement of the Problem.

1.3 Research Objectives.

1.4 Overview.

1.1 Introduction

The project "Design and Implementation of an IoT-Based Home Automation System with Security and Remote Monitoring Using NodeMCU and ESP32-CAM" aims to develop a smart home automation system that enhances security, convenience, and efficiency using IoT technologies. The system allows for remote control and real-time monitoring of home appliances while integrating security features such as motion detection, gas leakage alerts, and seismic safety shutdown.

In modern home automation, Wi-Fi-based IoT networks play a crucial role in enabling seamless communication between smart devices. This project utilizes ESP8266 NodeMCU and ESP32-CAM to provide a wireless automation system that ensures secure, real-time interaction between users and their home appliances. The system leverages Blynk cloud-based communication, allowing users to control and monitor devices from anywhere via a mobile application.

However, in cases where network disruptions or power failures occur, the system is designed to automatically trigger safety mechanisms. The ADXL345 accelerometer detects seismic activity and immediately shuts down all appliances to prevent electrical hazards. Similarly, the MQ-5 gas sensor monitors the environment for gas leaks, activating an alarm and sending real-time notifications to the user. The ESP32-CAM, combined with PIR motion sensor**s**, enhances security surveillance by capturing images upon detecting unauthorized motion and transmitting them to the user's mobile app.

This project involves comprehensive testing and validation to ensure that the system meets the desired performance, reliability, and safety standards. Various conditions and real-world scenarios will be simulated to assess the system’s effectiveness and identify areas for improvement.

The hardware components include ESP8266 NodeMCU, ESP32-CAM, PIR motion sensors, ADXL345 accelerometer, MQ-5 gas sensor, XKC-Y26-V water level sensor, and a relay module, all integrated for efficient automation and security. The software components involve programming through Arduino IDE and Blynk IoT platform, ensuring seamless data exchange and user control.

Overall, the implementation of Design and Implementation of an IoT-Based Home Automation System with Security and Remote Monitoring Using NodeMCU and ESP32-CAM will provide a reliable, secure, and user-friendly home automation solution. This system will enhance safety, optimize energy use, and offer smart control of home appliances, making daily life more convenient and efficient.

1.2 Statement of the Problem

The current home automation systems lack the ability to provide real-time monitoring and remote control of household appliances and security. This leads to inconvenience for users, who often have to manually operate appliances and have limited awareness of security threats at home. Additionally, existing systems do not provide an effective solution for automated safety responses, which can pose risks in cases of gas leaks, motion detection, or seismic activity.

Without a proper monitoring system, homeowners may face delays in responding to security threats or hazards, leading to potential safety risks and inefficient energy use. Additionally, the absence of automated control mechanisms means that appliances may remain turned on unnecessarily, increasing electricity consumption and operational costs.

To address these problems, a smart home automation system is needed that will provide real-time control and monitoring of home appliances, along with automated security and safety features. This will enable homeowners to remotely manage appliances, receive instant alerts, and improve home security. The system will also help reduce energy waste and provide a more efficient and reliable home management solution.

Therefore, the problem that this project aims to solve is the lack of real-time monitoring, remote accessibility, and automated safety features in existing home automation systems. By implementing a smart home automation solution**,** this project will improve home security, energy efficiency, and user convenience.

1.3 Research Objectives

Based on the problem statement above, this study is motivated by the following objectives which are:

To design and develop a smart home automation system for real-time remote control and monitoring of appliances.

To improve the efficiency of home management by reducing manual operation of appliances, optimizing energy usage, and enhancing convenience for users.

To enhance home security with real-time alerts for motion detection, gas leaks, and other hazards.

To provide a reliable system that offers instant notifications and automated safety responses.

To increase safety and convenience by enabling remote access and control via a smartphone app.

To reduce unnecessary power consumption by ensuring appliances are automatically controlled based on real-time data**,** helping to improve energy efficiency and cost savings**.**

1.4 Overview

The demand for smart home automation has grown significantly, leading to the integration of IoT-based systems for improved security, convenience, and energy efficiency. Traditional home management often relies on manual operation of appliances, which can result in unnecessary energy consumption and delays in responding to safety concerns. This project focuses on developing a real-time home automation system that provides remote access and monitoring of household appliances while incorporating automated safety features.

The system allows users to control home appliances remotely via a mobile application, ensuring efficient management of energy and security. It also includes automated responses to potential risks such as unauthorized movement, gas leaks, and environmental hazards, enhancing home

safety. Additionally, the system helps in optimizing power usage, leading to improved efficiency and cost savings.

By integrating smart automation with real-time monitoring, this project provides a comprehensive solution for modern home management. The ability to receive instant notifications and remotely control appliances ensures a safer, more efficient, and user-friendly home environment**.**

CHAPTER 2

LITERATURE REVIEW

2.1 Literature Review.

2.1 Literature Review

Home automation systems have significantly evolved with the advancement of IoT-based technologies**,** allowing for real-time monitoring, security automation, and energy-efficient control of home appliances**.** Several research studies have been conducted to explore different approaches to smart home automation**,** leading to improved solutions for remote access, safety features, and automation mechanisms. This literature review examines previous studies that have contributed to this field.

1. IoT-Based Home Automation Using NodeMCU and Blynk IoT App (2022).This study developed a Wi-Fi-based home automation system that enables users to control household appliances remotely using a mobile application. The system utilizes NodeMCU as the central controller, integrating sensors and relays to allow seamless communication between appliances and the Blynk IoT platform. The research demonstrated how mobile-based automation could improve convenience and efficiency in home management.

2. Gas Leak Detection, Monitoring, and Safety System Using IoT (2020).This study focused on developing a gas leakage detection system using IoT technology. The system incorporated gas sensors to monitor the presence of hazardous gases and send real-time alerts to users through a cloud-based notification system. By integrating wireless communication, the system provided instant safety warnings, allowing users to take prompt action to prevent accidents.

3. Arduino-Based Smart Security for Home Automation (2021).This research explored motion detection-based security systems for home automation. It integrated PIR motion sensors and a camera module to detect unauthorized movement within a designated area. The system was designed to capture images upon detection of motion, providing homeowners with real-time security updates. The study highlighted how IoT-based security mechanisms could enhance home safety and intrusion prevention.

4. Theft Detection System Using PIR Sensor (2020).This study developed an intrusion detection system for home security using PIR motion sensors. The system was designed to detect human movement and trigger an alert notification to the user. The research emphasized the effectiveness of motion sensors in smart security systems, providing an additional layer of protection for unauthorized access prevention.

5. Arduino-Based Home Automation Using IoT (2019).This study focused on Wi-Fi-enabled home automation, allowing users to control home appliances remotely using an IoT-based system. The research demonstrated how smart home solutions could be designed using Arduino-based controllers and wireless communication, providing an efficient alternative to traditional manual controls. The system aimed to improve energy efficiency and user convenience through automated appliance management.

6. An IoT-Based Home Automation Using Android Application (2020). This research explored the implementation of Android-based home automation, allowing users to control household devices through a mobile application. The system utilized IoT connectivity to establish

communication between home appliances and the Android app, enabling wireless control over lighting, fans, and other electrical devices. The study highlighted the advantages of smartphone-based automation in enhancing accessibility and ease of operation**.**

7. Arduino-Based Earthquake Detector Using an Accelerometer (2018).This study presented an earthquake detection system for home safety applications. The system utilized an accelerometer to detect seismic activity and activate a buzzer and LED indicator as an alert mechanism. The research emphasized the importance of early warning systems, showcasing how sensor-based solutions could improve home safety and preparedness by providing immediate alerts during seismic events.

8.The Development of an Earthquake Early Warning System Using an ADXL335 Accelerometer (2022).An earthquake early warning system was developed using the ADXL335 accelerometer to detect seismic vibrations and provide real-time alerts. The research focused on designing an effective detection system, calibrating the accelerometer for accuracy, and ensuring reliable data transmission to a central server. Additionally, challenges related to data loss and communication errors were addressed to improve system reliability. The findings highlighted the potential of 3-axis accelerometer sensors for monitoring motion, vibrations, and earthquake detection, contributing to early warning and safety applications.

9. QuakeGuard Insight: An IoT-Enabled Machine Learning System (2024).This study introduced an earthquake prediction system that integrates machine learning (ML) techniques with IoT technologies for seismic event detection. The system utilizes an ADXL335 accelerometer to record multi-axis acceleration data, which is then analyzed using various machine learning models, including decision trees and support vector machines (SVMs). The research focused on improving earthquake early warning capabilities by identifying seismic patterns through real-time sensor data processing. By integrating IoT infrastructure, the system ensures efficient data transmission and predictive analysis, helping communities and disaster management authorities prepare for seismic events. The findings demonstrated the potential of sensor-driven machine learning models in enhancing earthquake detection and mitigation strategies.

10. IoT-Based Smart Home Automation System (2021).This paper presents a comprehensive system that interconnects sensors, actuators, and other data sources to facilitate multiple home automation functions. The proposed system aims to enhance user convenience and energy efficiency by enabling seamless communication between devices. The architecture allows for real-time monitoring and control of household appliances, contributing to a more intelligent and responsive home environment.

11. IoT-Based Smart Intruder Detection System for Smart Homes (2021).An IoT-based intruder detection system was developed to enhance home security by detecting unauthorized entries and sending real-time alerts to homeowners. The system utilized a NodeMCU microcontroller with an ultrasonic sensor to monitor movement and detect intrusions. Data was processed and transmitted via Wi-Fi to the Blynk cloud server, which then triggered instant notifications to the user's smartphone. This system demonstrated the effectiveness of real-time monitoring, remote access, and cost-effective IoT security solution**s** for modern smart homes.

CHAPTER 3

METHODOLOGY

3.1 Introduction.

3.2 Methodology Flowchart.

3.3 System Architecture.

3.4 Sensor Data Collection and Transmission.

3.5 System Flowchart.

3.6 Block Diagram of the Working Principle of this Project.

Methodology

3.1 Introduction

The objective of this project is the Design and Implementation of an IoT-Based Home Automation System with Security and Remote Monitoring Using NodeMCU and ESP32-CAM. This system integrates remote appliance control, real-time security monitoring, and automated safety features, allowing users to manage household devices efficiently while ensuring safety. The core components include ESP8266 NodeMCU and ESP32-CAM, which facilitate seamless wireless communication between various sensors, actuators, and a cloud-based platform.

This section outlines the methodology adopted for the design, development, and testing of the system. The research follows a structured approach, which includes:

Identifying the research problem and objectives related to home automation and security.

Reviewing existing literature on IoT-based home automation systems.

Designing the system architecture, including hardware and software specifications.

Developing and implementing the system using embedded programming and cloud integration.

Testing and validating the system to ensure functionality, reliability, and efficiency.

The hardware development involves selecting and integrating PIR motion sensors, MQ-5 gas sensors, ADXL345 accelerometer, relay modules, and Wi-Fi-enabled controllers. The software development includes programming microcontrollers using Arduino IDE, integrating the Blynk IoT platform for real-time monitoring, and implementing automation logic for smart control.

The testing phase evaluates system performance, response time, and communication reliability under different conditions. Various test cases are conducted to verify motion detection accuracy, gas leak response, seismic activity detection, and remote appliance control functionality. Additionally, the Wi-Fi connectivity stability and data transmission accuracy are analyzed to ensure seamless operation.

This methodology section provides a step-by-step explanation of the research design, hardware and software implementation, and system evaluation techniques used in the project. It serves as a comprehensive guide to understanding the development process, technical contributions, and key findings of this IoT-based home automation system.

3.2 Methodology Flowchart:

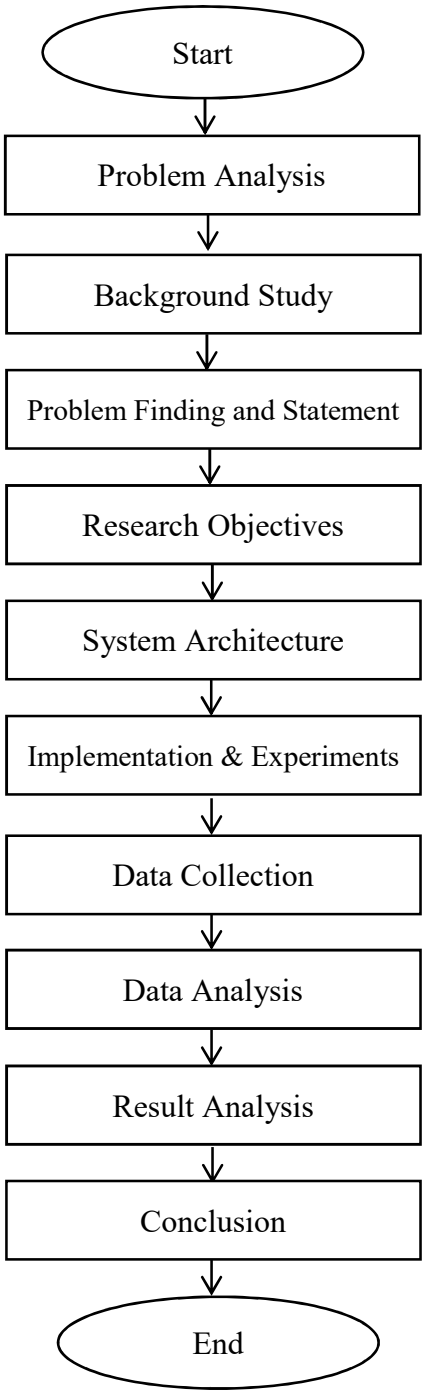

Figure 3.2: Methodology Flowchart

3.3 System Architecture

The system architecture is composed of the following components:

Hardware Components

ESP8266 NodeMCU: Serves as the main microcontroller, processing data and controlling appliances.

ESP32-CAM: Captures images upon motion detection for security monitoring.

PIR Motion Sensors: Detects movement and triggers security alerts.

MQ-5 Gas Sensor: Monitors gas leakage and activates an alert if dangerous levels are detected.

ADXL345 Accelerometer: Detects seismic activity and triggers an emergency shutdown of appliances.

XKC-Y26-V Water Level Sensor: Monitors the water tank level and automatically controls the water pump to prevent overflow.

Relay Module: Controls electrical appliances (light, fan, water pump) based on user commands.

Power Supply: Provides stable power to the microcontrollers and sensors.

Software Components

Arduino IDE: Used for programming microcontrollers and integrating sensors.

Blynk IoT Platform: Enables real-time remote control and monitoring via a smartphone app.

Embedded C Programming: Defines logic for sensor data processing and automation.

Communication & Connectivity

Wi-Fi Communication: Enables data transfer between ESP8266, ESP32-CAM, and the Blynk cloud server.

Cloud Integration: Stores sensor data and allows real-time access via the mobile app.

Hardware Setup

Connect PIR motion sensors, gas sensors, and accelerometers to the ESP8266 for security and safety monitoring.

Integrate ESP32-CAM with PIR sensors to capture images upon motion detection.

Connect relay modules to control appliances such as lights, fans, and water pumps.

Integrate the XKC-Y26-V Water Level Sensor with the system to monitor water tank levels and automate pump operation.

Ensure a stable power supply for uninterrupted operation.

Mobile Application (Blynk App)

User-friendly interface with features for real-time home monitoring and appliance control.

Receives real-time data from the system, displaying motion alerts, gas leak notifications, and appliance status.

Allows users to switch appliances ON/OFF remotely and receive emergency alerts.

3.4 Sensor Data Collection and Transmission

Data Collection:

The system continuously gathers real-time data from various sensors to monitor security, environmental conditions, and appliance status**:**

PIR Motion Sensor: Detects motion and sends alerts.

MQ-5 Gas Sensor: Detects gas leaks and transmits warning signals.

ADXL345 Accelerometer: Monitors vibrations for earthquake detection.

XKC-Y26-V Water Level Sensor: Measures water tank levels and prevents overflow.

ESP32-CAM: Captures images when motion is detected.

The ESP8266 NodeMCU microcontroller collects data from these sensors, processes it, and sends the information to the cloud server.

Data Transmission:

The ESP8266 NodeMCU acts as the central processor, handling data from sensors and relays for appliance control**.**

Data is transmitted via Wi-Fi to the Blynk cloud server for real-time monitoring.

Users can access the data and control appliances through the Blynk mobile app.

Server Setup:

A cloud-based server (Blynk IoT platform) is configured to store and process data.

The server updates the Blynk app in real-time, providing users with instant alerts and appliance status.

The system allows data analytics for security monitoring, helping users analyze motion detection history, gas leak trends, and water consumption.

Data Security:

The system uses secure protocols (SSL encryption**)** for communication between the microcontroller and the cloud.

The Blynk app and server require secure login authentication, ensuring only authorized users can access and control the system.

3.5 System Flowchart

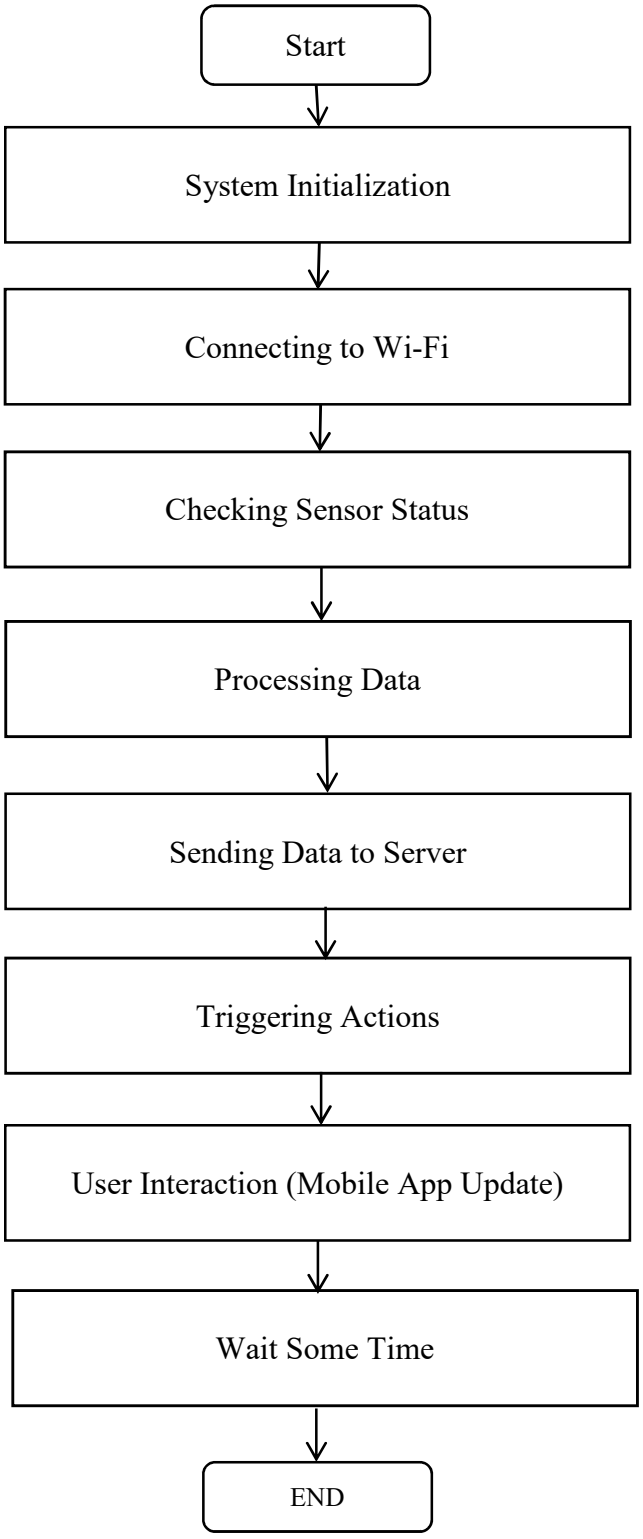

Fig 3.5: System Flowchart

This methodology shows the project working principle that how this project running. When the system is powered ON, the ESP8266 NodeMCU connects to the Wi-Fi network and initializes all sensors. Once connected, it begins collecting data from the PIR motion sensor, gas sensor, water level sensor, and accelerometer.

The processed data is then sent to the Blynk cloud server, where users can monitor real-time updates on the mobile app. If a trigger condition is met, the system activates relays, sends alerts, or captures images. After sending data, the system waits for a short interval before collecting the next set of readings.

The system follows a continuous cycle, checking sensor values at intervals. After each data transmission, it waits for a short period before collecting the next set of readings. This ensures real-time monitoring and efficient system operation.

3.6 Block Diagram of the Working Principle of this Project:

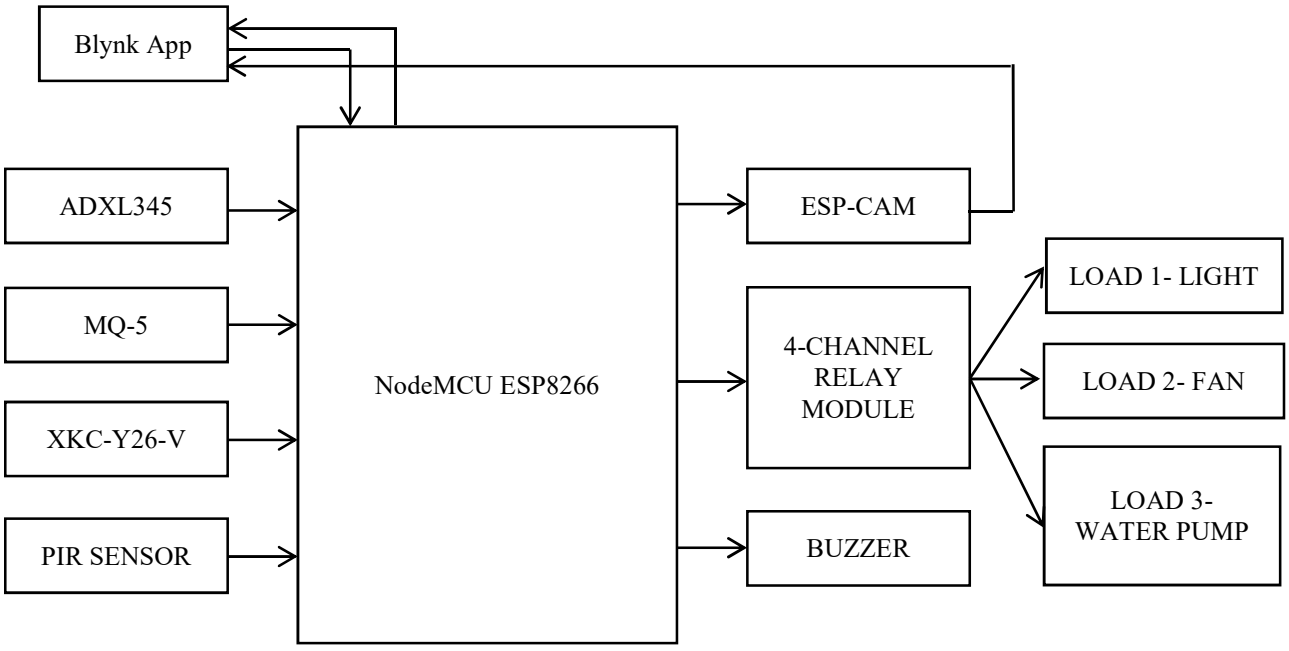

Figure 3.6: System Block Diagram

an embedded system is proposed for home automation with security and monitoring features**.** The system is designed to automate electrical appliances while integrating security sensors and real-time monitoring through a mobile application.

The NodeMCU ESP8266 microcontroller is used to interface with various hardware peripherals, including motion sensors, gas sensors, liquid level sensors, and a camera module. This embedded system continuously monitors sensor data and controls appliances based on user commands.

The ESP-CAM module captures images when motion is detected by the PIR sensors, enhancing security. The MQ-5 gas sensor detects harmful gases and sends alerts to the user via the Blynk app and a buzzer**.** Additionally, the ADXL345 accelerometer detects vibrations, while the XKC-Y26-V liquid level sensor monitors water levels.

The 4-channel relay module is responsible for controlling electrical loads such as lights, fans, and other appliances. All collected data is processed by ESP8266 and transmitted to the Blynk app, enabling remote monitoring and control.

Conclusion:

In conclusion, the proposed system provides an efficient and reliable IoT-based home automation solution with integrated security and remote monitoring. The system utilizes NodeMCU ESP8266 and ESP32-CAM to enable seamless control of home appliances and real-time surveillance through a mobile application. The combination of sensors, relays, and a camera module ensures both automation and security, making the system suitable for smart home applications.

The Design and Implementation of an IoT-Based Home Automation System with Security and Remote Monitoring Using NodeMCU and ESP32-CAM project thesis book has detailed the approach taken to design and implement the proposed system. The methodology began with a thorough literature review to analyze existing home automation technologies and their

limitations. This was followed by a comprehensive requirement analysis, identifying the functional and non-functional specifications necessary for system development.

To design the system, a combination of agile and waterfall methodologies was employed, ensuring iterative development, continuous testing, and stakeholder feedback. The implementation of the system was carried out using a combination of Arduino programming (C++), IoT protocols, and the Blynk platform for seamless communication between hardware and the user interface.

The testing phase included unit testing, integration testing, and user acceptance testing to validate the system's functionality. The evaluation demonstrated that the system successfully automates appliances, responds to app commands, and enhances home security through motion detection and real-time image capture using ESP32-CAM.

In summary, the methodology section has provided a detailed and systematic approach to the design, implementation, testing, and evaluation of the Home Automation System with Security Features. The methodology has ensured that the system meets stakeholder requirements and has been implemented using appropriate technologies. The results demonstrate the effectiveness of the system and its potential to enhance the security, automation, and convenience of home environments.

CHAPTER 4

SYSTEM REQUIRMENT

4.1 System Requirement

4.1.1 Arduino Compiler (IDE)

The Arduino IDE (Integrated Development Environment) is the software used to write, compile, and upload code to NodeMCU ESP8266 and ESP32-CAM. It supports **C** and C++ programming and provides an easy-to-use interface for developing home automation systems.

Key Features:

Code Editor: Allows writing and editing code.

Compiler: Converts code into instructions for the microcontroller.

Library Manager: Provides built-in libraries for Wi-Fi, Blynk, and sensors.

Board Manager: Supports multiple microcontrollers, including ESP8266 and ESP32-CAM.

Serial Monitor: Helps debug and monitor sensor data in real time.

For this project, the Arduino IDE is used to program the microcontroller, control appliances, process sensor data, and enable remote monitoring.

4.1.2 App Integration

The mobile application is a crucial part of the System, allowing users to remotely control and monitor home appliances. In this project, the Blynk app is used to provide a user-friendly interface for controlling devices connected to the NodeMCU ESP8266 and ESP32-CAM.

Development Process:

Purpose & Features: The app enables users to turn appliances ON/OFF, receive real-time status updates**,** and view security alerts from the PIR motion sensor and ESP32-CAM.

User Interface (UI) & Experience (UX): The Blynk app provides an intuitive UI with virtual buttons and widgets for seamless control.

Integration & Functionality: The app communicates with the Blynk Server, which then sends commands to the NodeMCU ESP8266 via the internet.When a user presses a button in the app, the server transmits data to the microcontroller, activating the corresponding appliance.

Testing & Deployment: The app is tested for responsiveness and real-time control performance before deployment.

The Blynk IoT architecture simplifies the app development process, eliminating the need for complex coding. The system ensures smooth wireless communication between the mobile app and home automation hardware, enhancing user convenience and security.

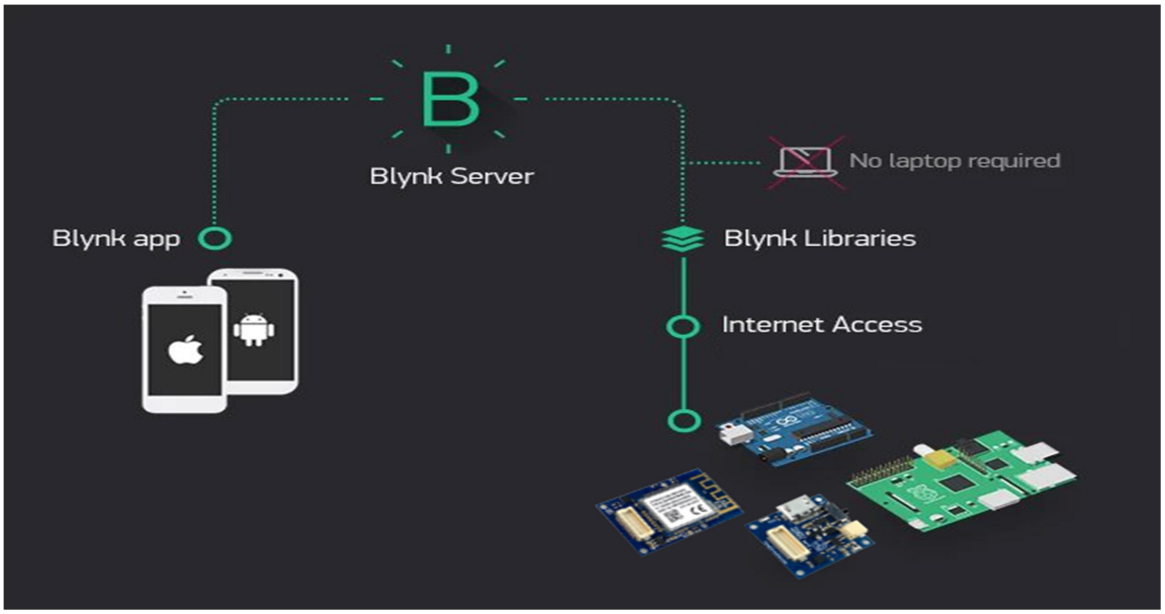

Fig 4.1.2: Blynk IoT architecture

4.2 Component Description

For designing this hardware, various electronic components and sensors are used to ensure the system functions correctly. These components are sourced from different manufacturers and assembled accordingly. The following list includes the key hardware components required for designing this system:

- **NodeMCU ESP8266 Microcontroller**
- **ESP32-CAM**
- **MQ-5 Gas Detector Sensor**
- **Contactless Water Level Sensor (XKC-Y26-V)**
- **HC-SR501 PIR Motion Sensor**
- **ADXL345 Triple-Axis Accelerometer**
- **Active Speaker Buzzer Module**
- **5V 4-Channel Relay Module**
- **Submersible 3V Mini DC Water Pump**
- **3V DC Motor With Fan**
- **LED**
- **Veroboard**
- **Wires**

4.2.1 NodeMCU ESP8266 Microcontroller

The ESP8266 is a low-cost, Wi-Fi enabled microcontroller chip that is widely used in Internet of Things (IoT) applications. It was first released in 2014 by the Chinese company Espressif Systems, and it quickly gained popularity due to its low cost, small size, and built-in wireless connectivity.

The ESP8266 microcontroller includes a 32-bit RISC CPU, 64KB of instruction RAM, and 96KB of data RAM. It also has 4MB of flash memory, which is used to store program code and other data. In addition, it includes a range of built-in peripherals, such as SPI, I2C, UART, and PWM interfaces, as well as analog-to-digital converters and general-purpose input/output (GPIO)pins.

One of the key features of the ESP8266 is its built-in Wi-Fi capability, which allows it to connect to wireless networks and communicate with other devices over the internet. This makes it well-suited for loT applications, where it can be used to control and monitor devices remotely.

The ESP8266 is also easy to program using a variety of different programming languages, including C, C++, Lua, and MicroPython. It can be programmed using a variety of different development environments, including the Arduino IDE, the ESP-IDF, and the NodeMCU firmware.

Overall, the ESP8266 microcontroller is a versatile and powerful platform for developing lot applications and other projects that require wireless connectivity. Its low cost, small size, and built-in Wi-Fi capability make it a popular choice for developers and hobbyists of all skill levels.

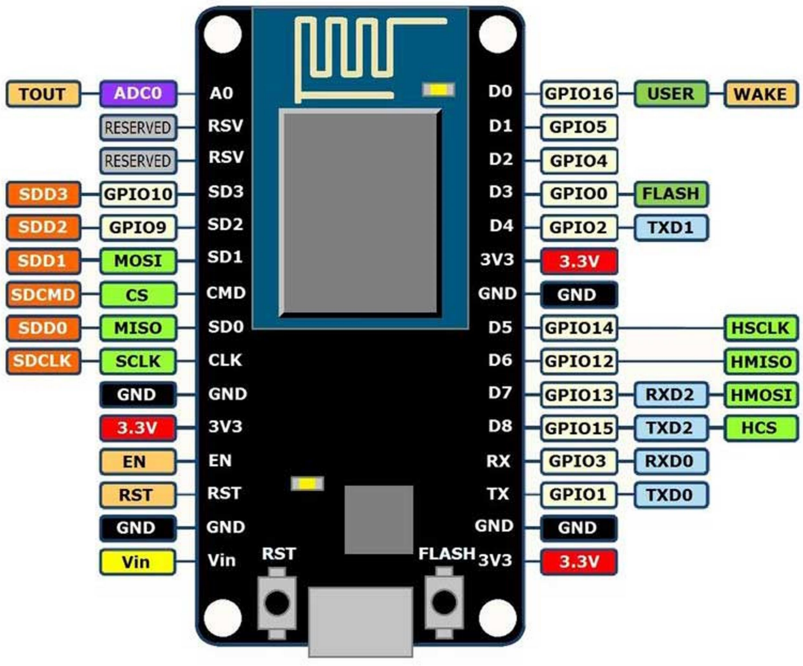

Figure 4.2.1: Pin Diagram of NodeMCU ESP8266 Microcontroller

Technical Specification:

Microcontroller	ESP-8266 32-bit
NodeMCU Model	Amica (official)
Clock Speed	80-160 MHz
USB to Serial	CP2102
USB Connector	Micro USB
Operating Voltage	3.3V
Input Voltage	4.5V-10V
Flash Memory/SRAM	4 MB / 128 KB
GPIO Pins	17
Digital I/O Pins	11
Analog In Pins	1
PWM Pins	4
ADC Range	0-3.3V
UART/SPI/I2C	2 / 1 / 1
WiFi Built-In	802.11 b/g/n
Temperature Range	-40°C to 125°C

Table 4.2.1: NodeMCU ESP8266 Microcontroller

4.2.2 ESP32-CAM

The ESP32-CAM is a compact, low-cost microcontroller module with built-in Wi-Fi, Bluetooth, and a camera interface, making it ideal for IoT-based security and surveillance applications. Developed by Espressif Systems, it is widely used in remote monitoring, facial recognition, and home automation.

The ESP32-CAM can be programmed using the Arduino IDE, ESP-IDF, and other development environments. It supports various libraries for image processing, face detection, and real-time streaming.

In this project, the ESP32-CAM is used for security monitoring. It captures images when motion is detected by the PIR sensors and sends alerts via the Blynk app. This enhances home security by allowing real-time surveillance through a smartphone.

With its compact size, wireless connectivity, and camera capabilities, the ESP32-CAM plays a crucial role in the IoT-based home automation system.

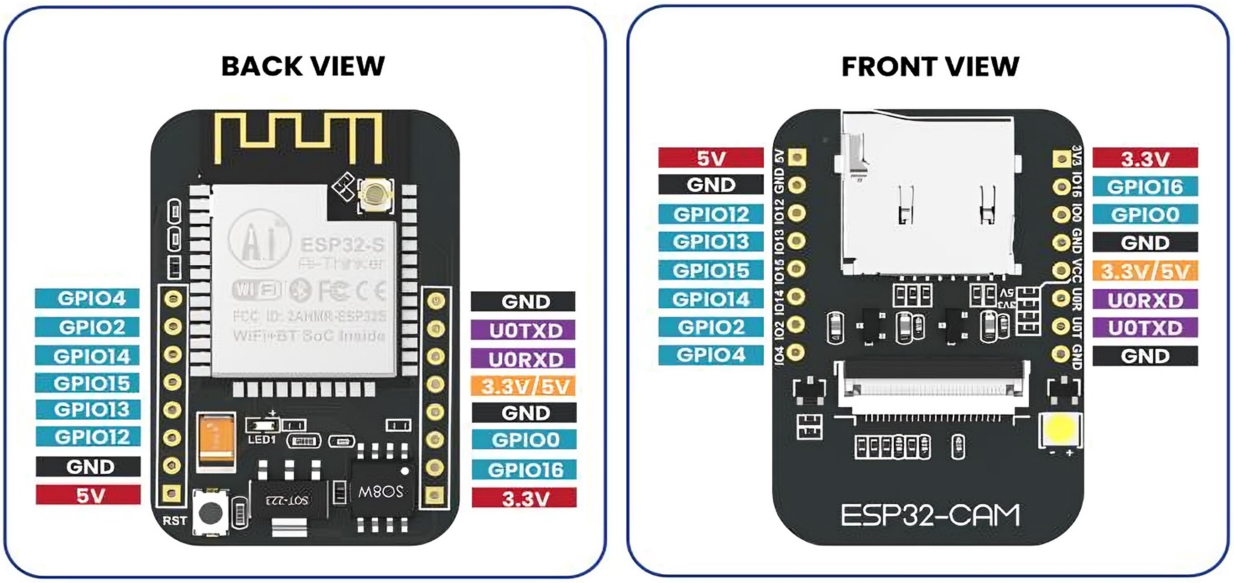

Figure 4.2.2: ESP32-CAM Pinout

Technical Specification:

Feature	Specification
Microcontroller	ESP32 Dual-Core Processor
Camera Module	2MP OV2640
Flash Memory	4MB
Wi-Fi Standard	802.11 b/g/n
Bluetooth	Bluetooth 4.2
GPIO Pins	Up to 10 usable GPIOs
Interfaces	UART, SPI, I2C, PWM, ADC
Storage	MicroSD Card Slot (Supports up to 4GB)
Operating Voltage	3.3V – 5V
Power Consumption	Low-power mode available
Programming Support	Arduino IDE, ESP-IDF

Table 4.2.2: ESP32-CAM Specification

4.2.3 MQ-5 Gas Detector Sensor

The MQ-5 Gas Sensor is a widely used semiconductor-based gas detection module that can detect LPG (liquefied petroleum gas), natural gas, and other combustible gases**.** It is commonly used in home automation and industrial safety systems to monitor gas leaks and trigger alarms.

The MQ-5 sensor consists of a heated tin dioxide (SnO_2) sensing element that changes its resistance in the presence of gas. When combustible gases are detected, the sensor produces a varying analog output voltage, which is processed by a microcontroller (such as ESP8266 in this project) to generate alerts.

In this project, the MQ-5 sensor is used to detect gas leaks in real time. If the gas concentration exceeds a predefined threshold, the system activates a buzzer and sends a notification to the Blynk app**,** alerting the user.

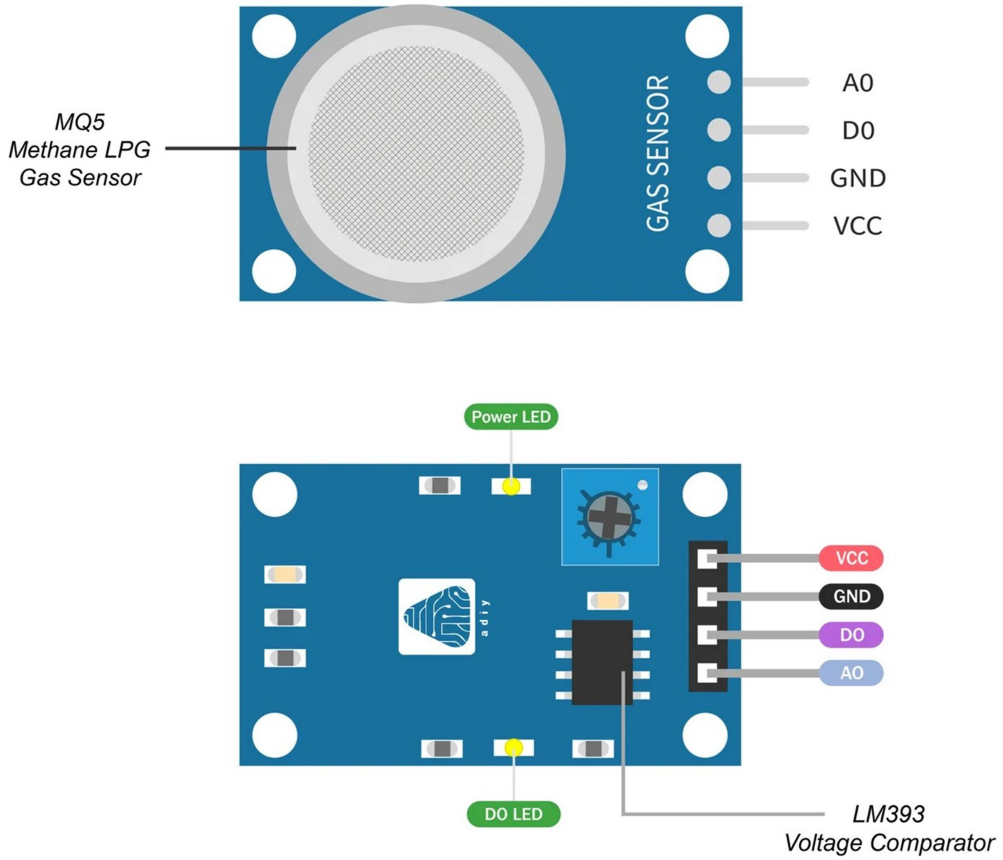

Figure 4.2.3: MQ5- Gas Sensor Module

Technical Specification:

Feature	Specification
Gas Detection	LPG, natural gas, coal gas
Operating Voltage	5V Operation
Power Consumption	~150 mA
Output Type	Analog & Digital
Preheat Time	≤ 60 seconds
Sensing Range	200 – 10,000 ppm
Sensitivity	High sensitivity to flammable gases
Response Time	≤ 10 seconds
Operating Temperature	-20°C to 40°C
Weight	7g

Table 4.2.3: MQ5- Gas Sensor Module Specification

4.2.4 Contactless Water Level Sensor (XKC-Y26-V)

The XKC-Y26-V is a non-contact liquid level sensor designed to detect the presence of liquid within non-metallic containers or pipelines without direct contact with the liquid. Utilizing advanced capacitive sensing technology, it offers a reliable solution for applications where traditional contact-based sensors are unsuitable, such as in detecting corrosive, toxic, or high-pressure liquids.

This sensor operates on the principle of capacitive sensing. When attached to the exterior surface of a non-metallic container, it detects changes in capacitance caused by the presence or absence of liquid on the inner side of the container wall. As the liquid level approaches the sensor's position, the capacitance increases, triggering the sensor to output a signal indicating the presence of liquid.

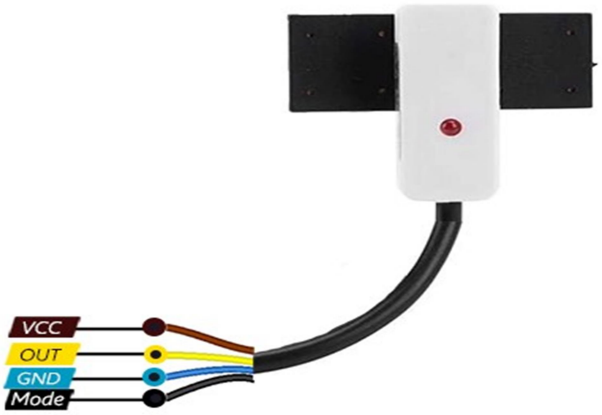

Figure 4.2.4: XKC-Y26-V Terminal

Technical Specification:

Feature	Specification
Operating Voltage	5V – 24V DC
Current Consumption	5mA
Response Time	500ms
Detection Distance	Up to 20mm (non-metallic containers)
Operating Temperature	0°C – 105°C
Output Voltage	High (Vin) / Low (0V)
Output Current	1-100mA
Material	ABS Plastic (Waterproof, IP67)

Table 4.2.4: XKC-Y26-V Water Level Sensor Specification

4.2.5 PIR Motion Sensor

A PIR (Passive Infrared) Sensor detects motion by sensing changes in infrared (IR) radiation emitted by objects, especially humans. It is widely used in home automation, security systems**,** and motion-activated lighting due to its low power consumption, reliability, and ease of use.The PIR sensor consists of a pyroelectric sensing element that detects infrared radiation. The sensor is divided into two halves, each measuring infrared (IR) radiation levels. **Static IR Levels:** If no movement occurs, both halves receive the same IR radiation, and the sensor output remains stable (LOW)**.** **Motion Detection:** When a warm object (e.g., a human) moves across the sensor's field of view, one half detects an increase in IR radiation before the other. This **change in infrared levels** generates a voltage difference, causing the sensor output to **go HIGH**, signaling motion detection.

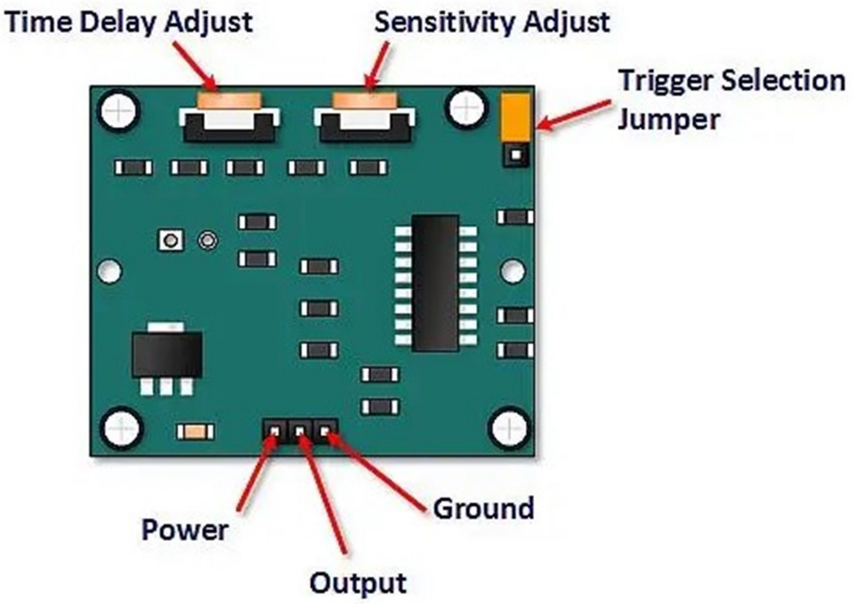

Figure 4.2.5.1: PIR Sensor Pinout (Back Side of HC-SR501 Module)

Technical Specification:

Feature	Specification
Model	HC-SR501
Operating Voltage	4.5V – 20V DC
Current Consumption	60µA (standby)
Detection Range	3 – 7 meters (can be adjusted)
Detection Angle	110°
Output Voltage	HIGH: 3.3V / LOW: 0V
Default Retrigger Time	2.5 seconds

Table 4.2.5: PIR Motion Sensor Specification

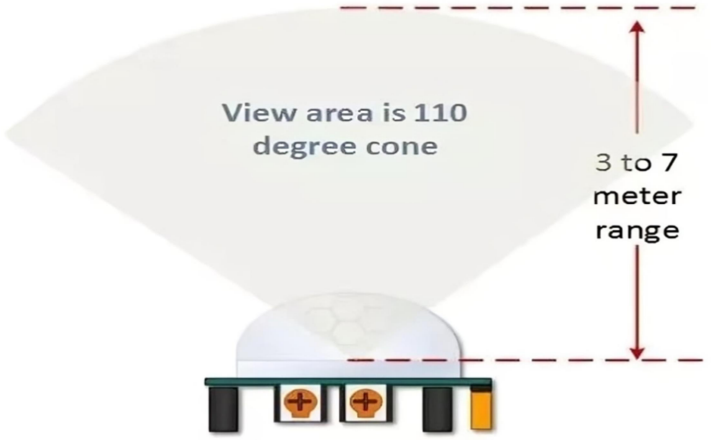

Figure 4.2.5.2: HC-SR501 Area of Detection

4.2.6 ADXL345 Triple-Axis Accelerometer

The ADXL345 is a compact, low-power 3-axis accelerometer**.** It provides high-resolution (13-bit) measurements of acceleration in X, Y, and Z axes, making it ideal for applications such as earthquake detection, fall detection, and smart automation systems.

The ADXL345 operates on the principle of microelectromechanical systems (MEMS**)**, which use tiny mechanical structures inside the sensor to measure acceleration.

Capacitive Sensing Mechanism: The sensor contains a microscopic suspended mass between fixed plates. When the sensor moves, the mass shifts slightly due to inertia, changing the capacitance between the plates.

Analog to Digital Conversion: These capacitance changes are converted into analog voltage signals**.**The internal ADC (Analog-to-Digital Converter) processes these signals into digital acceleration data.

Output Data & Communication (I²C or SPI Protocols): The ADXL345 supports two communicationinterfaces:

I²C (Inter-Integrated Circuit): Uses only two lines (SDA for data and SCL for clock) to communicate with the microcontroller. Efficient for connecting multiple devices on the same bus.

SPI (Serial Peripheral Interface): Uses four lines (MOSI, MISO, SCLK, and SS) for faster data transmission. Suitable for high-speed applications but requires more wiring.

The ADXL345 communicates with NodeMCU ESP8266 using I²C to efficiently transmit real-time acceleration data for earthquake detection.

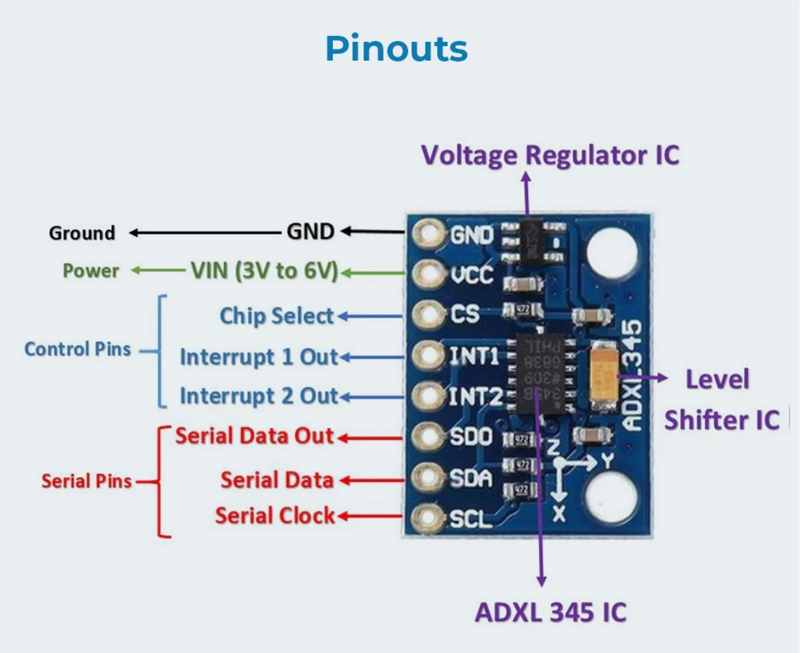

Figure 4.2.6: ADXL345 Pinouts

Technical Specification:

Feature	Specification
Model	ADXL345
Supply Voltage	3.3V to 5V DC
Sensing Range	±2g, ±4g, ±8g, ±16g
Resolution	13-bit
Sensitivity:	X-axis: 28.6 LSB/g, Y-axis: 31.2 LSB/g, Z-axis: 34.5 LSB/g
Power Consumption	Measurement Mode: 40µA, Standby Mode: 0.1µA at 2.5V
Operating Temperature Range	-40°C to +85°C
Shock Survival	Up to 10,000 g

Table 4.2.6: ADXL345 Triple-Axis Accelerometer Specification

4.2.7 5V 4-Channel Relay Module

The 5V 4-Channel Relay Module is an electromechanical switching device used to control high-power electrical appliances using low-power signals from a microcontroller like NodeMCU ESP8266. It provides electrical isolation between the control circuit (low voltage DC) and the load circuit (high voltage AC/DC).

When power is supplied to the relay, current flows through the coil, energizing the electromagnet and attracting the armature, which closes the circuit. When power is removed, the spring returns the armature, breaking the connection. This allows a low-power signal from a microcontroller to control high-power electrical loads.

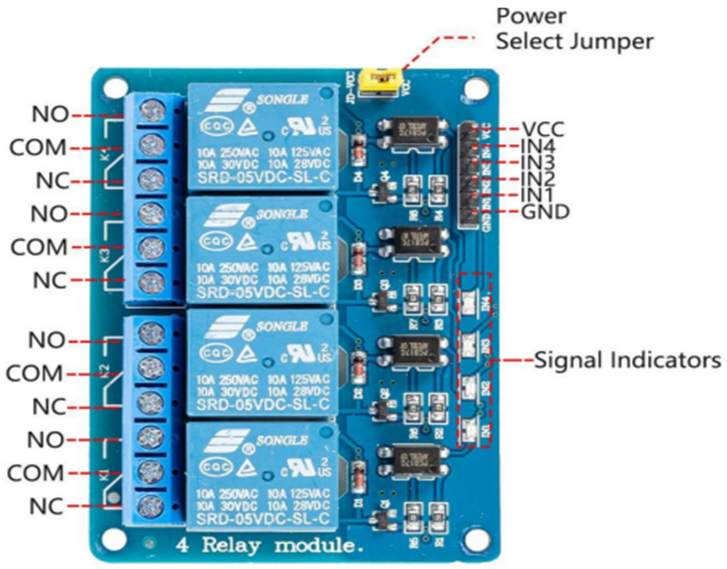

Figure 4.2.7: 5V-4 Channel Relay Module

Technical Specification:

Feature	Specification
Operating Voltage	5V DC
Driver Current	15-20mA per channel
Relay Switching Voltage	AC 250V / DC 30V
Relay Switching Current	10A
Control Signal	LOW-level trigger
Number of Channels	4
Isolation	Optocoupler isolation for protection

Table 4.2.7: 5V 4-Channel Realy Module Specification

CHAPTER 5

SYSTEM DESIGN AND IMPLEMENTATION

5.1 Hardware Design.

5.1.1 Gas Detector .

5.1.1 Gas Detector

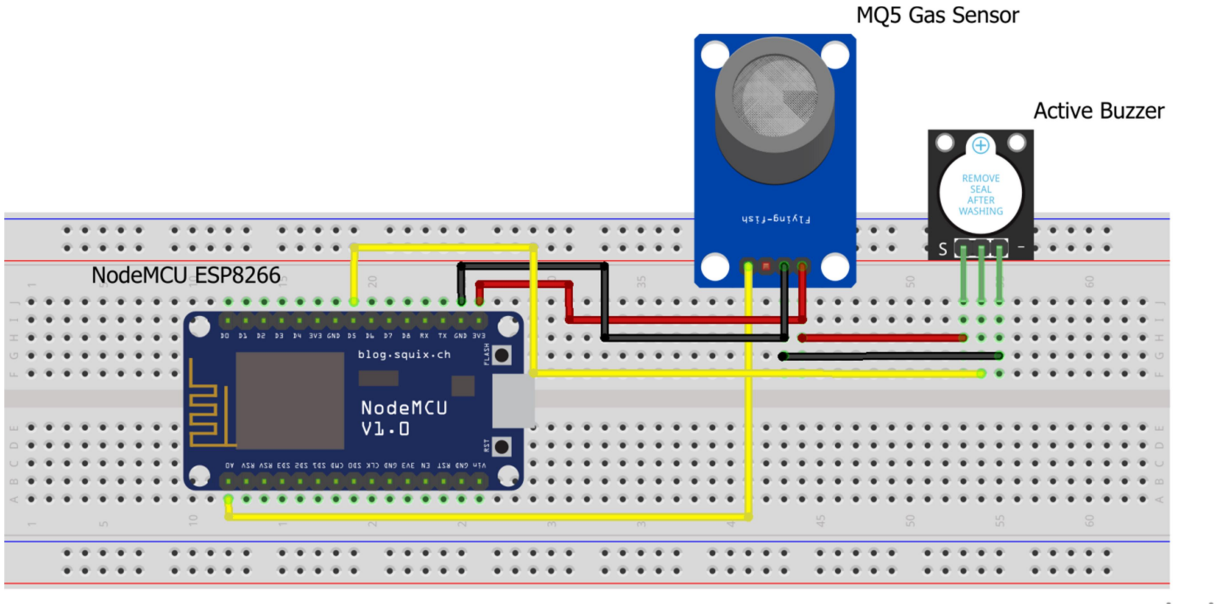

Figure 5.1.1: Gas Detector Connection

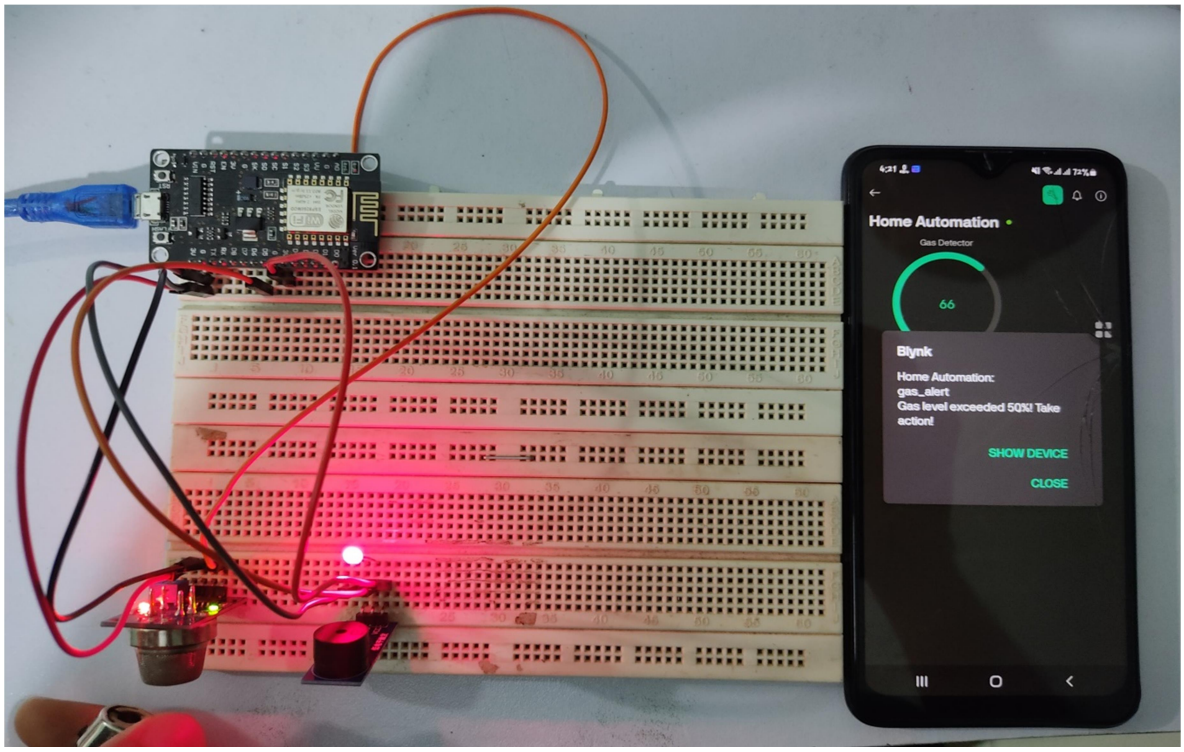

Figure 5.1.2: Gas Detector Implementation

CHAPTER 6

PROBLEM, LIMITATION AND ADVANTAGES

6.1 Problem and Limitations.

6.1.1 Hardware Limitations.

6.1.2 Communication and Network Issues.

6.1.3 Software & Coding Issues.

6.1.4 Power supply Limitations.

6.1.5 User and Installing Challenges.

6.2 Advantage.

6.3 Cost Analysis.

6.1 Problems and Limitations

During the development of this Automation System, various challenges were encountered related to hardware components, communication modules, and software development. Each component has its own characteristics, which sometimes led to unexpected results and required troubleshooting.

6.1.1 Hardware Limitations

1. Sensor and Module Limitations:

(A) PIR Motion Sensor:

False Triggers: PIR sensors are sensitive to temperature changes and can be affected by environmental factors such as direct sunlight, heating systems, or sudden airflow changes.

Limited Detection Range: The effective detection range is 3-7 meters, which may not be suitable for large rooms or outdoor areas.

(B) MQ-5 Gas Detector Sensor:

Response Time: The sensor requires a preheat time of up to 60 seconds before it can detect gas leaks accurately.

Environmental Sensitivity: Humidity and temperature fluctuations can affect the sensor's accuracy**.**

(C) Contactless Water Level Sensor (XKC-Y26-V):

Non-Metallic Limitation: The sensor only works with non-metallic containers, limiting its use in certain applications.

Limited Sensing Thickness: It can only detect liquid through walls up to 20mm thick**.**

(D) ADXL345 Accelerometer (Earthquake Detection):

Vibration Sensitivity: The accelerometer may detect false vibrations due to surrounding movements, requiring precise threshold calibration**.**

Power Consumption: Continuous real-time monitoring consumes power, which can affect battery-powered systems**.**

(E) 5V 4-Channel Relay Module:

Mechanical Wear: Relays have moving parts, which can wear out over time, reducing reliability**.**

Delay in Switching: There is a small delay in relay activation compared to solid-state switches.

6.1.2 Communication and Network Issues

1. Wi-Fi Connectivity (NodeMCU ESP8266 & ESP32-CAM):

Signal Interference: Wi-Fi performance degrades due to obstacles, interference, or weak signals in large houses.

Limited Range: The ESP8266 has a limited range (~30 meters indoors), requiring a strong router or Wi-Fi extender.

2. Mobile App (Blynk) Limitations:

Internet Dependency: The system relies on a stable internet connection for remote monitoring.

Cloud Dependency: Blynk stores data in the cloud, which means server downtimes may affect accessibility**.**

6.1.3 Software & Coding Issues

1. Arduino Code Compilation & Errors:

Syntax Errors: Mistakes in writing code (e.g., missing semicolons, incorrect function calls) lead to compilation failures.

Memory Limitations: Microcontrollers like NodeMCU ESP8266 have limited RAM and flash memory, restricting complex operations.

Pin Configuration Issues: Incorrect pin assignments or conflicts between multiple components (e.g., ESP32-CAM and sensors) may cause malfunctions.

2. Library Conflicts & Compatibility Issues:

Multiple Libraries: Using many Arduino libraries may cause conflicts, requiring manual debugging.

Version Compatibility: Some libraries may not support the latest updates, requiring adjustments in the code.

6.1.4 Power Supply Limitations

Current Demand: Some sensors (e.g., ESP32-CAM) require higher current, needing a stable 5V 2A power supply**.**

Power Interruptions: If power fluctuates, the system may restart unexpectedly, affecting automation reliability.

6.1.5 User and Installation Challenges

Technical Knowledge: Users with limited technical expertise may find it difficult to troubleshoot or modify the system.

Initial Setup Complexity: Installing and configuring the ESP32-CAM, relay module, and Blynk app requires proper gidnace.

Despite these challenges and limitations**,** the Home Automation System successfully provides remote monitoring, security, and automation features. Most of these issues can be resolved through careful calibration, optimization, and future improvements**.**

6.2 Advantages

The IoT-Based Home Automation System with Security and Remote Monitoring offers several advantages in terms of convenience, energy efficiency, security, and remote accessibility**.** This system enhances the way users interact with their home appliances while ensuring safety and automation**.**

1. Remote Monitoring and Control:

- Users can control and monitor home appliances such as lights, fans, and sockets from anywhere using the Blynk app.
- Real-time notifications provide updates on motion detection, gas leaks, and water level alerts, ensuring better safety and management.

2. Enhanced Home Security:

- The integration of ESP32-CAM allows for live surveillance and intrusion detection, making homes more secure.
- PIR motion sensors detect unauthorized movement, automatically triggering alerts and capturing images via the camera.
- The system can send instant alerts to the user's mobile app, reducing security risks.

3. Energy Efficiency & Cost Savings:

- The system helps reduce unnecessary power consumption by automating appliances based on real-time data.

- Relay-controlled automation ensures appliances are turned off when not in use, preventing energy wastage.
- Smart control of appliances lowers electricity bills and promotes an energy-efficient lifestyle.

4. Automatic Earthquake Safety Shutdown:

The ADXL345 accelerometer detects seismic activity and automatically shuts down all appliances when vibrations exceed a predefined threshold, preventing electrical hazards. This feature enhances safety during earthquakes, minimizing fire risks caused by electrical faults.

5. Smart Water and Gas Monitoring:

- The XKC-Y26-V contactless water level sensor ensures better water usage management by detecting tank levels and preventing overflows.
- The MQ-5 gas detector sensor detects gas leaks and sends alerts, helping to prevent fire hazards in the home.

6. Easy Integration & User-Friendly Operation:

- The system is easy to set up and can be expanded to include additional features such as voice control, AI automation, and more sensors.
- The Blynk app provides a simple and intuitive interface, making it accessible even for users with limited technical knowledge.

The IoT-Based Home Automation System enhances security, efficiency, and user convenience by allowing remote control, real-time alerts, and automated safety features. The integration of sensors, smart relays, and monitoring devices ensures a modern, safe, and energy-efficient home environment.

6.3 Cost Analysis

The cost analysis for the System includes the expenses for hardware components, software tools, and additional accessories required for implementation.

SL	Component Name	Quantity	Unit Price (BDT)	Total Price (BDT)
01	NodeMCU ESP8266	1	450	450
02	ESP32-CAM with Mounting Accessories	1	1495	1495
03	MQ-5 Gas Detector Sensor	1	195	195
04	Contactless Water Level Sensor (XKC-Y26-V)	1	1150	1150
05	HC-SR501 PIR Motion Sensor	2	93	186
06	ADXL345 Triple-Axis Accelerometer	1	468	468
07	5V 4-Channel Relay Module	1	270	270
08	Active Speaker Buzzer Module	1	49	49
09	Veroboard	1	150	150
10	5V 2A Power Adapter	1	300	300
11	Submersible 3V Mini DC Water Pump	1	140	140
12	Others		300	300
	Total Cost (BDT)			4703 TK

Additional Costs (If Applicable):

1. **Software & Tools:** Blynk App, Arduino IDE (Free)
2. **PCB Fabrication (if used)**
3. **3D Printing for Casing (if applicable)**
4. **Miscellaneous (Screws, Wires, Soldering Materials, etc.)**

This cost analysis provides a detailed breakdown of the hardware expense**s** for the IoT-Based Home Automation System**.** The total cost depends on component prices, availability, and optional enhancements like custom PCB design or additional sensors**.**

CHAPTER 7

FUTURE SCOPE AND CONCLUSION

7.1 Future Enhancement

7.2 Conclusion

Reference

7.1 Future Enhancements

The IoT-Based Home Automation System has significant potential for further development. As smart home technology evolves, several enhancements can be introduced to improve its efficiency, security, and automation capabilities.

1. AI and Machine Learning for Smart Automation:

- Implementing AI-based predictive automation to learn user behavior and automatically control appliances based on usage patterns.
- Using machine learning algorithms to optimize energy consumption and predict faults **in** electrical systems.

2. Voice and Gesture-Based Control:

- Integrating voice control through platforms like Google Assistant, Amazon Alexa, or Apple Siri to enhance user convenience.

3. Advanced Security and Surveillance:

- Upgrading ESP32-CAM with AI-powered face recognition **for** secure access control.
- Adding biometric authentication **for** enhanced home security.

4. Cloud-Based Data Storage and IoT Expansion:

- Using cloud services (AWS, Firebase, Google Cloud) for real-time storage and historical data analytics.
- Expanding the system to support multiple homes under one centralized platform, making it scalable for smart communities.

5. Renewable Energy Integration:

- Integrating the system with solar energy monitoring to optimize power consumption.
- Using automated power management to switch between grid power and renewable energy sources for improved energy efficiency.

6. Smart Fire and Gas Leakage Prevention:

- Implementing multi-sensor fusion (combining temperature, smoke, and gas sensors) for improved fire and gas leakage detection.
- Developing an automated emergency response system that triggers fire suppression mechanisms and notifies emergency services.

7.2 Conclusion

The IoT-Based Home Automation System with Security and Remote Monitoring Using NodeMCU and ESP32-CAM presents a cost-effective and user-friendly solution for smart home automation. The integration of ESP32-CAM, NodeMCU ESP8266, PIR motion sensors, relays, gas detectors, and accelerometers allows users to remotely monitor, control, and secure their homes.The project successfully automates appliances, enhances security, and improves safety through features Like:

Remote control via the blynk app

Intrusion detection with ESP32-CAM and PIR sensor

Gas leak monitoring using MQ-5

Earthquake detection with ADXL345 and automatic shutdown

Despite certain limitations such as network dependency and sensor accuracy variations, the system demonstrates a practical approach to modern home automation.

With the integration of AI, renewable energy, and cloud-based analytics, this project has the potential to evolve into a fully autonomous smart home system, offering greater efficiency, security, and sustainability.

Reference

[1] T. Deenadayalan, J. J. Shirley, R. Prajapati, P. Singh, A. Biswas, and U. Soni, "An IoT-based Home Automation System using NodeMCU and Blynk IoT App," 2023 International Conference on System, Computation, Automation and Networking (ICSCAN), Pondicherry, India, 2023, pp. 1-7. [Online]. Available: https://ieeexplore.ieee.org/document/10394759/

[2] P. Kalpana, R. Vignesh, R. Sakthi Vignesh, and K. R. Saanjeev Kumar, "Gas Leak Detection, Monitoring, and Safety System Using IoT," International Journal of Recent Technology and Engineering (IJRTE), vol. 8, no. 6, March 2020. ISSN: 2277-3878 (Online). DOI: 10.35940/ijrte.F9041.038620. [Online]. Available: https://www.ijrte.org/

[3] T. Akilan, R. Srivastava, C. Chandraprabha, B. Ahmad, H. Islam, and M. Saif, "Arduino-Based Smart Security for Home Automation," 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), Greater Noida, India, 2021, pp. 804-809. ISBN: 978-1-6654-3811-7. [Online]. Available: https://ieeexplore.ieee.org/document/9725436.

[4] P. N. Saranu, G. Abirami, S. Sivakumar, M. RameshKumar, U. Arul, and J. Seetha, "Theft Detection System using PIR Sensor," 2018 4th International Conference on Electrical Energy Systems (ICEES), Chennai, India, 2018, pp. 656-660. ISBN: 978-1-5386-3695-4. [Online]. Available: https://ieeexplore.ieee.org/document/8443215.

[5] L. M. Satapathy, S. K. Bastia, and N. Mohanty, "Arduino Based Home Automation Using Internet of Things (IoT)," International Journal of Pure and Applied Mathematics, vol. 118, no. 17, pp. 769-778, 2018. [Online]. Available: http://www.ijpam.eu.

[6] P. S. N. Reddy, K. T. K. Reddy, P. A. K. Reddy, G. N. K. Ramaiah, and S. N. Kishor, "An IoT based Home Automation Using Android Application," in Proc. Int. Conf. Signal Process., Commun., Power Embedded Syst. (SCOPES), 2016, pp. 1-5. [Online]. Available: https://ieeexplore.ieee.org/document/9318084

[7] G. Naik S, A. R, A. D, A. M.D, and A. P, "Arduino Based Earthquake Detector using Accelerometer," International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, vol. 9, no. 7, pp. 2005-2008, July 2020. [Online]. Available: https://www.ijareeie.com/upload/2020/july/40_ARDUINO_NC.PDF

Appendix

A. Code:

ESP8266 code without accelerometer:

```
#define BLYNK_TEMPLATE_ID "TMPL6Wgb60DP3"
#define BLYNK_TEMPLATE_NAME "Home Automation"
#define BLYNK_AUTH_TOKEN "KNXVCEfcQXwUXzgq1s6ZVECpU63spCec"

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <Wire.h>

#define BLYNK_PRINT Serial

// WiFi Credentials
char auth[] = "KNXVCEfcQXwUXzgq1s6ZVECpU63spCec";
char ssid[] = ""; //user wifi ssid
char pass[] = "";  //user wifi password

// Relay & Sensor Pins
#define RELAY_1 D0
#define RELAY_2 D1
#define RELAY_3 D2  // Motor relay
#define RELAY_4 D3
#define BUZZER_PIN D5
#define WATER_SENSOR D7
#define GAS_SENSOR_PIN A0

BlynkTimer timer;

void setup() {
  Serial.begin(9600);

  // Set relay pins as output
  pinMode(RELAY_1, OUTPUT);
  pinMode(RELAY_2, OUTPUT);
  pinMode(RELAY_3, OUTPUT);
  pinMode(RELAY_4, OUTPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(WATER_SENSOR, INPUT);
  pinMode(GAS_SENSOR_PIN, INPUT);

  // Default all relays OFF
```

```
  digitalWrite(RELAY_1, HIGH);
  digitalWrite(RELAY_2, HIGH);
  digitalWrite(RELAY_3, HIGH);
  digitalWrite(RELAY_4, HIGH);
  digitalWrite(BUZZER_PIN, LOW);

  // Connect to Blynk
  Blynk.begin(auth, ssid, pass, "blynk.cloud", 80);

  // Timer functions
  timer.setInterval(5000L, sendGasData);      // Check gas sensor every 1 sec
  timer.setInterval(5000L, waterLabelSensor); // Check water sensor every 1 sec
}

// □ Blynk Button Control for Relays
BLYNK_WRITE(V0) {
  digitalWrite(RELAY_1, param.asInt() ? LOW : HIGH);
}
BLYNK_WRITE(V1) {
  digitalWrite(RELAY_2, param.asInt() ? LOW : HIGH);
}
BLYNK_WRITE(V2) {
  digitalWrite(RELAY_3, param.asInt() ? LOW : HIGH);
}
BLYNK_WRITE(V3) {
  digitalWrite(RELAY_4, param.asInt() ? LOW : HIGH);
}

// □ Gas Sensor Function
void sendGasData() {
    int gasValue = analogRead(GAS_SENSOR_PIN);
    int gasPercentage = map(gasValue, 0, 1023, 0, 100);

    Blynk.virtualWrite(V5, gasPercentage);
    Serial.print("Gas Level: ");
    Serial.print(gasPercentage);
    Serial.println("%");

    if (gasPercentage >= 50) {
        digitalWrite(BUZZER_PIN, HIGH);  // Turn ON buzzer if gas > 50%
        Blynk.logEvent("gas_alert", "Gas level exceeded 50%! Take action!");
    } else {
        digitalWrite(BUZZER_PIN, LOW);   // Turn OFF buzzer
    }
}
```

```
// □ Water Level Sensor Function
void waterLabelSensor() {
  int waterValue = digitalRead(WATER_SENSOR);
  Serial.print("Water Level: ");
  Serial.println(waterValue);

  if (waterValue == HIGH) {
    digitalWrite(RELAY_3, HIGH);  // Turn OFF motor relay
    Serial.println("□ Water detected! Motor OFF!");
  } else {
    digitalWrite(RELAY_3, LOW);   // Motor can run
  }

  Blynk.virtualWrite(V6, waterValue);
}

void loop() {
  Blynk.run();
  timer.run();
}
```

ESP32 cam :

```
Board: AI-Thinker ESP32-CAM

*/

#include "src/OV2640.h"
#include <WiFi.h>
#include <WebServer.h>
#include <WiFiClient.h>

// Select camera model
//#define CAMERA_MODEL_WROVER_KIT
//#define CAMERA_MODEL_ESP_EYE
//#define CAMERA_MODEL_M5STACK_PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE
#define CAMERA_MODEL_AI_THINKER

#include "camera_pins.h"


#define SSID1 ""//Enter your WIFI name
#define PWD1 ""//Enter your WIFI password
```

```
OV2640 cam;

WebServer server(80);

const char HEADER[] = "HTTP/1.1 200 OK\r\n" \
                      "Access-Control-Allow-Origin: *\r\n" \
                      "Content-Type: multipart/x-mixed-replace;
boundary=123456789000000000000987654321\r\n";
const char BOUNDARY[] = "\r\n--123456789000000000000987654321\r\n";
const char CTNTTYPE[] = "Content-Type: image/jpeg\r\nContent-Length: ";
const int hdrLen = strlen(HEADER);
const int bdrLen = strlen(BOUNDARY);
const int cntLen = strlen(CTNTTYPE);

void handle_jpg_stream(void)
{
  char buf[32];
  int s;

  WiFiClient client = server.client();

  client.write(HEADER, hdrLen);
  client.write(BOUNDARY, bdrLen);

  while (true)
  {
    if (!client.connected()) break;
    cam.run();
    s = cam.getSize();
    client.write(CTNTTYPE, cntLen);
    sprintf( buf, "%d\r\n\r\n", s );
    client.write(buf, strlen(buf));
    client.write((char *)cam.getfb(), s);
    client.write(BOUNDARY, bdrLen);
  }
}

const char JHEADER[] = "HTTP/1.1 200 OK\r\n" \
                       "Content-disposition: inline; filename=capture.jpg\r\n" \
                       "Content-type: image/jpeg\r\n\r\n";
const int jhdLen = strlen(JHEADER);

void handle_jpg(void)
{
  WiFiClient client = server.client();
```

```
  cam.run();
  if (!client.connected()) return;

  client.write(JHEADER, jhdLen);
  client.write((char *)cam.getfb(), cam.getSize());
}

void handleNotFound()
{
  String message = "Server is running!\n\n";
  message += "URI: ";
  message += server.uri();
  message += "\nMethod: ";
  message += (server.method() == HTTP_GET) ? "GET" : "POST";
  message += "\nArguments: ";
  message += server.args();
  message += "\n";
  server.send(200, "text / plain", message);
}

void setup()
{

  Serial.begin(115200);
  //while (!Serial);            //wait for serial connection.

  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
```

```
  config.pixel_format = PIXFORMAT_JPEG;

  // Frame parameters
  //  config.frame_size = FRAMESIZE_UXGA;
  config.frame_size = FRAMESIZE_QVGA;
  config.jpeg_quality = 12;
  config.fb_count = 2;

#if defined(CAMERA_MODEL_ESP_EYE)
  pinMode(13, INPUT_PULLUP);
  pinMode(14, INPUT_PULLUP);
#endif

  cam.init(config);

  IPAddress ip;

  WiFi.mode(WIFI_STA);
  WiFi.begin(SSID1, PWD1);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(F("."));
  }
  ip = WiFi.localIP();
  Serial.println(F("WiFi connected"));
  Serial.println("");
  Serial.println(ip);
  Serial.print("Stream Link: http://");
  Serial.print(ip);
  Serial.println("/mjpeg/1");
  server.on("/mjpeg/1", HTTP_GET, handle_jpg_stream);
  server.on("/jpg", HTTP_GET, handle_jpg);
  server.onNotFound(handleNotFound);
  server.begin();
}

void loop()
{
  server.handleClient();
}
```