Faculty of Electronic and Computer Engineering and Technology (FTKEK)

DEVELOPMENT OF AN INTEGRATED SMART MIRROR SYSTEM THROUGH RASPBERRY PI FOR ENHANCED HOME AUTOMATION AND SECURITY.

UKASSYAH BIN UZAIR

Bachelor of Electronics Engineering Technology (Telecommunications) with Honours

2024

DEVELOPMENT OF AN INTEGRATED SMART MIRROR SYSTEM THROUGH RASPBERRY PI FOR ENHANCED HOME AUTOMATION AND SECURITY.

UKASSYAH BIN UZAIR

A project report submitted
in partial fulfilment of the requirements for the degree of
Bachelor of Electronics Engineering Technology (Telecommunications) with Honours

Faculty of Electronic and Computer Engineering and Technology (FTKEK)

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2024

UNIVERSITI TEKNIKAL MALAYSIA MELAKA
FAKULTI TEKNOLOGI DAN KEJUTERAAN ELEKTRONIK DAN KOMPUTER.

BORANG PENGESAHAN STATUS LAPORAN
PROJEK SARJANA MUDA II

Tajuk Projek : Development of an Integrated Smart Mirror System Through Raspberry Pi For Enhanced Home Automation and Security.

Sesi Pengajian : 2023/2024

Saya UKASSYAH BIN UZAIR mengaku membenarkan laporan Projek Sarjana Muda ini disimpan di Perpustakaan dengan syarat-syarat kegunaan seperti berikut:
1. Laporan adalah hakmilik Universiti Teknikal Malaysia Melaka.
2. Perpustakaan dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan laporan ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. Sila tandakan (✓):

✓	**SULIT***	(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)
✓	**TERHAD***	(Mengandungi maklumat terhad yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan)
☐	**TIDAK TERHAD**	

Disahkan oleh:

UKASSYAH

(TANDATANGAN PENULIS)

Alamat Tetap: 234 JALAN SPRINGHILL, 1/27 BANDAR SPRINGHILL, 71010 PORT DICKSON , NEGERI SEMBILAN.

[illegible]

(COP DAN TANDATANGAN PENYELIA)

MOHD FAIZAL BIN ZULKIFLI
Jurutera Pengajar
Jabatan Teknologi Kejuruteraan Elektronik & Komputer
Fakulti Teknologi Kejuruteraan Elektrik & Elektronik
Universiti Teknikal Malaysia Melaka

Tarikh: 12/1/2024

Tarikh: 12/1/2024

*CATATAN: Jika laporan ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali tempoh laporan ini perlu dikelaskan sebagai SULIT atau TERHAD.

DECLARATION

I declare that this project report entitled “Development of an intergrated smart mirror system through Rasberry Pi for enhanced Home Automation and Security.” is the result of my own research except as cited in the references. The project report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature	:	UKASSYAH
Student Name	:	UKASSYAH BIN UZAIR
Date	:	12/1/2024

APPROVAL

I hereby declare that I have checked this project report and in my opinion, this project report is adequate in terms of scope and quality for the award of the degree of Bachelor of Electronics Engineering Technology (Telecommunications) with Honours.

Signature	:	[illegible] MOHD FAIZAL BIN ZULKIFLI Jurutera Pengajar Jabatan Teknologi Kejuruteraan Elektronik & Komputer Fakulti Teknologi Kejuruteraan Elektrik & Elektronik Universiti Teknikal Malaysia Melaka
Supervisor Name	:	TS. MOHD FAIZAL BIN ZULKIFLI
Date	:	12/1/2024

DEDICATION

I would like to dedicate this work to my esteemed parents, Uzair Bin Abu Samah and Rubiah Binti Abu Jaappar, whose unwavering support and love have been a constant source of inspiration throughout my life. Your unselfish devotion and tireless encouragement have helped shape my academic and personal endeavors and have been instrumental in my pursuit of higher education and passion for research. I am deeply grateful for the many sacrifices you have made to provide me with every opportunity to succeed, and for the unwavering support you have shown me in every aspect of my life.

This dedication is a small token of my immense appreciation for your guidance, patience, and wisdom, which have helped shape me into the person I am today. Your unwavering love and support have been an invaluable gift, and I can only hope to make you both proud through my academic and personal achievements. Once again, I offer my heartfelt gratitude and deepest appreciation for all that you have done for me, and for the endless support you continue to provide.

ABSTRACT

In this digital transformation era, the evolution of the Internet of Things (IoT) has facilitated the seamless integration of devices in our daily lives, promoting smart living and enhancing security at home. This project explores the development of a smart mirror system using Raspberry Pi, combining home automation, security and voice assistance. The seemingly ordinary smart mirror becomes an interactive device offering diverse functionalities that extend beyond traditional reflections. The system integrates a Raspberry Pi single-board computer with a two-way mirror and an LCD screen to create an aesthetically pleasing yet functional device that provides useful information like time, weather updates, news, calendar reminders, etc. It employs the use of artificial intelligence for voice recognition and natural language processing, allowing users to control the system and other home automation devices using voice commands. Furthermore, the mirror system boasts a robust home security feature, incorporating facial recognition technology to identify household members and alert homeowners of any unauthorised access. This not only modernises home security but also provides a personalised user experience. The project underpins a paradigm shift in the home automation and security domain, offering a holistic, intelligent, and user-friendly solution. Through IoT, artificial intelligence, and Raspberry Pi, the smart mirror system presents a compelling blend of innovation, aesthetics, and functionality. It exemplifies the potential for technology to streamline daily routines, foster home safety and ultimately contribute to a more convenient and secure lifestyle.

ABSTRAK

Projek ini membincangkan pembangunan sebuah sistem cermin pintar, berasaskan Raspberry Pi, yang direka untuk mengautomasi dan mempertingkatkan keselamatan rumah dengan bantuan suara. Cermin pintar ini berfungsi bukan sahaja sebagai cermin biasa, tetapi juga sebagai pusat komunikasi dan maklumat, membolehkan pengguna melihat maklumat penting seperti cuaca, tarikh dan masa, serta notifikasi lain pada permukaan cermin sambil melakukan rutin harian mereka.Menggunakan teknologi Raspberry Pi sebagai otak operasi, sistem ini diintegrasikan dengan modul-modul IoT (Internet of Things) untuk memantau dan mengawal peranti yang berhubungan dalam rumah. Dengan ini, pengguna boleh mengendalikan sistem pencahayaan, penghawa dingin, dan sistem keselamatan rumah hanya dengan suara mereka.Selain itu, sistem ini juga dilengkapi dengan fungsi keselamatan yang canggih. Ia berfungsi sebagai sistem pengawasan video, merekod dan memantau aktiviti di sekitar rumah, dan memberi amaran kepada pengguna jika ada aktiviti yang mencurigakan. Kesimpulannya, cermin pintar berdasarkan Raspberry Pi ini merupakan solusi yang holistik dan inovatif untuk automasi rumah dan keselamatan, memberikan keselesaan dan ketenangan kepada penggunanya.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my supervisor, Ts. Mohd Faizal Bin Zulkifli for their precious guidance, words of wisdom and patient throughout this project.

I am also indebted to Universiti Teknikal Malaysia Melaka (UTeM) and for the financial support through claiming which enables me to accomplish the project. Not forgetting my fellow colleague, Muhammad Nur Ikhwan , Basyirah , Lee Won Jet , Muhammad Faiz, and Muhammad Hakim for the willingness of sharing his thoughts and ideas regarding the project.

My highest appreciation goes to my parents, parents in-law, and family members for their love and prayer during the period of my study. An honourable mention also goes to my father, Uzair Bin Abu Samah and my mother, Rubiah Binti Abu Jaappar for all the motivation and understanding. And to sister, Ammira, thanks for always supporting me through the end.

Finally, I would like to thank all fellow colleagues and classmates, fellow colleagues and classmates, the Faculty members, as well as other individuals who are not listed here for being co-operative and helpful.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

±	-	Plus minus
%	-	Percent
cm	-	Centimetre
mm	-	millimetre

LIST OF ABBREVIATIONS

V	-	Voltage
DC	-	Direct current
AC	-	Alternating current
LCD	-	Liquid crystal display
Wi-Fi	-	Wireless Fidelity
Hz	-	Hertz
I/O	-	Input/Output
USB	-	Universal serial bus
IoT	-	Internet of things
LED	-	Light-emitting diode
GSM	-	Global System for Mobile communication
SMS	-	Short Message Service
OTG	-	On The Go adapter

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Background

The In the dawn of an era increasingly reliant on technological advances, the implementation of smart, automated security systems in homes has become a topic of paramount importance. The urgency of this need is underscored by the growing rate of residential crimes in Malaysia, which have left families, such as the one reported by Norzamira Che Noh in the New Straits Times on February 18, 2023, in Taman Petaling, Klang, traumatised following their encounter with masked intruders. The present report explores the development of an intelligent Smart Mirror System based on Raspberry Pi, tailored towards improving home automation and security with the added convenience of voice assistance.

Such a system not only increases the security of a home but also integrates seamlessly with daily routines by providing organisational tools like calendars, maps, and the ability to control home switches. This dual functionality addresses the pressing need for enhanced security measures and contributes towards a more organised and efficient living environment. In this context, the project demonstrates its potential to be a game-changer for residential living, transforming houses into smart homes equipped with the tools necessary to deter crime and streamline daily life, offering a compelling argument for its broad implementation.

In conclusion, the development of a smart, automated system such as this is not merely an option. it's an essential step forward in combating rising crime rates and promoting a safer, more efficient living environment.

1.2 Problem Statement

In the current landscape, an escalating number of burglaries and robberies in residential areas are posing serious threats to home security. Traditional security measures are frequently found to be inadequate, thus, accentuating the need for a smarter, more proactive solution. Parallelly, the issue of time management is increasingly pressing. With our agendas growing denser, traditional organisational tools such as paper lists or physical calendars are proving to be time-consuming and susceptible to human error.

The challenge extends to the efficient organisation of our daily activities, which becomes vital amidst our fast-paced lives. Existing tools often lack the flexibility to accommodate unexpected changes swiftly and effectively. As future technology permeates deeper into our lives, there's a growing need to look beyond standard interaction methods like clicking on apps manually. A hands-free control mechanism, such as voice assistance, is being recognised as an urgent requirement in home automation systems.

Furthermore, the question of data storage and accessibility needs to be addressed. Home security systems today usually store CCTV data on local drives that are vulnerable to physical damage or theft. Hence, a safer, remote, and more easily accessible storage solution is a necessity. At present, home automation and security systems operate independently, which leads to a disjointed user experience. The call for a seamlessly integrated system combining these functionalities has never been more urgent.

The present systems also show rigidity and fail to provide personalised features that can adapt to each user's preferences. This rigidness hinders user satisfaction and usability. The proposed solution, a Smart Mirror System, aims to address all these concerns. This innovative system integrates home automation and security into a single platform. It offers user-friendly, voice-assisted controls and the unique feature of storing data securely on Telegram, ensuring a highly adaptable and personalised home environment.

1.3 Project Objective

These three objectives must be achieved by the end of this project, which are as follows:

1) To study and propose the implementation of IoT for Smart Mirror System For home automation and security with Voice Assistance Using Raspberry Pi. This system is intended to be an integral part of a home automation setup, providing users with access to a wide range of information such as the date, time, weather, and personal navigation, directly from their mirrors.
2) To develop and integrate home automation and security features into the smart mirror system. This will involve the development of software and hardware interfaces for various smart devices, such as lights and security cameras.
3) To analyse a voice assistance module for the smart mirror system. This module will enable users to interact with the smart mirror and control their home automation systems using voice commands, further improving user experience and accessibility.

1.4 Scope of Project

The scope of the project is defined as follows:

Hardware:

- Limited processing capabilities of the Raspberry Pi: The functionality of the smart mirror may be constrained by the processing power of the Raspberry Pi, affecting system responsiveness and speed.

Software (Voice Assistance):

- Speech recognition accuracy: The accuracy of the voice recognition system may vary based on factors like ambient noise, accents, and speech patterns, occasionally resulting in errors or misinterpretations.

Internet Connectivity:

- Reliance on stable Wi-Fi connection: The smart mirror system may require a stable and reliable Wi-Fi connection to function optimally, and any disruptions in the connection could affect its performance.
- Limited offline capabilities: Certain features or functionalities of the smart mirror system, such as weather updates or voice recognition, may necessitate an active internet connection and may not be available when offline.

1.5 Expected Results

The project "Development of a Smart Mirror System Based on Raspberry Pi for Home Automation and Security" anticipates achieving several outcomes that will substantially improve the interaction and integration of technology within homes.

1. User-friendly Interface: The development of the Smart Mirror System is expected to produce an intuitive and user-friendly interface. By integrating the Raspberry Pi, ESP module, LCD screen, and the Google Assistant, the project aims to create a seamless interaction between the user and the system.

2. Home Automation: The integration of the ESP module for home automation is expected to result in improved convenience. Users should be able to manage multiple home devices, such as lights, heating, or television, with simple voice commands or through the touch interface on the smart mirror.

3. Enhanced Security: The project also expects to achieve an increased level of security. Using the smart mirror system, users will receive alerts for any potential security risks, be it an unrecognised visitor at the door or any unusual activity within the house.

4. Efficient Communication: With the implementation of a speaker and microphone for Google Assistant integration, the system aims to facilitate efficient communication. This is expected to lead to quicker response times for tasks, providing real-time updates about home automation and security.

Overall, the project aims to provide a harmonious blend of comfort, convenience, security and aesthetic appeal by creating a versatile Smart Mirror System.

1.6 Project Organization

The development of the "Smart Mirror System Based on Raspberry Pi for Home Automation and Security" project will be organised into five main chapters, each with its own focus and objectives.

Chapter 1 will introduce the concept of the Smart Mirror System, including its use for home automation and security with Google Assistant integration. This chapter will also lay out the problem statement and the specific objectives of the project, along with the scope of the project framework. It will culminate with a discussion on the expected outcomes of the project.

Chapter 2 will delve into a thorough exploration of the relevant literature and previous work related to this project, including a conceptual framework and functionality discussion. A comparative analysis of the device components used in the project will be presented to guide the selection of components. This chapter aims to provide a comprehensive understanding of the underpinning technology and a context for the device component selection.

The methodology will be detailed in Chapter 3. This will include the precise procedures and designs that are employed to achieve the project's objectives. The chapter will provide a detailed overview of the step-by-step procedures involved in the project development, illustrated through a flowchart or block diagram for clarity.

Chapter 4 will focus on the presentation and analysis of the project results. It will detail the data collected and its subsequent analysis, presented through tables or figures. This chapter will also provide a comparison between the expected outcomes and the project's objectives. The primary purpose of this chapter is to objectively present the project results.

Finally, Chapter 5 will summarise the research report, revisiting the points made in the introduction and the methods employed. It will also reiterate the significant findings of the project. Limitations encountered during the project will be identified in this chapter, along with suggestions for potential improvements in future projects. This concluding chapter will encapsulate the project's journey, from its conception to its realisation and potential for further development.

1.7 Summary

Chapter 1 introduces the "Development of a Smart Mirror System Based on Raspberry Pi for Home Automation and Security" project, an innovative effort to enhance residential security and simplify home automation. The project utilises Raspberry Pi and an ESP module, alongside an LCD screen, speaker, and microphone to enable Google Assistant integration.

The chapter lays out the problem statement, highlighting the gap in current smart home systems which often necessitate multiple platforms or applications to manage various devices, thus increasing complexity for users. Existing systems also often lack robust security measures, and they tend to have non-intuitive user interfaces that reduce system usage over time. To address these issues, this project proposes the development of an integrated, user-friendly Smart Mirror System that combines both home automation and security features.

Chapter 1 also outlines the project's objectives, with the main goal being the creation of a smart mirror system using Raspberry Pi. The objectives also include setting up a user-friendly interface on the LCD screen, enabling communication between the Raspberry Pi and ESP module for control of home devices, implementing a camera module for security, and developing a voice control system using a microphone and speaker.

In terms of scope, the project covers the setup and configuration of the Raspberry Pi, ESP module, LCD screen, speaker and microphone, and the integration of a robust security system. Lastly, the expected outcomes are discussed, which include a user-friendly interface, improved home automation, enhanced security, efficient communication, and aesthetic appeal.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter, the focus is on the analysis of completed projects related to smart safety device systems. The aim is to assess the reasons for developing a personal safety and protection system, as well as the ideas, concepts, and techniques employed in their creation.

The study spans over five years from the project implementation date and serves as a valuable resource for the proposed project. Additionally, the review evaluates the appropriateness of the search methods utilized in the completed projects. This evaluation will assist in the assembly session by ensuring the accuracy of the software and hardware used in the proposed project.

It is also important to carefully evaluate the advantages and disadvantages of software and hardware systems to make informed decisions regarding their use in the project. By conducting a thorough literature review, the researcher can gain a better understanding of the subject matter and identify gaps in existing research, which can lead to innovation and new ideas. Ultimately, the literature review serves as a crucial component of the research project, enabling the researcher to build upon the knowledge and practices of others to create a successful and effective personal safety and protection system.

2.2 Existing Project

2.2.1 Visual Machine Intelligence for Home Automation

This paper was proposed Suraj, Ish Kool, Dharmendra Kumar& Shovan Barma (2018)[1] in order to design and implement a combination of home automation using visual intelligence and machine learning techniques[1]. The proposed system aims to eliminate the need for discrete sensors by leveraging the power of vision-based sensing and machine learning models to monitor and control appliances in a home environment. The system consists of an image acquisition device comprising cameras mounted on servo motors, a Raspberry Pi, and an Intel Galileo Gen2 board[1].

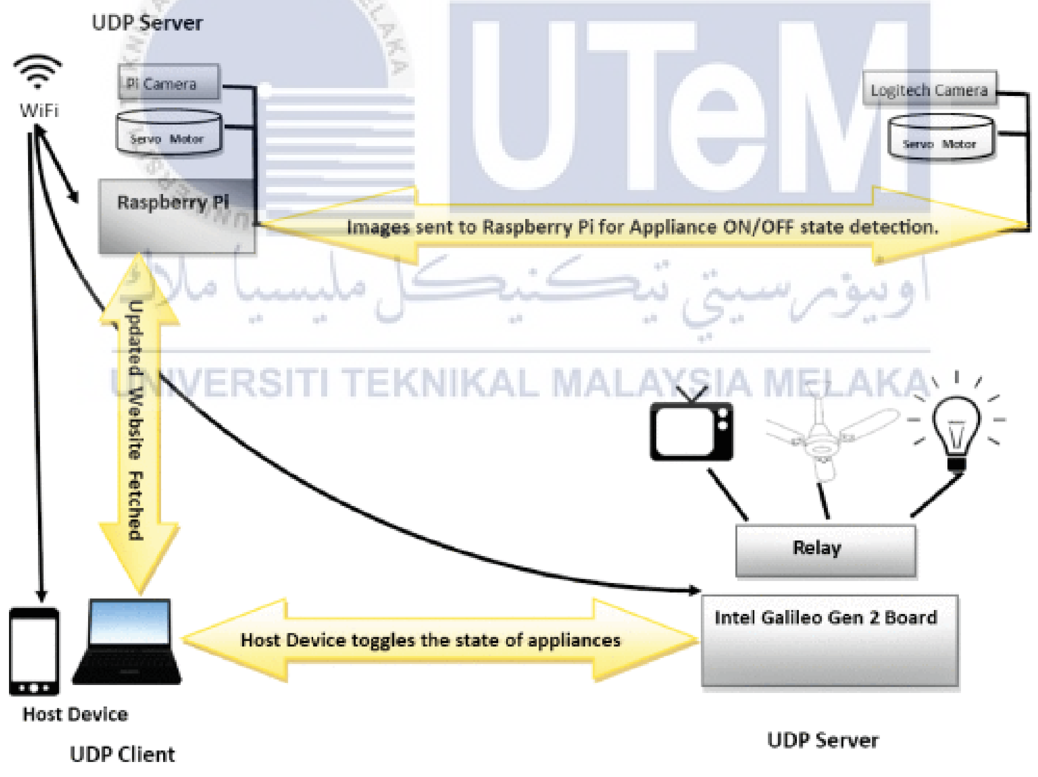

UDP Server
WiFi
Pi Camera
Servo Motor
Raspberry Pi
Logitech Camera
Servo Motor
Images sent to Raspberry Pi for Appliance ON/OFF state detection.
Updated Website Fetched
Relay
Intel Galileo Gen 2 Board
Host Device toggles the state of appliances
Host Device
UDP Client
UDP Server

Figure 2.1 System block diagram[1]

The training phase involves collecting annotated images of appliances in different ON/OFF states and training machine learning models using the dib-C++ library. Machine learning models then process the acquired images to detect the states of the appliances. The detected states are updated on an appliance control website, which allows users to remotely monitor and toggle the states of the appliances.

The results of the implemented system show promising performance in detecting the ON/OFF states of appliances such as televisions and fans. However, there are limitations, such as the system's dependence on specific colours and textures of appliances and false negatives when sunlight or reflections are detected as the ON state of a tube light[1].

The future work proposed includes collecting a more diverse database of appliance images to improve generalization, overcoming the false negative issue caused by sunlight, and integrating notifications of energy wastage through social networking platforms. The system could also be extended to support cellular networks for remote monitoring and automation. Additionally, adding security measures such as face recognition could enhance access control and protect sensitive data captured by the system.

2.2.2 Interactive IoT-based Speech-Controlled Home Automation System

According to this article, the authors Nombulelo CC Noruwana, Pius Adewale Owolawi & Temitope Mapayi, (2020)[2] defined the development of an interactive Internet of Things (IoT) based Speech-Controlled Home Automation system using Google Assistant. The system aims to provide users with remote control over their home appliances through voice commands using Google Assistant[2]. The system utilizes components such as Raspberry Pi, Google Assistant, and a Logitech Webcam C170[2].

The authors conducted a literature survey to explore existing home automation systems and highlighted the advantages of wireless-based systems like Bluetooth, Wi-Fi, and IoT. They also reviewed related works on home surveillance, intelligent access control, and home navigation systems.

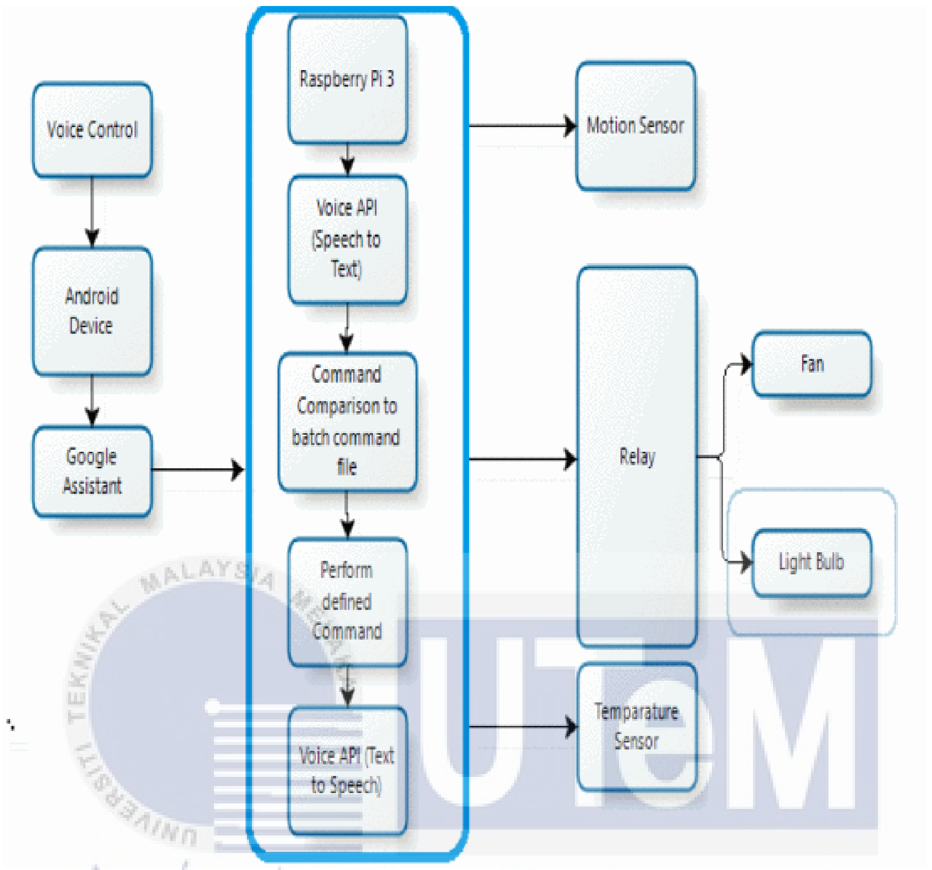


Figure 2.2 Detailed system design of Interactive IoT-based Speech-Controlled Home Automation System[2]

The system architecture was presented, showcasing the flow of voice commands from the user to the Raspberry Pi, which communicates with Google Assistant and controls various components and appliances. The authors described the detailed system design, including the specifications of Raspberry Pi, the capabilities of Google Assistant, and the features of the Logitech Webcam C170.

The developed software applications were tested for functionality, including user registration, login, video streaming, and communication through WhatsApp and email. The system successfully controlled appliances like lights and fans, activated and deactivated motion sensors, and displayed temperature readings on an LCD.

The experimental results showed the system's performance in different scenarios, considering factors like noise levels, room furnishing, device proximity, temperature, humidity, and wind. The system achieved varying levels of accuracy, with higher accuracy in quiet environments, furnished rooms, closer device proximity, and favourable temperature and humidity conditions.

In conclusion, the proposed IoT-based Speech-Controlled Home Automation system using Google Assistant provides convenience, control, and energy savings for users. The experimental results demonstrate the system's effectiveness and future research may explore incorporating voice commands in native languages and expanding the system's capabilities.

2.2.3 iHAS: An Intelligent Home Automation Based System for Smart City

This paper was presented by Shaik Mulla Shabber, Mohan Bansal, P Mrudula Devi & Prateek Jain (2021)[3] to propose the importance of smart homes and the need for home automation systems. The authors highlight the advantages of smart homes, including energy efficiency, improved comfort, increased security, and convenience. They emphasize that smart home technology is becoming more common in India due to its affordability and accessibility[3].

The article presents a literature review of prior related work in the field of home automation[3]. Different technologies such as GSM-based systems, RF-based systems, and Wi-Fi-based IoT frameworks are discussed. The authors argue that Wi-Fi-based IoT frameworks offer power optimization and are a preferable solution for future home automation systems.

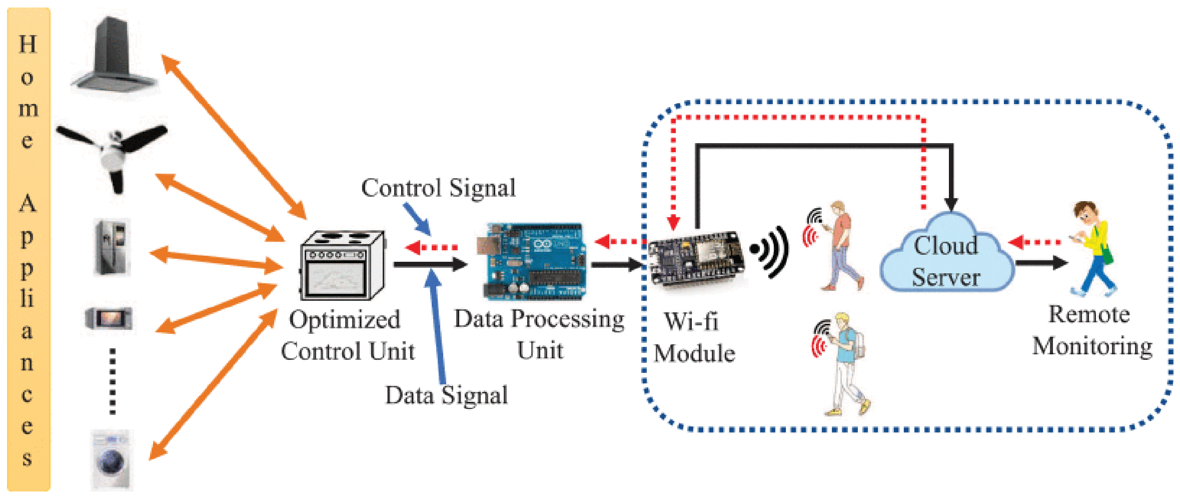


Figure 2.3 The design flow of proposed iHAS system using IoT framework for smart home care[3]

The proposed system, called the Intelligent Home Automation System (iHAS), is based on an IoT architecture. It consists of hardware components such as a Wi-Fi enabled microcontroller, a relay board, and various appliances connected to the relay outputs. The software aspect of the system utilizes the Blynk app, which allows users to control and monitor the appliances remotely using a smartphone.

The authors describe the experimental setup, including the microcontroller programming using the Arduino IDE and the configuration of the Blynk app. They provide illustrations and diagrams to explain the system design and user interface.

The results of the experiments show the successful control and monitoring of the appliances using the iHAS system. The LEDs on the relay board indicate the state of the appliances, and the authors demonstrate the functionality of the system with and without external loads[3].

In conclusion, the proposed iHAS system offers a solution for intelligent home management through remote monitoring and control. The system provides convenience, efficiency, and affordability for users, making their lives simpler and more comfortable. The experiments validate the system's effectiveness, and the authors suggest that further research can explore additional features and improvements for smart home automation.

2.2.4 Integration of a Video Surveillance System into a Smart Home Using the Home Assistant Platform

This paper was introduced by Batyrzhan K. Akhmetzhanov, Omar Aslan Gazizuly, Zhanserik Nurlan & Nurkhat Zhakiyev (2022)[4] to design and implement concept of the Internet of Things (IoT) and its growing importance in various fields, including smart buildings, telemedicine, and smart homes[4]. The IoT is seen as a way to interconnect and control devices, offering numerous business opportunities. The authors highlight the increasing number of connected devices and the potential they hold for improving people's lives.

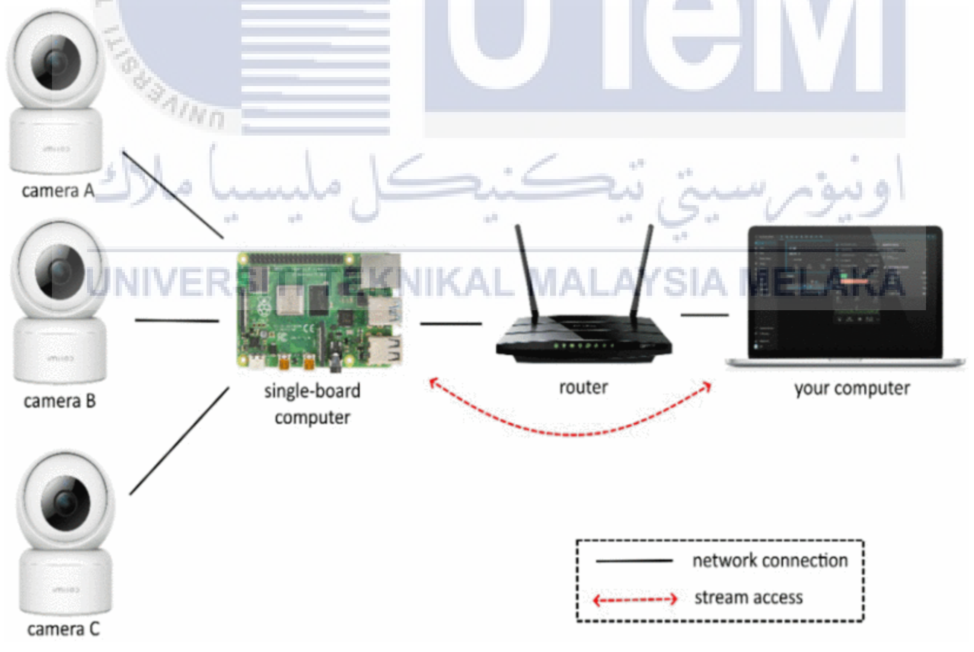


Figure 2.4 System architecture using multiple surveillance cameras[4]

The paper focuses on two key aspects of home automation: access and security control systems, and the integration of video surveillance systems using the MotionEye web interface[4]. The authors propose a modified solution called Home Assistant for controlling devices in smart homes or buildings. Home Assistant is an open-source project that provides a standard framework for managing and communicating with different devices[4]. It is compatible with various platforms and can be easily controlled via a smartphone app.

The integration of video surveillance systems is achieved using the MotionEye web interface, which allows users to control their cameras remotely. The authors explain the setup process and highlight the benefits of using this interface, such as easy access to live video feeds, remote monitoring, and notification features.

The proposed solutions are implemented using Raspberry Pi single-board computers and ESP8266 microcontrollers, making them cost-effective and accessible. The authors emphasize the simplicity and affordability of their approach, which makes it suitable for personal homes and small businesses[4].

In conclusion, the paper presents a cost-effective and user-friendly approach to home automation and video surveillance using open-source technologies. The proposed solutions offer convenience, security, and the potential for further development and expansion. Overall, the paper provides valuable insights into the practical implementation of IoT-based home automation and security systems.

2.2.5 Smart Energy Efficient Home Automation System Using IoT

This project had been conducted by Satyendra K. Vishwakarma, Prashant Upadhyaya ,Babita Kumari& Arun Kumar Mishra (2019)[5] in which introduces the concept of smart home automation using Internet of Things (IoT) technology[5]. The authors highlight the increasing demand for electricity and the need for energy-efficient solutions. They propose a smart home automation system that aims to reduce power consumption while providing safety and security for home equipment.

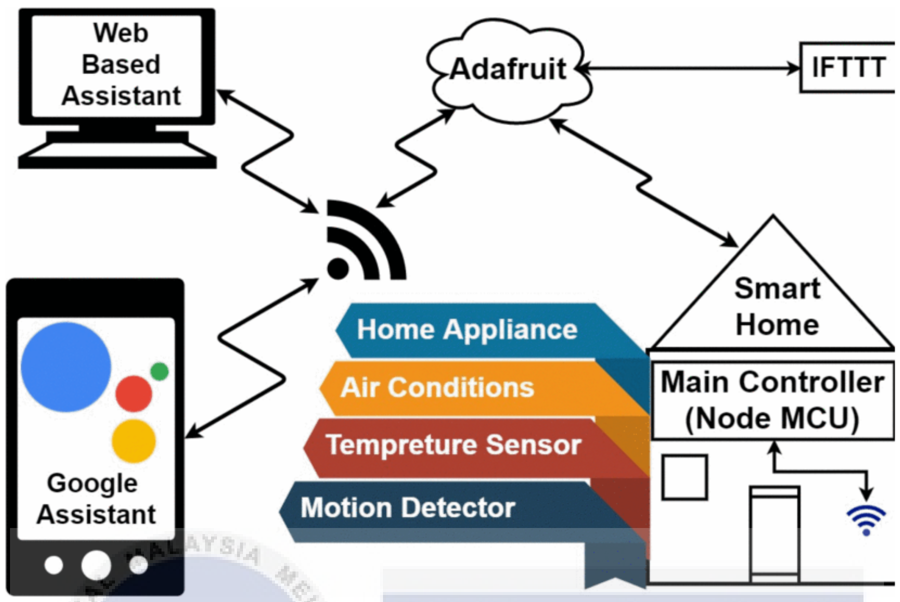


Figure 2.5 System architecture Smart Energy Efficient Home Automation[5]

The system utilizes IoT connectivity and integrates voice control through Google Assistant and web-based services. The main controller unit, connected to a 24-hour Wi-Fi network, acts as the central hub for controlling various home appliances[5]. Sub-units are connected to the main controller, transforming non-smart devices into smart appliances. Users can access and control their smart homes remotely using Google Assistant and a web-based application[5].

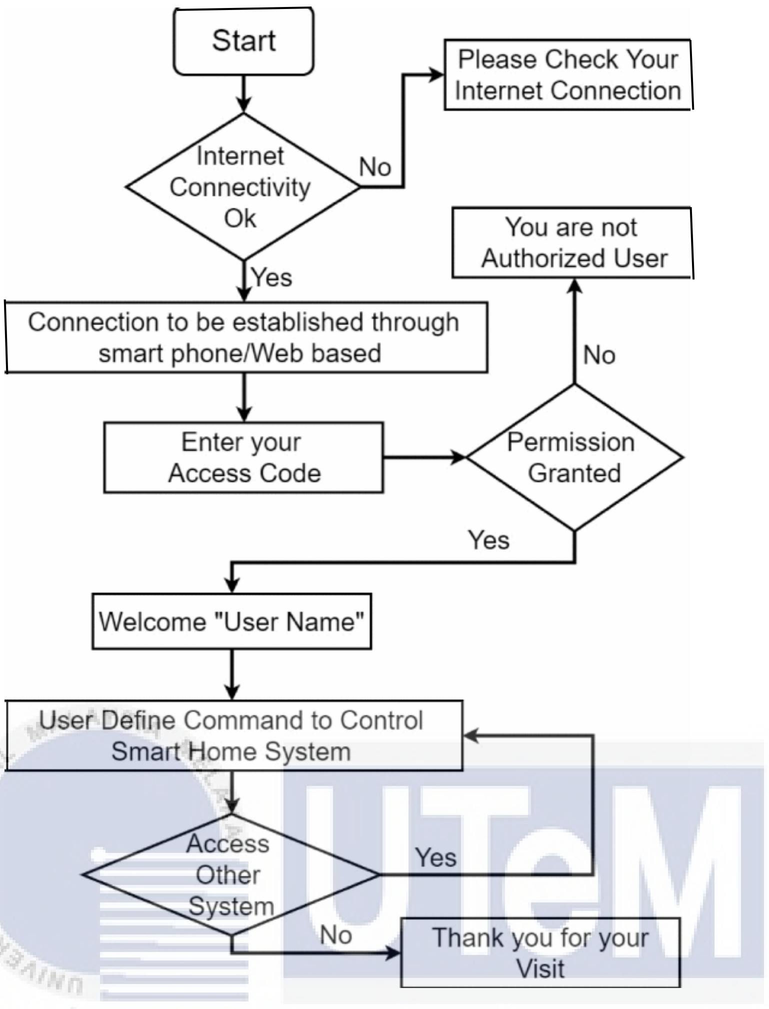

Start
Please Check Your Internet Connection
Internet Connectivity Ok
No
Yes
You are not Authorized User
Connection to be established through smart phone/Web based
No
Enter your Access Code
Permission Granted
Yes
Welcome "User Name"
User Define Command to Control Smart Home System
Access Other System
Yes
No
Thank you for your Visit

Figure 2.6 System flow on the smart home automation system using google assistant[5]

The system requirements include using NodeMcu (ESP8266) for IoT connectivity, IFTTT for web-based services, and Adafruit for MQTT communication. The authors explain the working models of the system, demonstrating the flow of commands from Google Assistant to the main controller unit and the activation of relays to control the home appliances[5].

The proposed system offers several advantages: energy efficiency, remote access and control, and enhanced security through user-defined commands. It particularly benefits elderly and disabled individuals with difficulty operating traditional home appliances.

The authors provide detailed system design and implementation, including the internal architecture of the controller unit and the development of an IoT home automation dashboard. They also present a prototype model of the control unit and connected home appliances[5].

In conclusion, the proposed smart home automation system offers a practical and energy-efficient solution for controlling and monitoring home appliances. Integrating IoT technology, voice control, and web-based services enhances the user experience and reduces power consumption. The authors suggest future work to expand and deploy the system's capabilities in real-life scenarios.

2.2.6 IoT Smart Home Assistant for Physically Challenged and Elderly People

Development of an IoT-based smart home system aimed at assisting elderly and physically challenged individuals proposed by S.K Sooraj, E Sundaravel, Babu Shreesh & K. Sireesha (2020)[6]. The system incorporates features such as voice control of appliances, remote monitoring of the house, and live streaming of surveillance footage[6]. The proposed methodology includes two main use cases: a security and surveillance system and a conventional system for controlling appliances remotely.

For the security and surveillance system, the authors utilize a Raspberry Pi as the main component, along with a PIR sensor for event detection and a Pi Camera for capturing images. When motion is detected, the Raspberry Pi sends commands to the Pi Camera to take pictures and sends them via email to a registered address. The system also enables live streaming of surveillance footage to a web page or app, allowing real-time house monitoring[6].

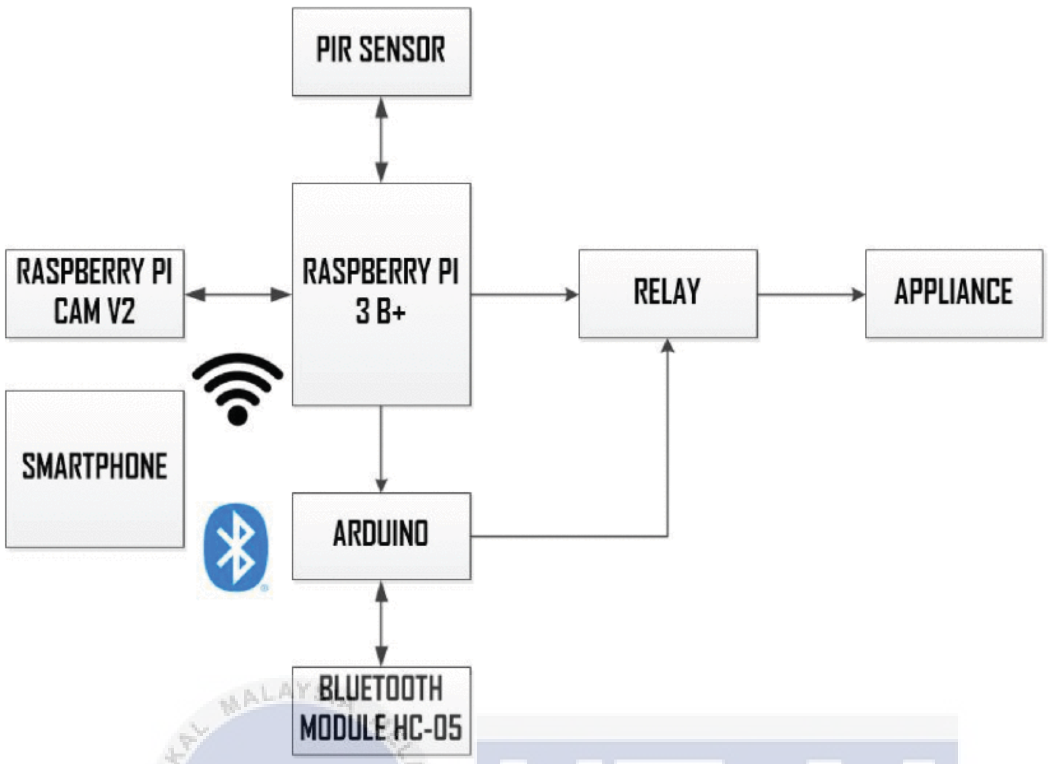


Figure 2.7 Block diagram of a proposed system[6].

In the conventional system, a graphical user interface (GUI) is created for users to control appliances remotely. The GUI is designed to be user-friendly and accessible for elderly users. Users can control appliances through a web interface or via voice commands using a mobile app. Bluetooth communication connects the mobile app with an Arduino board, which interfaces with the relay to control the appliances.

The hardware requirements for the system include a Raspberry Pi, Pi Camera, PIR sensor, relay, Arduino board, and Bluetooth module. The software requirements involve PHP, Nginx, CSS, HTML, and JavaScript for coding and creating the GUI[6].

The hardware implementation of the system is demonstrated, showcasing the connection and functionality of the various components. The authors provide flowcharts and diagrams to illustrate the system's operation, including event sensing, image capturing, email notifications, voice control, and appliance control.

In conclusion, the IoT-based smart home system presented in this paper offers practical solutions for improving the safety, convenience, and independence of elderly and physically challenged individuals[6]. Integrating IoT technology, surveillance cameras, voice control, and remote appliance control enhances the system's overall functionality. The proposed system demonstrates promising results and can be scaled and customized based on user requirements.

2.2.7 Enhanced Home Automation and Security Using IoT Architecture

In this study, Sudiir Mohan, Nithish Arro J, R Sitharthan (2022)[7] had proposed an "Smart Energy Efficient Home Automation System Using IoT" explores a smart home automation system that leverages Internet of Things (IoT) technology[7]. The paper emphasizes the importance of energy efficiency in homes and proposes an innovative solution to automate and control various aspects of a household using IoT devices.

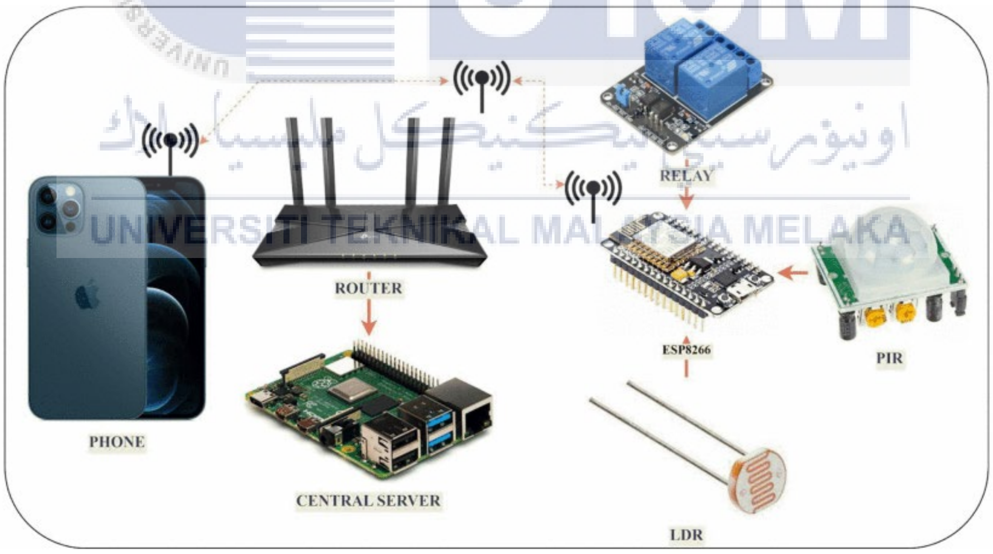


Figure 2.8 System block diagram using Internet of Things[7]

The authors compare their proposed system with existing approaches and highlight the advantages of using a Raspberry Pi as the main controller instead of an Arduino or ESP8266. They discuss the system architecture, which includes the device layer consisting of sensors such as passive infrared (PIR) sensors and light-dependent resistors (LDRs), the network layer utilizing Wi-Fi communication, and the application layer managed by the Raspberry Pi running the Home Assistant OS.

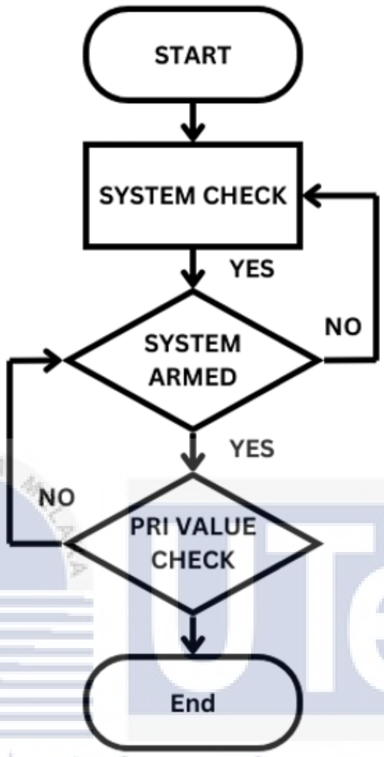


Figure 2.9 Flow chart of the system[7]

The functionality of the system is demonstrated through various working scenarios. The PIR sensors are utilized for security purposes, detecting motion and sending push notifications to the user's mobile device. The LDRs control lighting based on ambient light levels, automatically turning on/off the lights as needed. The relay modules enable the control of appliances, which can be operated remotely through a web-based interface.

The experiments' results show the system's successful operation, with fast response times and reliable performance. The user interface provided by the Home Assistant mobile app allows for easy monitoring and control of the home automation system.

The paper concludes by highlighting the scalability and cost-effectiveness of the proposed system, making it accessible for implementation in real-world scenarios. The authors suggest future enhancements, such as incorporating image processing for enhanced security and exploring advanced authentication methods.

Overall, the journal presents a comprehensive and practical approach to smart home automation using IoT, offering energy efficiency, convenience, and security benefits for households.

2.2.8 Implementation of Home Automation using Smart Mirror

This paper was proposed by P Karthik, U G Keerthiv Sanjay, S Abhiram, V Dinesh Kumar (2021)[8] to develop an innovative smart mirror concept with integrated home automation. Unlike conventional mirrors, this intelligent system uses a Raspberry Pi and a Blynk app, which work in tandem to monitor and control various home appliances. The Raspberry Pi, acting as the brain of the system, is linked to an LCD monitor, which is positioned behind a two-way mirror. This allows the display of relevant information, such as time, date, weather reports, news feeds, and room conditions, to the user[8].

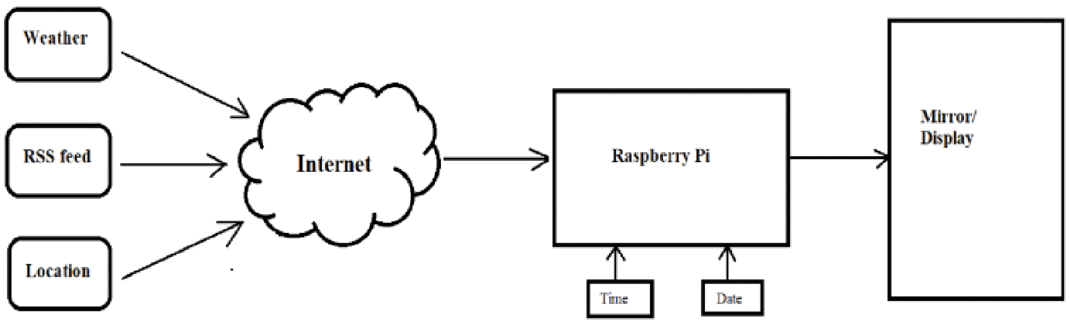


Figure 2.10 Flow chart of Home Automation using Smart Mirror[8]

The smart mirror has been developed with a user-friendly GUI, built using Python's Tkinter tool, enabling interaction by touch or voice assistance. An additional feature of the system is the inclusion of a mobile app for remote control of home appliances, which communicates with the smart mirror to ensure synchronised status updates.

The smart mirror is not only an information provider but also a control centre for home appliances via a 4-channel relay connected to the Raspberry Pi. The relay can be controlled through the display system or the Blynk app, enabling remote management of appliances while ensuring changes in appliance status are reflected both on the mirror and the app[8].

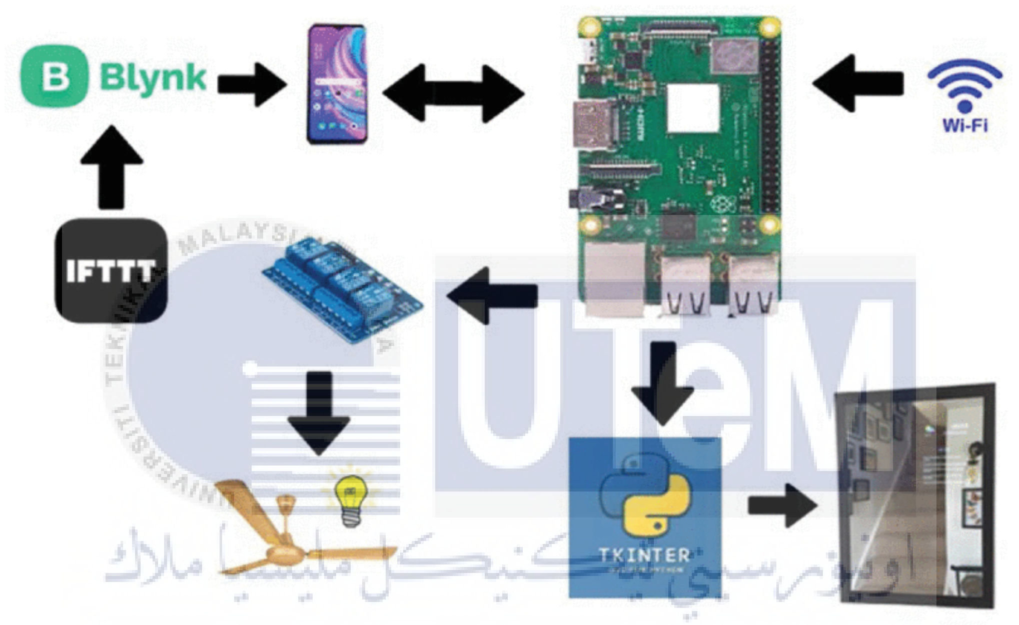


Figure 2.11 Block diagram of system connection[8]

The researchers envisage this technology having a broad appeal to consumers seeking a more comfortable and convenient lifestyle. Future improvements could include personalisation features, appliance power consumption analysis, and multimedia-based systems. This smart mirror represents a significant advancement in IoT home automation, combining the simplicity of everyday routines with the power of modern technology.

2.2.9 IoT Based Smart Security and Home Automation

This paper was proposed by Shradha Somani, Parikshit Solunke, Shaunak Oke, Parth Medhi & P.P. Laturkar (2018)[9] in order to implementing a Home Automation system leveraging the Internet of Things (IoT) capabilities. The paper discusses a system that not only automates home appliances but also provides security through motion sensors and a camera module[9]. It emphasizes the role of an Android application, transforming a smartphone into a remote control for home appliances and enabling real-time notifications of potential intrusions or hazardous situations like smoke detection[9].

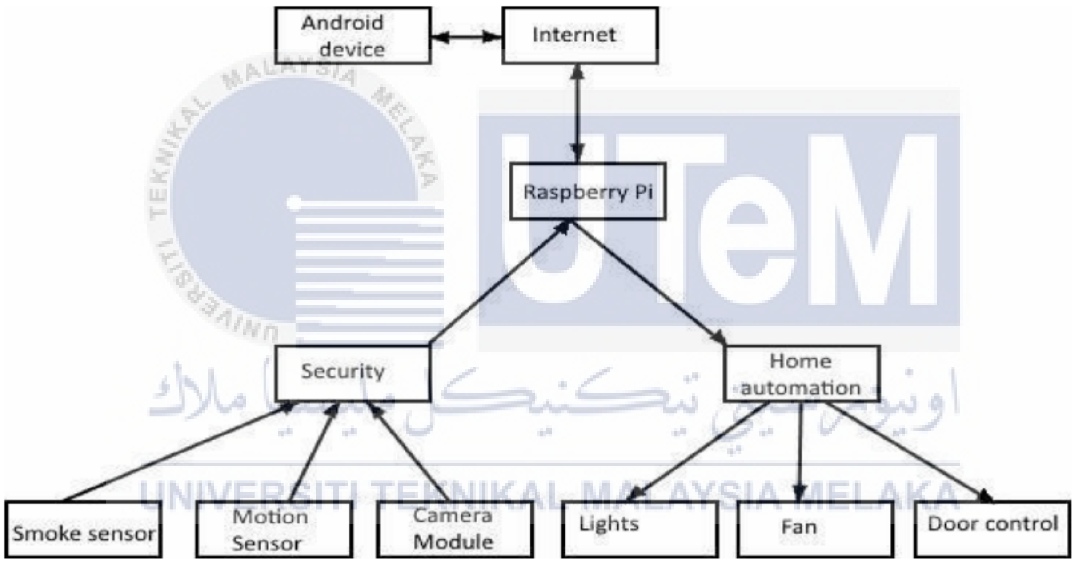


Figure 2.12 System Architecture Smart Security and Home Automation[9]

A Raspberry Pi, a miniaturized computer that acts as a server is central to the system. It connects to various sensors and appliances, and thanks to its GPIO pins and USB ports, it can control and interact with these devices. The paper further elaborates on the role of different sensors, like PIR motion detection, gas leakage detection, and temperature and humidity sensors, in enhancing the system's functionality[9].

Significant attention is paid to the system's security aspects, with the integration of AES encryption for data protection. The system alerts homeowners via a dedicated Android application and email, allowing them to respond to potential security threats. Additionally, the system ensures communication, even in the absence of internet access, via text messages.

The paper concludes by acknowledging the potential for future improvements, such as voice call integration, advanced biometric login, and image processing for increased security accuracy. The objective of this project is to provide a convenient, secure, and comprehensive home automation solution that not only improves quality of life but also enhances home safety[9].

2.2.10 "Smart Home Automation Device" Using Raspberry Pi and Arduino Uno

According to this article, the authors (Harsha Vardhan Tomar, Anagha Anand, H L Harsha, Anubhav Deshwal, & Badari Nath, 2022)[10] implementing a smart home automation system using advanced technologies such as the Internet of Things (IoT) and Artificial Intelligence (AI)[10]. The system aims to provide homeowners convenience, security, and cost savings by integrating various functionalities into a single device.

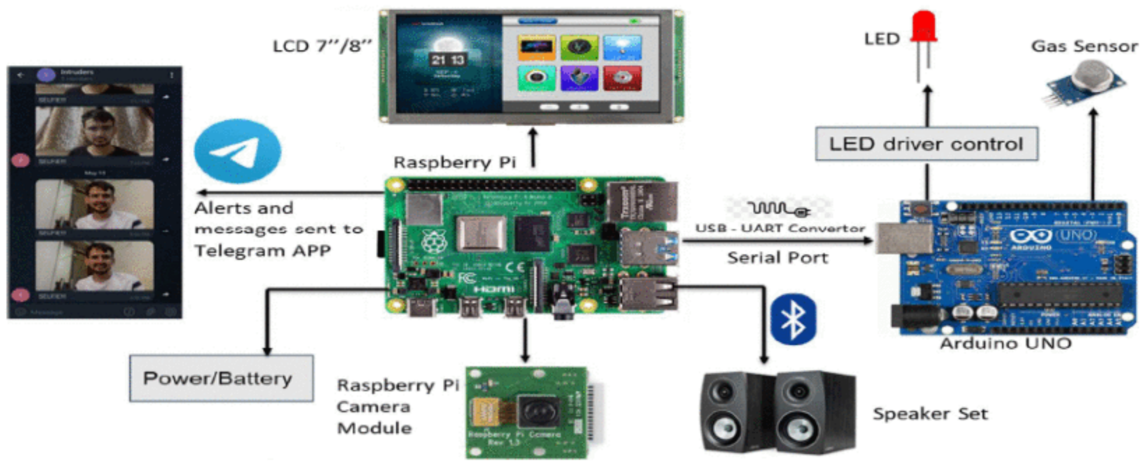
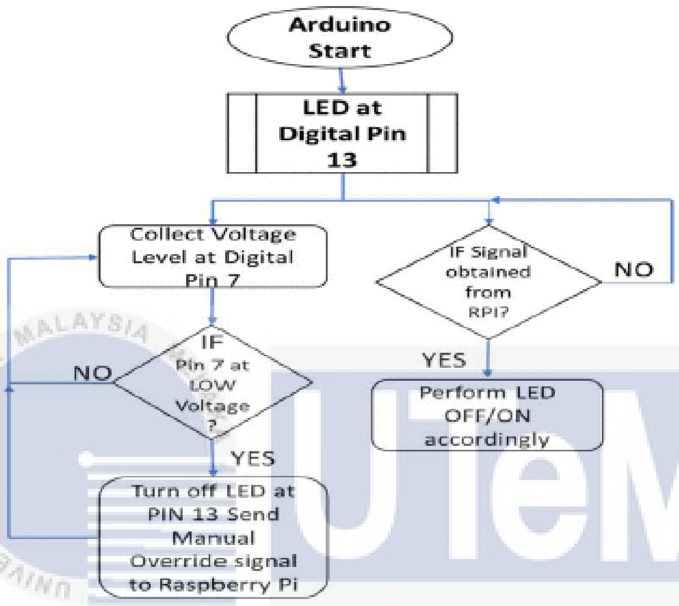


Figure 2.13 Block diagram of Smart Home Automation[10]

The model utilizes a low-cost microcomputer Raspberry Pi as the central controlling unit. It incorporates features such as gas leakage detection, intrusion detection, lights control, music player, real-time weather reports, and image viewer. The system is designed to enhance the user experience while being cost-effective and reliable[10].



Arduino Start
LED at Digital Pin 13
Collect Voltage Level at Digital Pin 7
IF Signal obtained from RPI?
NO
IF Pin 7 at LOW Voltage ?
NO
YES
Perform LED OFF/ON accordingly
YES
Turn off LED at PIN 13 Send Manual Override signal to Raspberry Pi

Figure 2.14 Flow chart of the system Arduino operation[10]

The article presents a literature survey that discusses related research in the field, including intrusion detection systems, face detection and tracking, smart door systems, and automated billing systems. These studies provide insights into the technologies and techniques for developing the proposed smart home automation system.

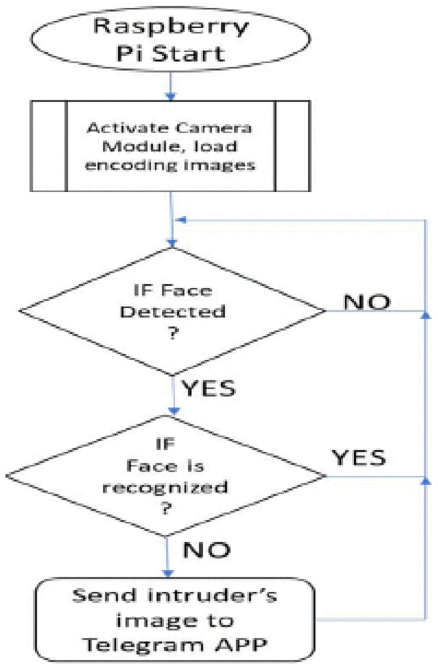

Raspberry Pi Start
Activate Camera Module, load encoding images
IF Face Detected ?
NO
YES
IF Face is recognized ?
YES
NO
Send intruder's image to Telegram APP

Figure 2.15 Flow chart of the system Raspberry Pi operation[10]

The methodology section explains the hardware and software tools used to implement the system, including Raspberry Pi, Arduino Uno, gas sensors, touch-sensitive LCD display, OpenCV, and Pygame mixer. Circuit diagrams and flowcharts are provided to illustrate the connections and functioning of different components[10].

The results and analysis section highlights the capabilities of the final model, including the touchscreen interface for controlling the device, intruder detection and access control through face recognition, gas leakage detection with alerts, smart lighting control, music player functionality, and real-time weather reports. The system also sends notifications and images of intruders to a Telegram group, ensuring real-time communication and enhancing security.

In conclusion, the proposed smart home automation system offers a comprehensive solution for homeowners, providing convenience, security, and energy efficiency[10]. Integrating multiple functionalities into a single device improves user experience and reduces costs. The article suggests future enhancements, such as implementing multithreading to enable simultaneous operations of different functions. Overall, the system demonstrates the potential of IoT and AI technologies in creating smart and sustainable homes.

2.2.11 Internet of Things (IoT) enabled Sustainable Home Automation along with Security using Solar Energy

This paper was presented by Swetha Amit, Anitha Sarah Koshy, S Samprita, Shwetali Joshi, & N Ranjitha (2019)[11] to propose a smart power management system for home automation and security, utilizing solar energy as the primary power source. The system is designed to be controlled through voice commands using Alexa or a mobile device. It incorporates various sensors such as door, PIR, gas, fire, and glass break sensors to ensure home security[11]. The power supply is managed through a solar charge controller, which regulates the voltage from the solar panel and switches between solar power and the main power source as needed[11].

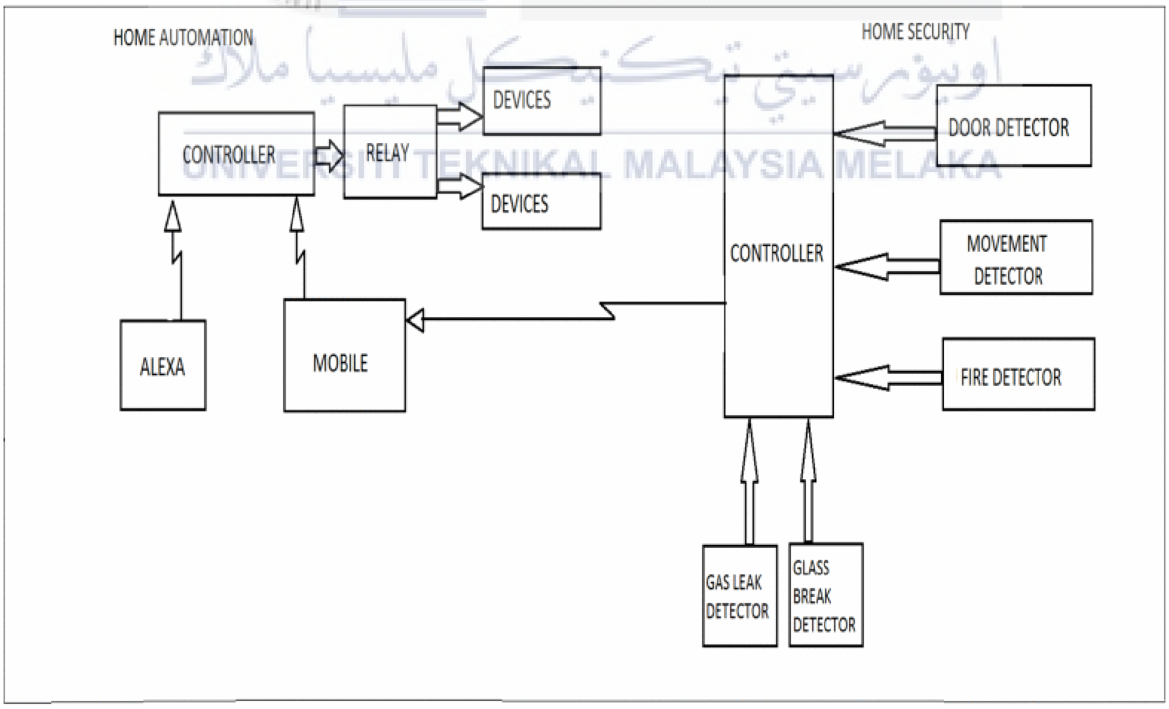


Figure 2.16 Block diagram of home Automation with Security using Solar Energy[11]

The block diagram of the entire system illustrates the flow of commands and data between the controller, sensors, and mobile devices. The voltage and current sensors, along with the Arduino, provide real-time information about power consumption and battery levels[11]. The Blynk app is used for IoT communication, enabling remote control and visualization of sensor data[11]. Additionally, the Alexa app plays a significant role in providing voice commands for device control, both at home and remotely.

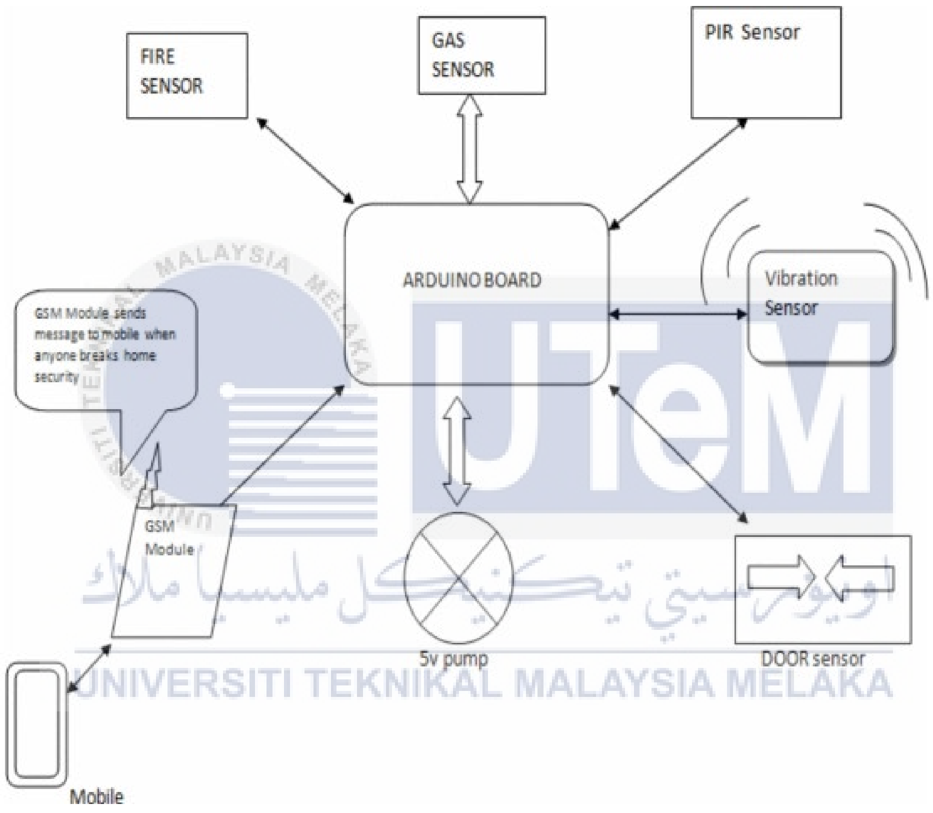


Figure 2.17 Block diagram connection of Arduino Board[11]

The paper reports successful implementation of the system, demonstrating the ability to control devices through Alexa and the Alexa app and achieving home automation. The solar power harvesting setup allows for backup power of up to 6 hours. The home security aspect is addressed through the deployment of various sensors, and alerts are sent to the user's mobile device via the Blynk app[11].

In conclusion, the paper presents a comprehensive solution for home automation and security, utilizing solar power and IoT technology. The successful implementation of the system highlights the potential of future homes to provide a wide range of automated services. The authors suggest that automation will continue to be a trending technology, with applications in remote healthcare services and other domains.

2.2.12 Home Automation using Smart Devices and IoT

This paper was introduced by Shruti Dash & Pallavi Choudekar (2021)[12] to design and implementation of an IoT-based home automation system using Raspberry Pi as the central control unit[12]. The system consists of three modules: a smart energy meter, a smart surveillance system, and IoT-based appliance control[12]. The paper begins with an introduction to the growing importance of home automation and the advantages of using Raspberry Pi and cloud services for this purpose[12].

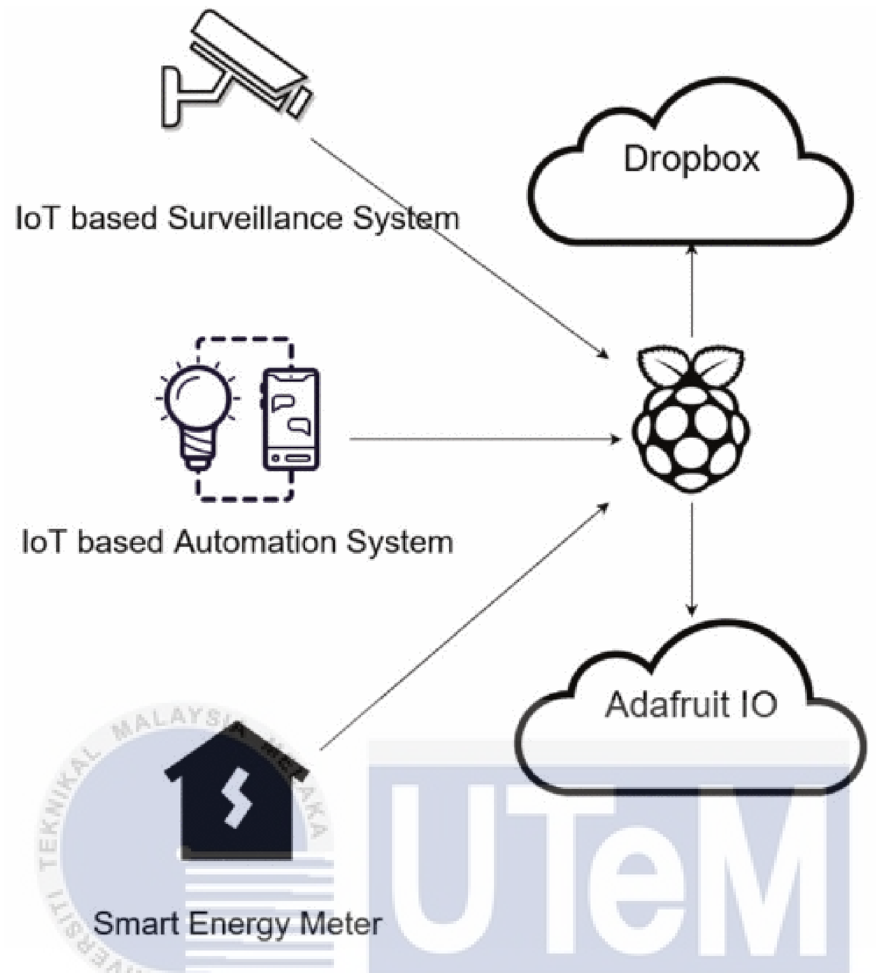


Figure 2.18 Schematic diagram of proposed system[12]

The literature review compares the proposed model with existing home automation systems, highlighting its unique features and advantages. The proposed methodology follows an Object-Oriented Design and Analysis Methodology, breaking down the system into three modules and detailing the functions of each component[12]. Flowcharts are provided to illustrate the working of the smart energy meter, IoT-based appliance control, and smart surveillance system.

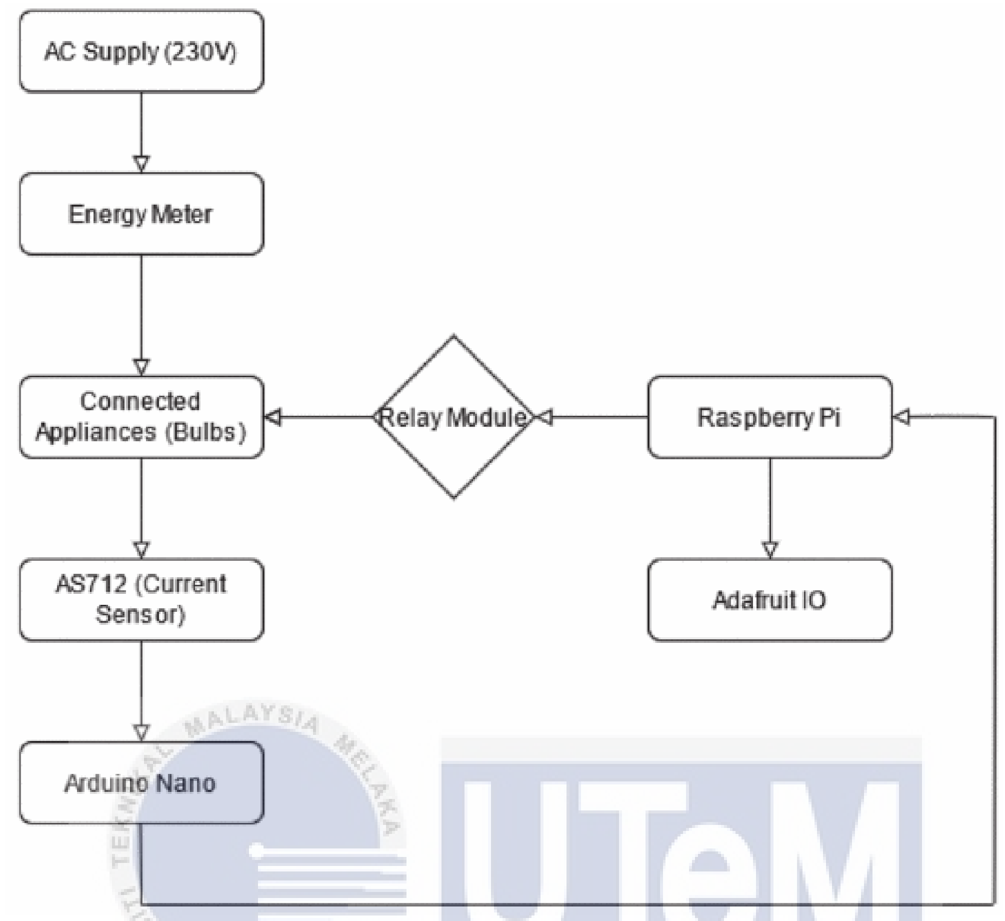


Figure 2.19 Flowchart of appliance control[12]

The results and discussion section presents the hardware setup of the system and showcases various dashboards and data representations[12]. The appliance control dashboard allows remote control of appliances, while the live current consumption dashboard provides real-time data. Historical data on current and energy consumption is displayed in graph form. The smart surveillance system includes motion detection and facial recognition capabilities, with images being stored on Dropbox.

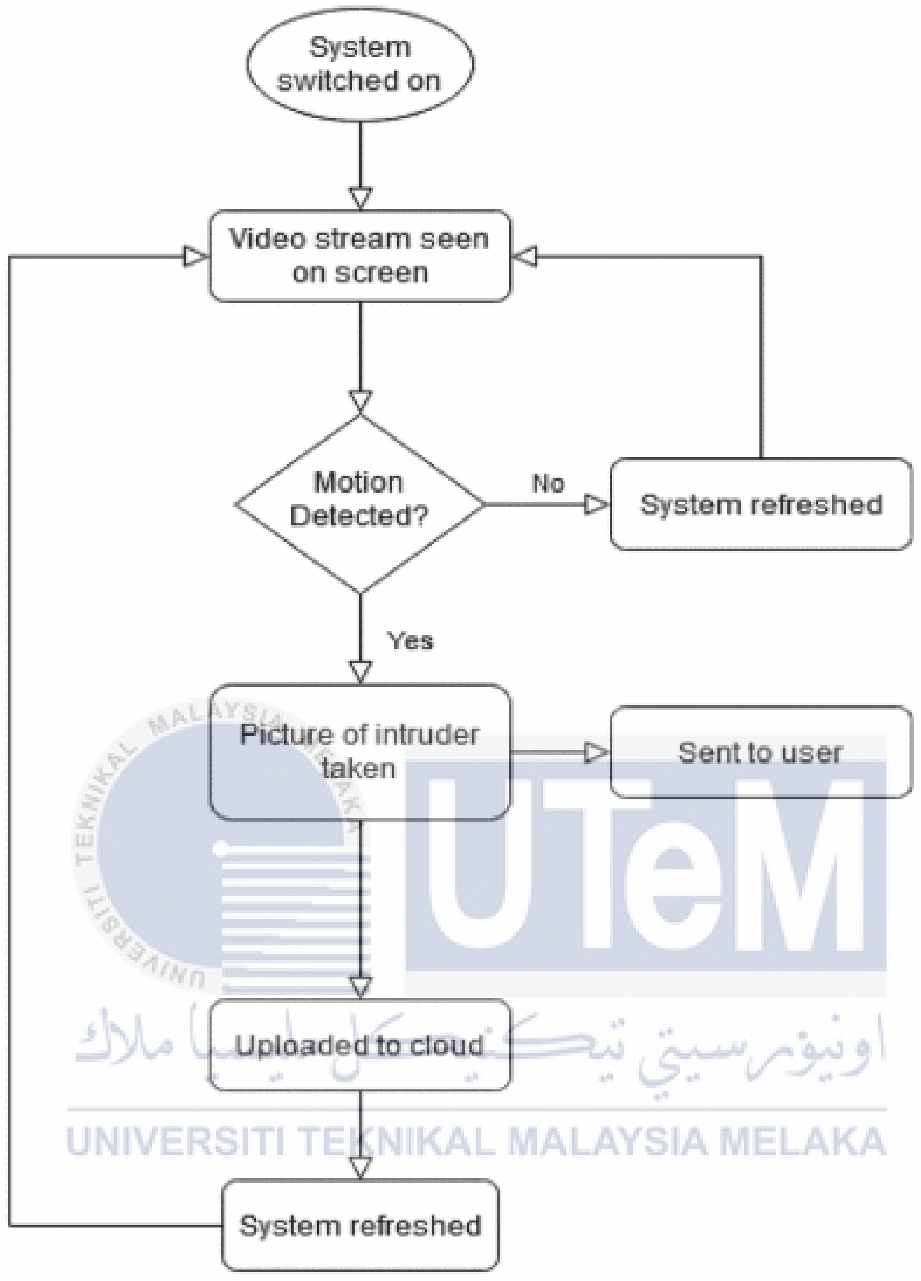


Figure 2.20 Flowchart of smart surveillance system[12]

The paper concludes by emphasizing the versatility and cost-effectiveness of the proposed home automation system. It suggests possible future modifications, such as integrating voice control or incorporating machine learning algorithms for enhanced functionality.

In summary, this research paper presents a comprehensive IoT-based home automation system that incorporates smart energy management, appliance control, and surveillance capabilities[12]. The system demonstrates successful implementation and provides valuable insights for further development and improvements in the field of home automation.

2.2.13 Home Automation including Switching of appliance speed control of ac fan and brightness control of incandescent bulb.

This project had been conducted by Mridul Prasad, Abhishek Pratap Singh, & Yogendra Kumar (2022)[13] in which to propose an IoT-based home automation system using the ESP8266 NodeMCU board. The system utilizes the MQTT protocol and the Blynk IoT platform for communication and control. The aim of the project is to enable remote control of appliances such as fans and bulbs through a mobile app, providing convenience for users, particularly senior citizens and individuals with disabilities[13].

The article begins by introducing the concept of IoT and its application in home automation. It highlights the benefits of IoT in creating a network of interconnected devices and assigns unique IP addresses to each device for identification. The authors then present related research papers that explore topics such as smart fan control, temperature control, environmental monitoring, and home automation using the NodeMCU board and MQTT protocol.

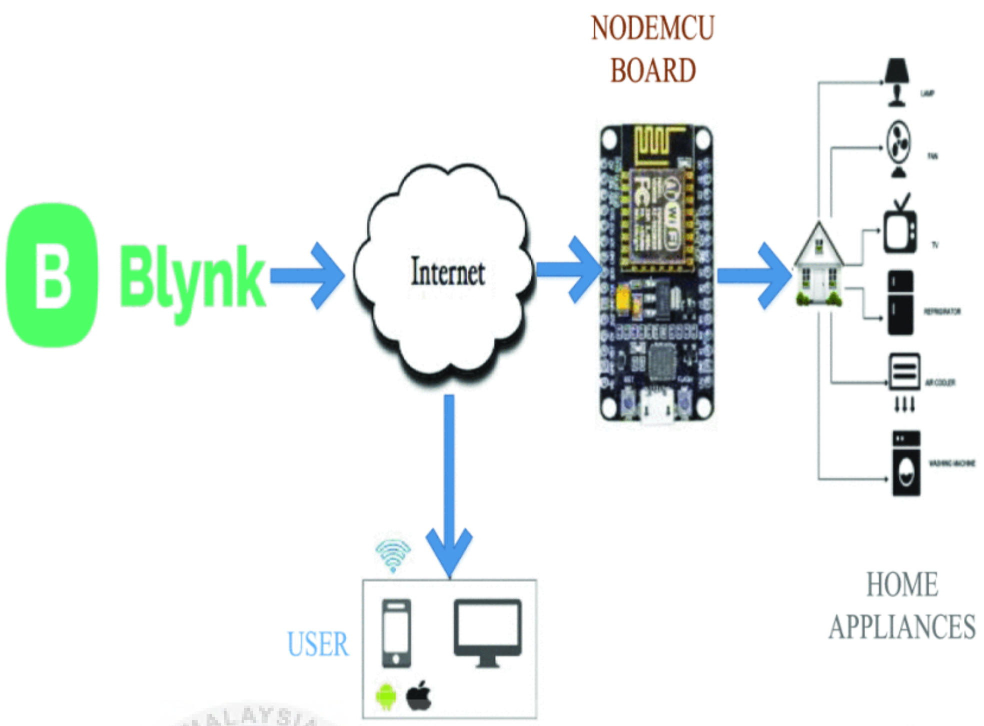


Figure 2.21 Block diagram of proposed IoT based home automation system[13]

The technical approach section describes the block diagram of the proposed home automation system. It illustrates the connections between the ESP8266 NodeMCU board, appliances, the Blynk IoT platform, and the mobile app[13]. The architecture of the NodeMCU board is explained, emphasizing its GPIO pins and PWM capabilities for motor speed control.

The implementation setup section outlines the hardware and software components required for the project[13]. It lists the necessary components, including the NodeMCU board, Proteus software for design simulation, the Blynk app for user interaction, and the Arduino IDE for programming.

The proposed system and methodology are presented through a complete circuit diagram. The circuit includes components such as resistors, bridge rectifier circuits, opt isolators, and triodes for alternating current (TRIACs) which enable phase angle control for appliance switching and speed control. The article explains the operation of the zero-crossing detection circuit and the opt isolators for maintaining electrical isolation.

The testing and discussion section demonstrates the experimental setup and results. It shows the successful switching of light bulbs and the control of fan speed and bulb brightness through the mobile app[13]. The variations in fan speed and bulb voltage are presented, indicating the system's effectiveness.

In conclusion, the article highlights the achievements of the proposed system in terms of appliance control, energy efficiency, low cost, and ease of use. It emphasizes the benefits for elderly and especially abled individuals, as well as the potential for significant energy savings. The project's success is attributed to the combination of the MQTT protocol, the NodeMCU board, and the Blynk IoT platform. Overall, the home automation system presented in the article offers a practical and accessible solution for controlling household appliances remotely.

2.3 Table Comparison

Table 2.1 Table comparison of existing project

No.	Author	Title	Functional	Improvement
1	Suraj, Ish Kool, Dharmendra Kumar& Shovan Barma (2018)[1]	Visual Machine Intelligence for Home Automation[1].	The detected states are updated on an appliance control website, which allows users to remotely monitor and toggle the states of the appliances.	Adding a voice assistance (google assistance) to controlling the relay of the switch beside toggle button in apps or software.
2	Nombulelo CC Noruwana, Pius Adewale Owolawi & Temitope Mapayi, (2020)[2]	Interactive IoT-based Speech-Controlled Home Automation System[2]	The system is communicating with Google Assistant and controls various components and appliances.	Control voice input directly using raspberry pi to minimize the cost. Implement in LCD screen for the output display.

3	Shaik Mulla Shabber, Mohan Bansal, P Mrudula Devi & Prateek Jain (2021)[3]	iHAS: An Intelligent Home Automation Based System for Smart City[3]	Utilizes an IoT framework, including a Wi-Fi enabled microcontroller (ESP8266), a four-channel relay board, and the Blynk app, for remote control and monitoring of home appliances using a smartphone.	Controlling the switch by using voice assistance (google assistance) beside just toggle the switch using app blynk.
4	Batyrzhan K. Akhmetzhanov, Omar Aslan Gazizuly, Zhanserik Nurlan & Nurkhat Zhakiyev (2022)[4]	Integration of a Video Surveillance System Into a Smart Home Using the Home Assistant Platform[4]	This project proposes a modified solution for a cost-effective and user-friendly video surveillance security system based on Raspberry Pi single-board computers and ESP8266 microcontrollers, integrated with the Home Assistant software for home automation control.	Voice assistance (google assistance) by adding the API google assistance to controlling the input and output of the project.

5	Satyendra K. Vishwakarma, Prashant Upadhyaya, Babita Kumari& Arun Kumar Mishra (2019)[5]	Smart Energy Efficient Home Automation System Using IoT[5]	The system utilizes voice commands through Google Assistant and a web-based application for controlling home appliances, while also incorporating security features such as sensors and cameras.	Sensor and camera for security purposes. Besides that, open the scale of voice assistance that not just using google assistance that have in the smartphone otherwise using built in voice assistance.
6	S.K Sooraj, E Sundaravel, Babu Shreesh & K. Sireesha (2020)[6]	IoT Smart Home Assistant for Physically Challenged and Elderly People[6]	The proposed project is an IoT-based smart home system that enables remote surveillance, control of appliances, and voice recognition using Smartphone to Raspberry Pi, PIR sensor, Pi Camera, Arduino, and Bluetooth, providing convenience and	Bigger or open scale the voice assistance that control the switch relay of home automation.

			security for elderly and physically challenged individuals.	
7	Sudiir Mohan, Nithish Arro J, R Sitharthan (2022)[7]	Enhanced Home Automation and Security Using IoT Architecture[7]	The proposed system is a scalable and configurable IoT-based home automation and security system, utilizing Raspberry Pi and ESP8266 nodes, PIR sensors, LDRs, and relay switches, with the ability to detect motion, control lighting based on ambient conditions, and send push notifications to users, resulting in an affordable and user-friendly solution for home automation and security.	Improving by adding the voice assistance (google assistance) to easily control the input and output of the project beside using smart phone application.

8	P Karthik, U G Keerthiv Sanjay, S Abhiram, V Dinesh Kumar (2021)[8]	Implementation of Home Automation using Smart Mirror[8]	The proposed system is a smart mirror integrated with a home automation system, allowing users to control appliances through the mirror, mobile app, or voice assistant, with the implementation using Raspberry Pi as the central control unit.	Improving the security purpose such as adding camera and sensor. Store the data using telegram API at raspberry pi to easily manage the data recording for future investigation.
9	Shradha Somani, Parikshit Solunke, Shaunak Oke, Parth Medhi & P.P. Laturkar (2018)[9]	IoT Based Smart Security and Home Automation[9]	The system utilizes a Raspberry Pi as the central control unit, various sensors for motion detection, gas leakage detection, temperature and humidity monitoring, and an Android application for user interaction and receiving alerts.	Improving the communication of the system flow by adding voice assistance (google assistance) to controlling the input and output of the project.

10	(Harsha Vardhan Tomar, Anagha Anand, H L Harsha, Anubhav Deshwal, & Badari Nath, 2022)[10]	"Smart Home Automation Device" Using Raspberry Pie and Arduino Uno[10]	The proposed smart home automation system integrates features such as intruder detection, access control, smart lighting, gas leakage detection, photo plate display, music player, and weather reporting, providing convenience and enhanced security for homeowners.	Make the screen touch screen and improving the communication of the system flow by adding the voice assistance (google assistance) to controlling the input and output of the project.
11	Swetha Amit, Anitha Sarah Koshy, S Samprita, Shwetali Joshi, & N Ranjitha (2019)[11]	Internet of Things (IoT) enabled Sustainable Home Automation along with Security using Solar Energy[11]	implementation of a solar-powered home automation and security system, integrating Alexa voice commands and a mobile app for remote control, while also showcasing the potential of future advancements in home automation technology.	Adding the system by adding the storing the data at telegram not just using GSM SMS.

12	Shruti Dash & Pallavi Choudekar (2021)[12]	Home Automation using Smart Devices and IoT[12]	the development of a comprehensive home automation system prototype incorporating smart energy metering, IoT-based appliance control, and a smart surveillance system, enabling efficient energy management, remote appliance control, and enhanced security, with potential for future enhancements.	Improving the system by adding the storing the data at telegram when detecting movement while the system in video stream.
13	Mridul Prasad, Abhishek Pratap Singh, & Yogendra Kumar (2022)[13]	Home Automation including Switching of appliance speed control of ac fan and brightness control of incandescent bulb[13]	IoT-based home automation system utilizing the ESP8266 NodeMCU board, Blynk app, and Arduino IDE, enabling remote control and automation of home appliances with features such as fan speed control,	Improving by adding the voice assistance (google assistance) to easily control the input and output of the project beside using Blynk application in smartphone.

			brightness adjustment of incandescent bulbs, and appliance switching, providing cost-effective and user-friendly home automation capabilities.	

2.4 Device Components

2.4.1 Microprocessor

Nowadays, Raspberry Pi was widely used in the home security system to protect belongings, guard residences, and monitor babies or their babysitters. It is a microcomputer, which able to run programs over a certain period. The Raspberry Pi has the ability to interact with the outside world in such it can make a call to the owner when visitor at the door. In the door notification system, the Raspberry Pi was connected to the camera module to produce high-definition video and observe the live streaming through web browser or mobile in real-time. The traditional Closed-Circuit Television (CCTV) can be replaced by this monitoring system, which served better resolutions clarify of IP cameras.There are several types of Raspberry Pi such as Raspberry Pi 3 Model B+, Raspberry Pi 3 Model B and Raspberry Pi Zero W. In order to choose an ideal Pi for the door notification system, the specifications of the Raspberry Pi had been compared.

Raspberry Pi 3 Model A+ proposed with low power consumption, light in weight and small size[14]. This model has 1.4GB 64-bit quad-core Arm Cortex-A53 processor. It has a more restricted space to use which offered only 512MB of RAM that will limit to the larger data size support. The Raspberry Pi 3 Model A+ is slightly complicated to use than the Raspberry Pi 3 Model B+ as it has a single USB port on the board. The USB port is directly connected to the SoC with using a micro-USB (OTG) port[14].

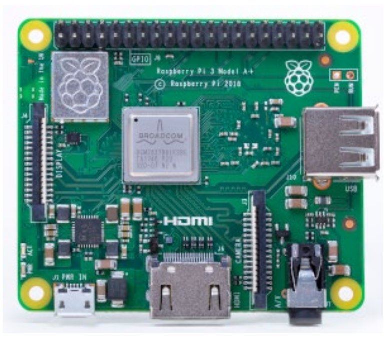


Figure 2.22 Raspberry Pi 3 Model A+[14]

Raspberry Pi 3 Model B+ is the latest version of the Raspberry Pi, which uses a 1.4GHz 64-bit quad-core Arm Cortex-A53 CPU[15]. This allows Raspberry Pi to complete more tasks effectively because it able to execute four tasks simultaneously across its separate cores. It also introduced an advanced computer having the most memory with 1GB of RAM to support larger data sizes and run a lot of work before slowing down as well as becoming unresponsive. However, Raspberry Pi 3 Model B+ considered larger and heavier among of the Raspberry Pi modules. This is because its board have four USB ports can be used to multiple USB devices for instance keyboards, mice, external storage and network adaptor. The next excellent characteristic is Raspberry Pi 3 Model B+ is the board that involve with a dual-band Wi-Fi that provide both 2.4GHz and 5GHz wireless. This feature makes Raspberry Pi 3B+ become adaptable and flexible to everyone. The combination of several radios on one band together with dual-band support provide much higher performance for home networking than what single-band routers can offer[15].

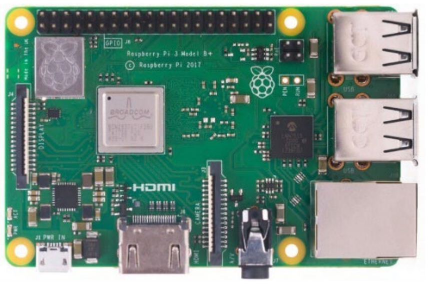


Figure 2.23 Raspberry Pi 3 Model B+[15]

Raspberry Pi Zero Wireless (Raspberry Pi Zero W) is the latest product of the Pi Zero family by building in wireless LAN connectivity[16]. It is relatively low cost and uses the 1GHz single core processor that performed lower processing capabilities than Raspberry Pi 3 Model B+. It has only 512MB of RAM and 1 micro-USB socket only in which a USB OTG cable needed for devices connection[16]. It also delivers Bluetooth connectivity to free up multiple connection devices by replacing a Bluetooth keyboard or mouse. However, its setup is more complicated when compared to the other Pi version. Because of the significantly small in size, the connectors are not considered as standard. The Raspberry Pi Zero W powered by 5V of power supply and needed a micro-SD when operation.

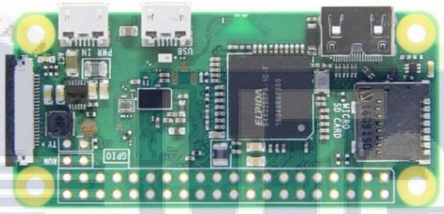


Figure 2.24 Raspberry Pi Zero Wireless[16]

The Raspberry Pi 4 Model B represents the most recent advancement in the Raspberry Pi series, packing a powerful 1.5GHz 64-bit quad-core Arm Cortex-A72 CPU[17], [18]. This ensures the Raspberry Pi can manage more functions effectively, courtesy of its ability to carry out four tasks concurrently across its discrete cores. Furthermore, it sets a new benchmark with its most substantial memory offering to date - 2GB, 4GB, or 8GB of LPDDR4 RAM - facilitating the handling of larger datasets and operating numerous tasks before encountering any performance slowdown or responsiveness issues[17], [18]. Despite offering superior capabilities, the Raspberry Pi 4 Model B remains slightly larger and heavier compared to other models in the series. This is attributable to its array of four USB ports that accommodate multiple devices such as keyboards, mice, external storage, and network adapters. One of its most noteworthy features is its dual-band Wi-Fi, supporting both 2.4GHz and 5GHz wireless connections. This flexibility makes the Raspberry Pi 4B an adaptable choice for many users. The amalgamation of various radios on a single band, along with the dual-band support, delivers a significantly enhanced performance for home networking, outpacing what single-band routers can offer.

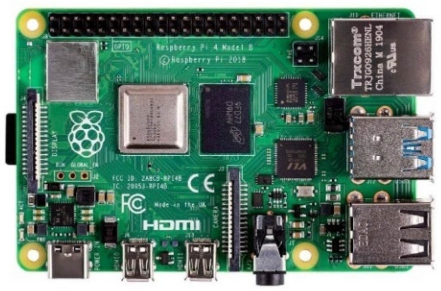


Figure 2.25 Raspberry Pi 4 Model B[17], [18]

Table 2.2 Comparison of Raspberry Pi[14]–[16], [18]

		Raspberry Pi 3 Model A+	Raspberry Pi 3 Model B+	Raspberry Pi Zero Wireless	Raspberry Pi 4 Model B
Specification	SoC/CPU	BCM2837B0 1.4GHz Cortex-A53 64-bit	BCM2837B0 1.4GHz Cortex-A53 64-bit	BCM2835 1GHz ARM1176JZF-S	BCM2711 1.5Hz Cortex-A72 64-bit
	RAM	512MB	1GB	512MB	2GB
	USB	1	4	1 Micro USB socket	4
	Wi-Fi	Dual Band 2.4GHz & 5GHz 802.11b/g/n/ac	Dual Band 2.4GHz & 5GHz 802.11b/g/n/ac	802.11n Wireless LAN	Dual Band 2.4GHz & 5GHz 802.11b/g/n/ac
	Bluetooth	Bluetooth 4.2	Bluetooth 4.2	Bluetooth 4.0	Bluetooth 5.0
	Video Output	HDMI	HDMI	Mini-HDMI	Mini-HDMI
	Audio Output	3.5mm jack	3.5mm jack	None	3.5mm jack
	Size	65mm x 56mm x 11mm	65mm x 56mm x 17mm	65mm x 30mm x 5mm	85.6mm x 56mm x 11mm

Raspberry Pi 4 Model B is an excellent value for money. It also brings essential changes to networking and offers better processor as compared to the other models of Raspberry Pi to operate faster. Every Pi has the same GPU, they all play HD video effortlessly, thus using it as a media centre is a good option. It has maximum 1GB RAM that system-on-a- chip (SoC) can be supported. The Raspberry Pi 4 Model B can be considered as an ideal microprocessor embedded in the door notification system as it has four USB ports, faster CPU, low power hungry-package and the most user-friendly Pi.

2.4.2 Camera

Camera plays an important role in surveillance system which operates on the advanced video camera technology. Basically, it connected with the networks to allow the network connection between the remote area and the assigned security area. This surveillance camera is widely used as a crime deterrent to monitor and record the suspicious activity used for police investigation.

Infrared Night Vision Surveillance Camera is a small size, light and user-friendly Raspberry Pi camera. It is able to support night vision and is compatible for both Raspberry Pi model A and Model B. It can connect directly with Camera Serial Interface (CSI) that offered high data rates capability and delivers pixel data to the BCM2835 processor. It provides normal field of view with 70 degree and support 1080p@30 fps, 720p@60 fps and 640x480p 60/90 in recording video and a maximum resolution of 2592x1944 pixels for static image[19]. This allows the image and video captured in larger or clearer format compared to traditional camera. However, the image or video are process in black and white colour.

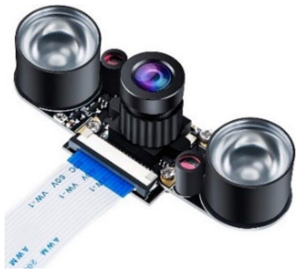


Figure 2.26 Infrared Night Vision Camera[19]

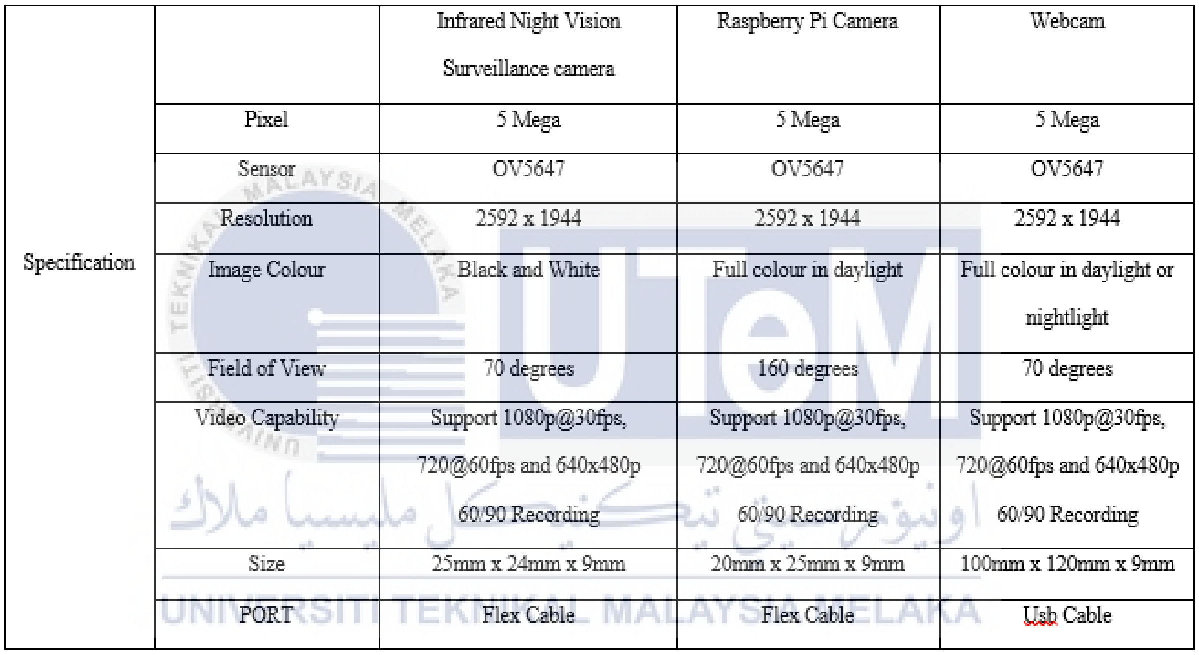
Raspberry Pi camera 5-Megapixel Camera Module is compatible with both Raspberry Pi A and Raspberry Pi B[20]. It is equipped with OV5647 sensor and offered 5 Megapixel resolution image or HD video[20]. The specifications of this module are almost same with the Infrared Night Vision Surveillance Camera in term of pixel, resolution and video capability. This module offers a wider field of view that covered up to 160 degrees to provide a wide vision for owner. It is also able to deliver full colour image in daylight and provide better night vision because this module equipped with IR cut switch, which automatically switch the infrared technology based on the light intensity.

Figure 2.27 Raspberry Pi[20]

The Webcam Camera is a compact, lightweight, and user-friendly device designed for use with Raspberry Pi. It does not support night vision and is specifically compatible with Raspberry Pi models A and B. The camera connects directly to the Camera Serial Interface (CSI), providing high data transfer rates and delivering pixel data to the BCM2835 processor. It offers a standard field of view of 70 degrees and supports various video recording resolutions, including 1080p at 30 fps, 720p at 60 fps, and 640x480p at 60/90 fps. For still images, it can capture a maximum resolution of 2592x1944 pixels, resulting in larger and clearer images compared to traditional cameras. It's important to note that the Webcam Camera processes images and videos in full colour, unlike the black and white processing of infrared cameras.

Figure 2.28 Example USB Camera

Table 2.3 Comparison of Camera[19], [20]



		Infrared Night Vision Surveillance camera	Raspberry Pi Camera	Webcam
Specification	Pixel	5 Mega	5 Mega	5 Mega
	Sensor	OV5647	OV5647	OV5647
	Resolution	2592 x 1944	2592 x 1944	2592 x 1944
	Image Colour	Black and White	Full colour in daylight	Full colour in daylight or nightlight
	Field of View	70 degrees	160 degrees	70 degrees
	Video Capability	Support 1080p@30fps, 720@60fps and 640x480p 60/90 Recording	Support 1080p@30fps, 720@60fps and 640x480p 60/90 Recording	Support 1080p@30fps, 720@60fps and 640x480p 60/90 Recording
	Size	25mm x 24mm x 9mm	20mm x 25mm x 9mm	100mm x 120mm x 9mm
	PORT	Flex Cable	Flex Cable	Usb Cable

Surveillance camera is one of the famous tools for homeowner to monitor and live stream in real-time at anywhere. It is also able to deliver a crime evidence video when trespass happened. Thus, the Wide Angle FOV1600 5Megapixel Camera is better Pi camera to be selected as it provides full colour image or video and a wide range of view for better vision. It is also fully compatible with the Raspberry Pi as it supports in the operating system of Raspberry Pi, which known as Raspbian.

2.4.3 Sensor

Sensor was mainly used to detect a trespasser and trigger the monitoring centre in the surveillance system. Sensors are also usually found in electronic appliances and gadgets used in homes. There are several type of sensor such as Passive Infrared (PIR) based motion detector sensor, Photosensor, Ultrasonic sensor and Motion Detection Algorithm.

PIR sensor is designed to identify slow surrounding changes, it responds by providing feedback once motion was detected. PIR based motion detector, which also can be considered as passive detector senses the infrared signal emitted by animals or human body. When the movement of human within approximately 10m from PIR sensor, it will receive thermal energy and a motion was detected. It required 5V of power supply to operate and respond to the change in infrared level subjected by human motion. It is small, low-price, low power and user-friendly. However, it cannot function well in an environment that changes abruptly and also under sunlight exposure or in cold condition[21].

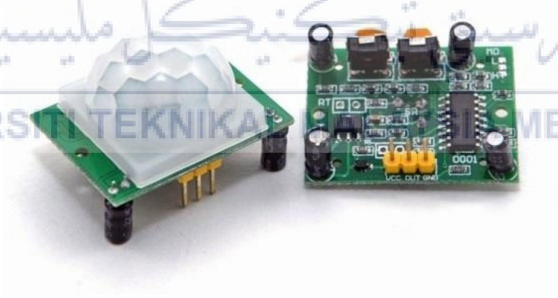


Figure 2.29 PIR motion sensor[21]

PIR sensors are commonly used to detect motion, particularly when a human enters or exits the sensor's range. They are small, inexpensive, low-power devices widely used in various appliances and gadgets found in homes and businesses. These sensors are also known as PIR, "Passive Infrared," "Pyroelectric," or "IR motion" sensors.

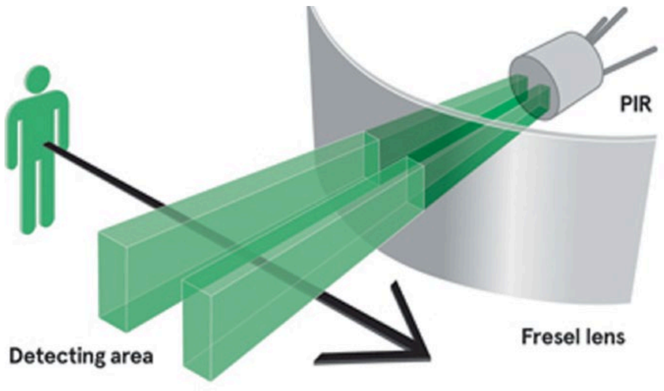


Figure 2.30 PIR motion sensor detection range[22]

The main component of a PIR sensor is the pyroelectric sensor, which is typically housed in a round metal can with a rectangular crystal in the centre. This sensor can detect levels of infrared (IR) radiation. Since everything emits some level of radiation, and hotter objects emit more radiation, the PIR sensor can sense these differences.

A motion detector using a PIR sensor is designed with two halves. This configuration is used to detect motion or changes in the IR levels rather than the average IR levels. The two halves of the sensor are connected in a way that cancels out the signal when the sensor is idle. Both halves detect the same amount of IR radiation emitted by the surroundings, such as the room, walls, or outdoors.

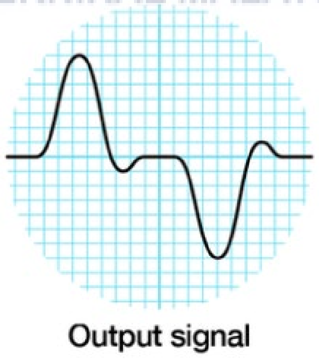
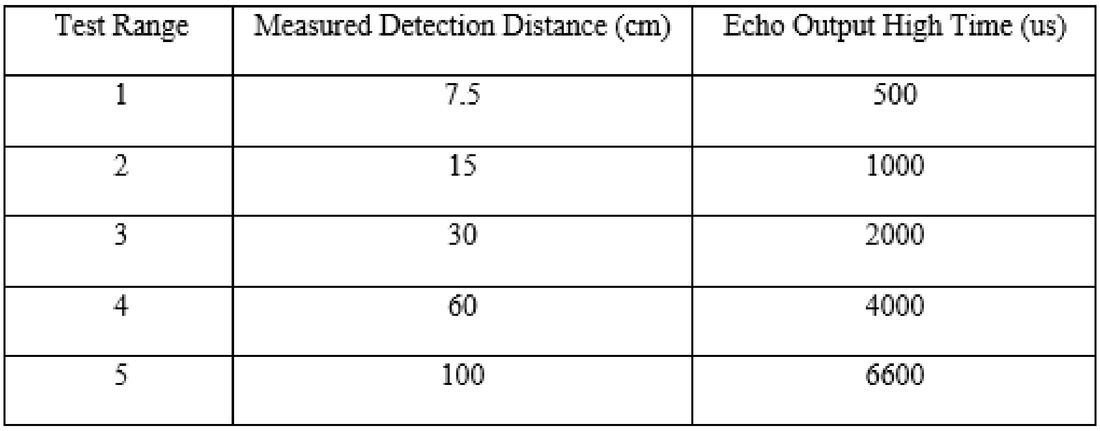


Figure 2.31 changing pulses when object was detected by sensor[22]

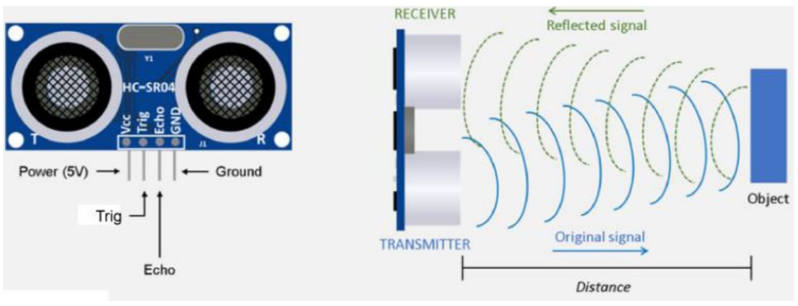
When a warm body, like a human or animal, passes by the sensor, it intercepts one-half of the PIR sensor, causing a positive differential change between the two halves. This change indicates the presence of motion. The reverse happens when the warm body moves away from the sensing area, resulting in a negative differential change. These change pulses are what the sensor detects to determine motion[22]. PIR sensors use the principles of detecting differential changes in IR radiation to sense motion. They consist of a pyroelectric sensor, which is split into two halves. They are widely used in various applications due to their small size, low cost, low power consumption, and reliability.

Security system based on Ultrasonic Sensor Technology that detects motion by sending a specific frequency of sound wave and reflected back to the sensor. Basically, the ultrasonic refers to inaudible sound with frequency above 20kHz and this sensor consist of two part, which are transmitter and receiver[23]. The transmitter transmits the ultrasonic wave while receiver receives the echo from the target when motion was detected. The detection accuracy of Ultrasonic Sensor will affect in a transparency object or a dark environment. The sensor is easy to use and no produce harm to the user during operation. Table 2.4 showed the maximum coverage range of the sensor is 1 meter. It is hard for the Ultrasonic Sensor to detect the soft fabric object as soft material will absorb the reflected sound wave to the receiver of Ultrasonic Sensor.

Table 2.4 Test range and detection measured[23]



Test Range	Measured Detection Distance (cm)	Echo Output High Time (us)
1	7.5	500
2	15	1000
3	30	2000
4	60	4000
5	100	6600



Detection of Ultrasonic sensor on an object[23]

The Ultrasonic Sensor can be considered as the ideal sensor to embed in the door notification system as it does not highly affected by environments and suitable to install at outdoor. The PIR Motion detector is more sensitive to the changing of heat sources and sunlight, so PIR Motion detector is more suitable for the indoor movement detection within the closed environment.

2.5 Summary

In chapter Two is going to look carefully at important books and earlier work (journal) that are connected to this project. It will also include a talk about the main ideas and how the project works. We will compare different parts of the device used in the project to help choose the best ones.

This chapter aims to help us fully understand the technology that the project is based on. It also wants to explain why we choose certain parts for the device. In this chapter, we will bring together all the knowledge from different places to give a clear and easy-to-understand explanation of the technology and parts used in our project.

CHAPTER 3

METHODOLOGY

3.1 Introduction

This chapter discusses the methodology of this project. It divides into several parts to ensure the objectives are achieved. The proposed system consists of three main components, which are Raspberry Pi, webcam, microphone, speaker and esp module. The flowchart, block diagram, hardware and software implementation of this home security system will be presented in this chapter.

3.2 Project Workflow

The project workflow includes five stages in order to ensure that the project is completed smoothly. The details of each stage as shown in the Figure will be discussed in this part. Preliminary Investigation is the primary stage in every establishment of a project. In this stage, the idea for this project is explored and hence, a list of objectives is set to provide a more detailed explanation of the result of the project.

The information and data of the literature that related with this project was collected from reference books, journals, or any reliable website. All the information and results obtained were recorded in the previous chapter. A comparison between components in the existing system had been made to select suitable component for this project. This will help to save a lot of time on the decision-making part. Software and Hardware Implementation will start and be followed by testing the developed system. The system is repeatedly troubleshot and modified so that it achieves the expected outcome at the end of the project. Lastly, analysis can be performed by comparing the characteristics of the developed system with the traditional system.

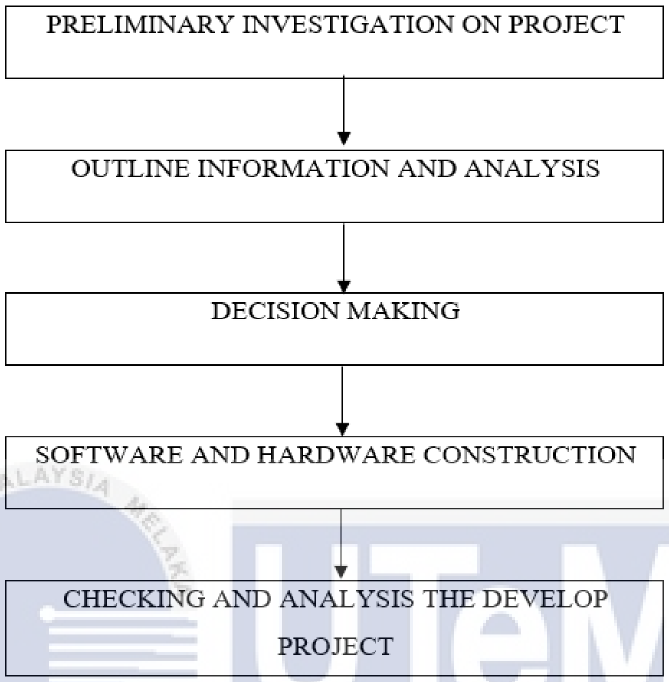

PRELIMINARY INVESTIGATION ON PROJECT
OUTLINE INFORMATION AND ANALYSIS
DECISION MAKING
SOFTWARE AND HARDWARE CONSTRUCTION
CHECKING AND ANALYSIS THE DEVELOP PROJECT

Figure 3.1 Overview of project workflow

3.3 Project System Architecture

In this project, a Raspberry Pi 4, a webcam, and a smart mirror are ingeniously combined to form an advanced system for home automation and security. The smart mirror, empowered by Google Assistant, offers interactive and voice-controlled functionalities, transforming the traditional mirror into a smart, informational hub.

The integrated HC-SR04 ultrasonic sensor detects movement or the presence of objects at the home's entrance. When the sensor picks up any motion, the webcam, another integral part of this system, is triggered to start video recording.

At the same time, a notification message is dispatched to the homeowner, leveraging the Wi-Fi module. This notification is received through the homeowner's Telegram account.

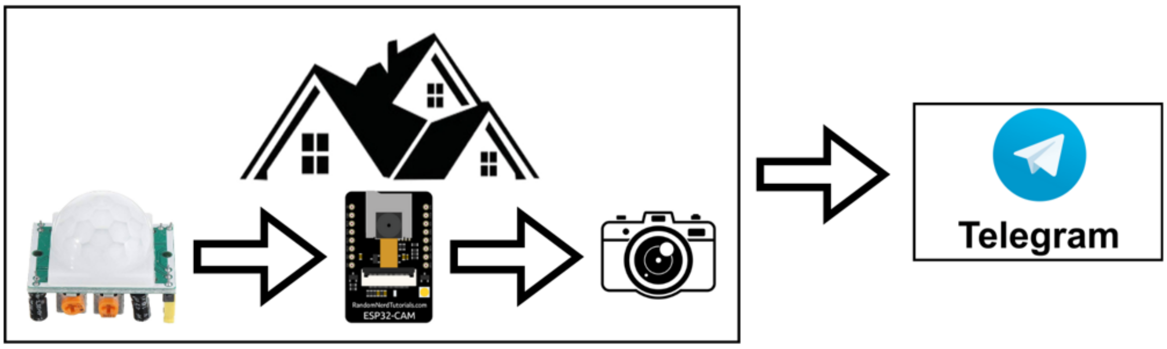


Figure 3.2 Block diagram of IoT security data system

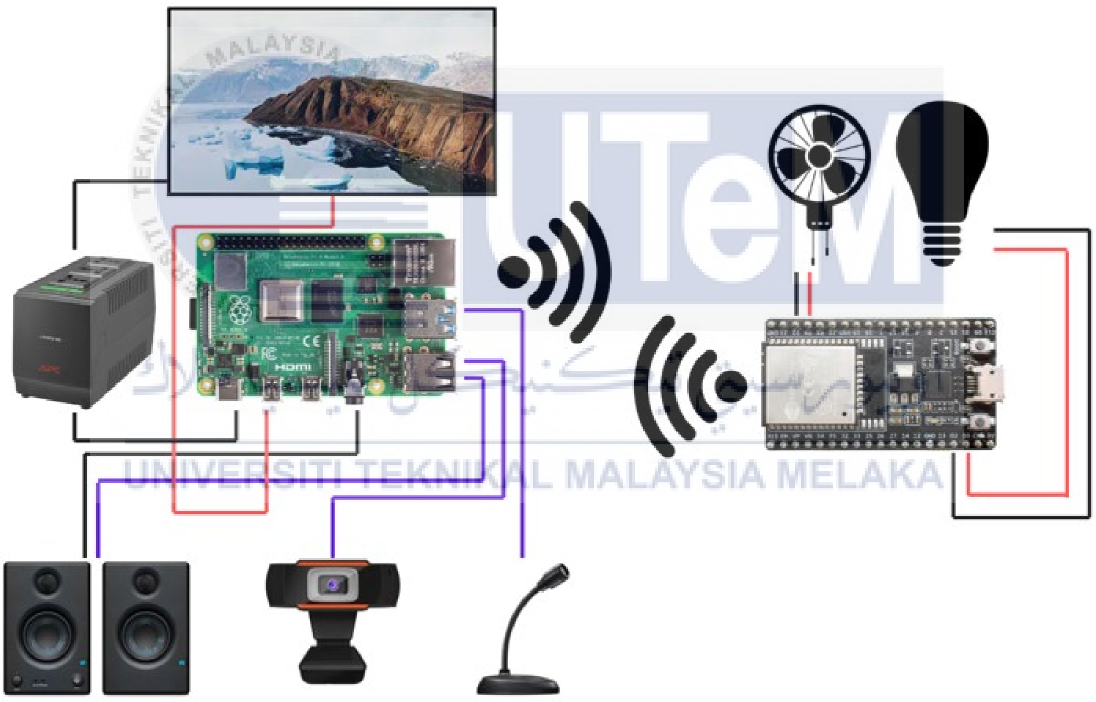


Figure 3.3 Overview of hardware wiring diagram

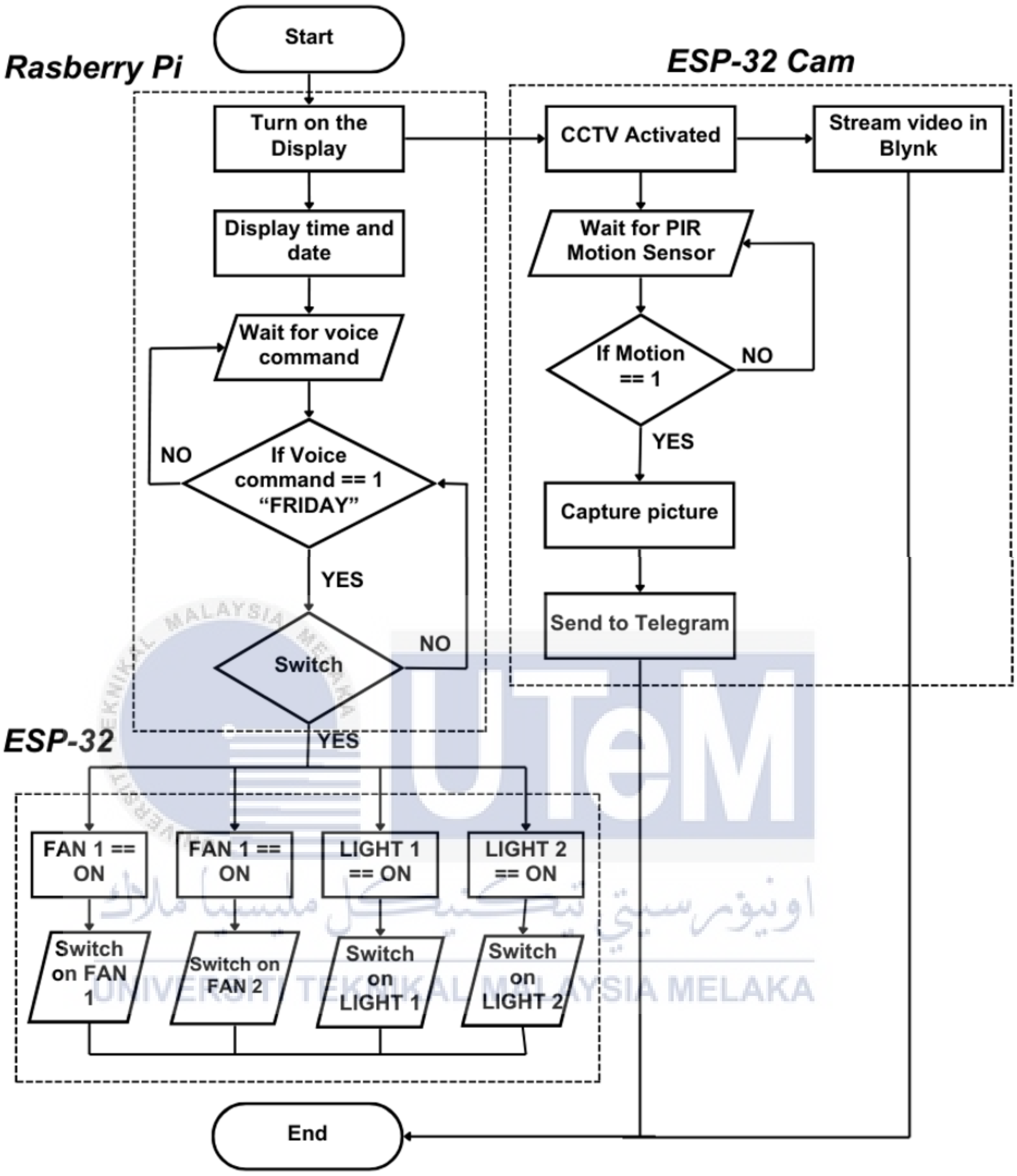


Figure 3.4 Flowchart of project proposed.

3.4 Hardware

3.4.1 Raspberry Pi 4B

The Raspberry Pi has a powerful characteristic, which is the row of general-purpose input or output (GPIO) pins locate along at the top edge of the board. There are two 5V and two 3.3V with a number of ground (GND) pins on the board. A GPIO pin dedicated as an output pin can be set to high (3.3V) or low (0V)[18]. The GPIO pins can be used for a wide range of functions. The GPIO pins are control using a number of programming languages for example like Python language. The detailed pin configuration of Raspberry Pi is illustrated in the Table 3.1 and Figure 3.5.

Table 3.1 Pin configuration of Raspberry Pi[17]

Pin Group	Pin Name	Description
Power Source	+5V, +3.3V GND and Vin	+5V power output +3.3V power output GND ground pin
Communication Interface	UATR Interface (RXD, TXD) [(GPIO15, GPIO14)]	Universal Asynchronous Receiver Transmitter (UART) used for interfacing sensors and other devices.
	SPI Interface (MOSI, MISO, CLK, CE) x2 [SPI0 (GPIO10, GPIO11, GPIO09, GPIO8)] [SPI1(GPIO20, GPIO19, GPIO21, GPIO7)]	Serial Peripheral Interface (SPI) use for communicating with other board or peripherals.
	TWI Interface (SDA, SCL) x2 [(GPIO2, GPIO3)] & [(ID_SD, ID_SC)]	Two Wire Interface (TWI) Interface can be used to connect peripherals.
Input and Output Pins	26 I/O	Although these some pins have multiple functions they can be considered as I/O pins.
PWM	Hardware PWM available on GPIO12, GPIO13, GPIO18, GPIO19	These 4 channels can provide Pulse Width Modulation (PWM) outputs. Software PWM available on all pins.
External Interrupt	All I/O	In the board all I/O pins can be used as Interrupts.

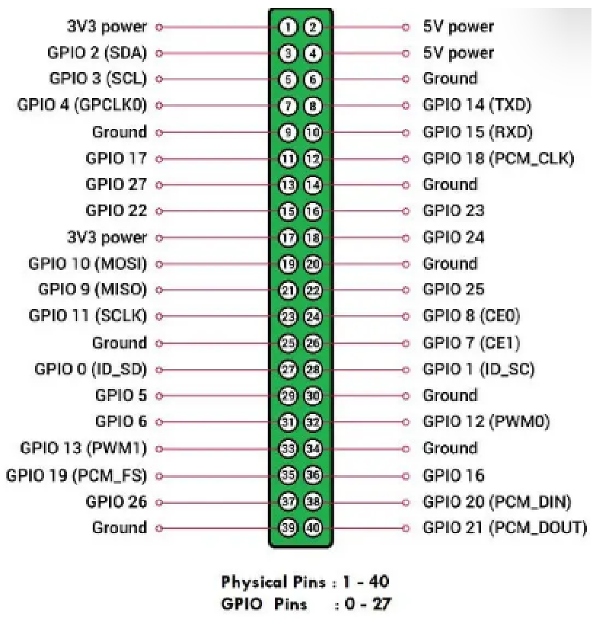


Figure 3.5 R-pi GPIO Pinout[17]

The Raspberry Pi 4 exhibits a number of significant advancements compared to its predecessor, the Raspberry Pi 3B+. Its central processing unit (CPU) is a 1.5GHz 64-bit quad-core ARM Cortex-A72 processor, part of the Broadcom BCM2711 chip and operating on an ARMv8-architecture core. The graphics processing unit (GPU) is the Broadcom Video Core VI, running at 500 MHz, released in 2009, capable of Blu-ray quality playback and OpenGL ES 3.0 graphics. Depending on the variant, it has either 2GB, 4GB, or 8GB of LPDDR4 SDRAM[18]. Connectivity is flexible, with two USB 3.0 and two USB 2.0 ports available for peripheral devices. Power is supplied via a 5.1V, 3A USB type-C port. Visual output is catered for by two micro-HDMI ports capable of delivering up to 4k@60HZ resolution. Network connectivity is provided by a true Gigabit Ethernet port, while audio and composite video outputs are combined in a single 4-pole 3.5mm socket. Finally, a micro-SD card slot serves booting and storage purposes.

Table 3.2 Board connectors of Raspberry Pi 4B

Name	Description
Ethernet	Base T Ethernet Socket
USB	3.0 (Two sockets) 2.0 (Two sockets)
Audio Output	3.5mm Jack
Video Output	Micro HDMI
Camera Connector	15-pin MIPI Camera Serial Interface (CSI-2)
Display Connector	Display Serial Interface (DSI) 15 way flat flex cable connector with two data lanes and a clock lane
Memory Card Slot	Push/Pull Micro SDIO

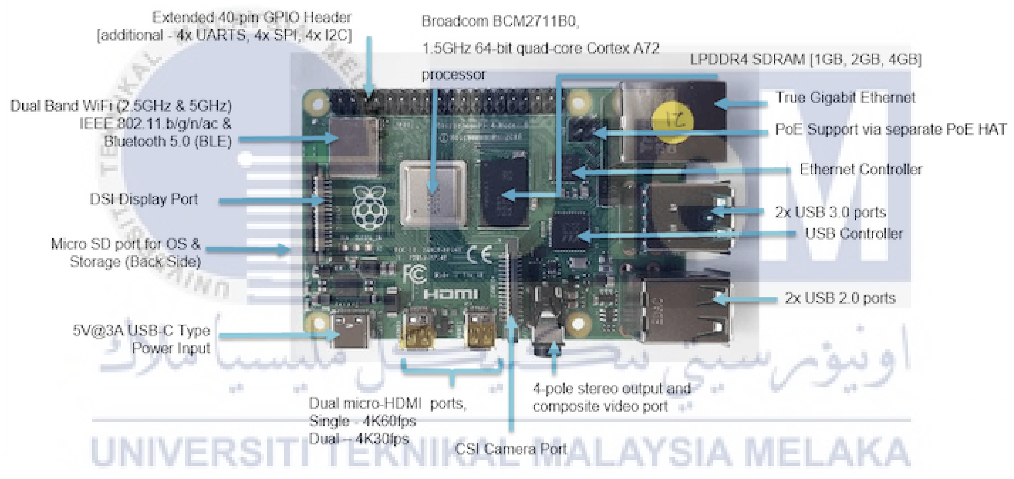
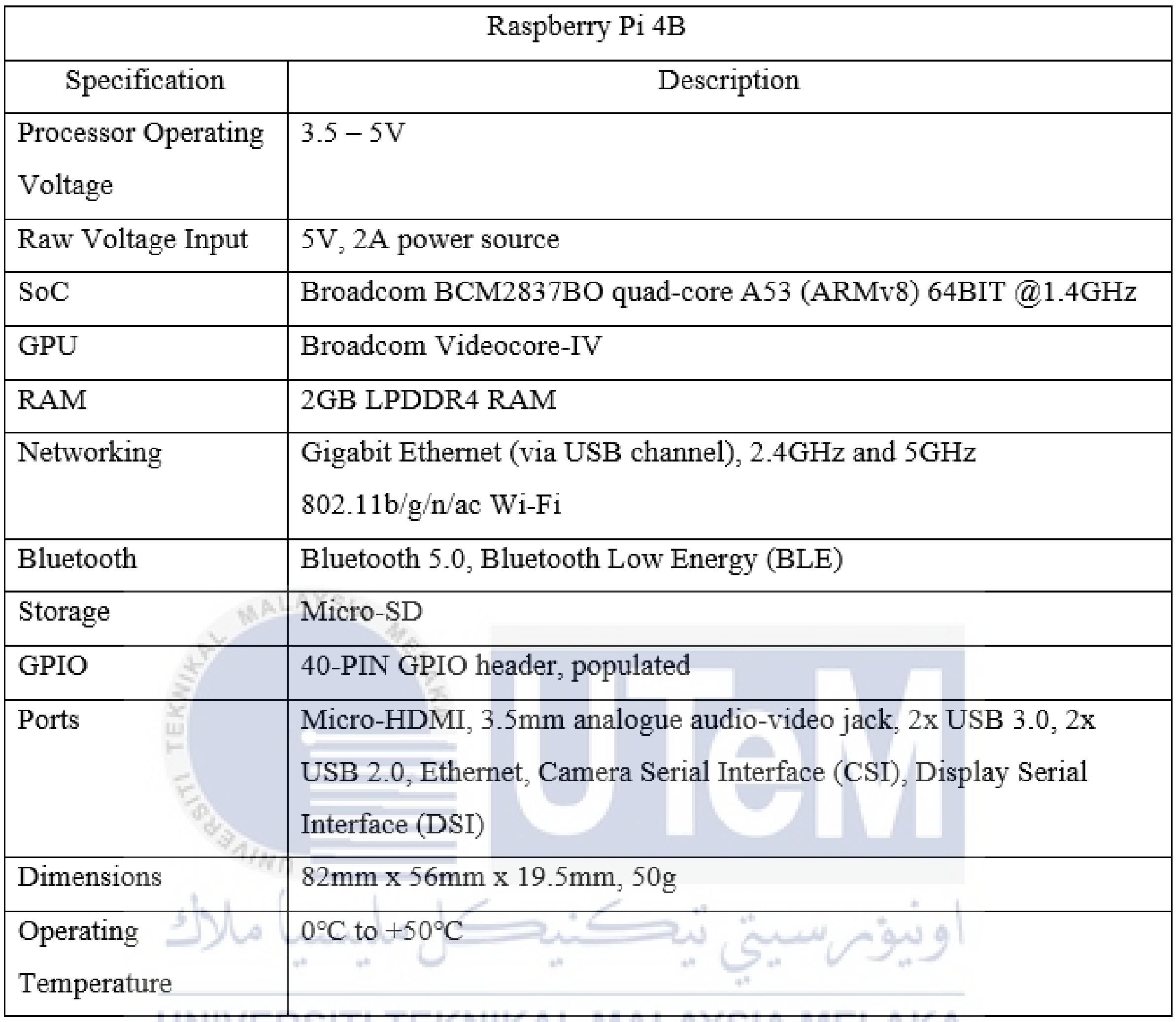

Extended 40-pin GPIO Header [additional - 4x UARTS, 4x SPI, 4x I2C]
Broadcom BCM2711B0, 1.5GHz 64-bit quad-core Cortex A72 processor
LPDDR4 SDRAM [1GB, 2GB, 4GB]
Dual Band WiFi (2.5GHz & 5GHz) IEEE 802.11.b/g/n/ac & Bluetooth 5.0 (BLE)
True Gigabit Ethernet
PoE Support via separate PoE HAT
Ethernet Controller
DSI Display Port
2x USB 3.0 ports
USB Controller
Micro SD port for OS & Storage (Back Side)
2x USB 2.0 ports
5V@3A USB-C Type Power Input
Dual micro-HDMI ports, Single - 4K60fps Dual - 4K30fps
4-pole stereo output and composite video port
CSI Camera Port

Figure 3.6 Anatomy of Raspberry Pi 4B[17]

Table 3.3 Raspberry Pi 4B specifications[17]



Raspberry Pi 4B	
Specification	Description
Processor Operating Voltage	3.5 – 5V
Raw Voltage Input	5V, 2A power source
SoC	Broadcom BCM2837BO quad-core A53 (ARMv8) 64BIT @1.4GHz
GPU	Broadcom Videocore-IV
RAM	2GB LPDDR4 RAM
Networking	Gigabit Ethernet (via USB channel), 2.4GHz and 5GHz 802.11b/g/n/ac Wi-Fi
Bluetooth	Bluetooth 5.0, Bluetooth Low Energy (BLE)
Storage	Micro-SD
GPIO	40-PIN GPIO header, populated
Ports	Micro-HDMI, 3.5mm analogue audio-video jack, 2x USB 3.0, 2x USB 2.0, Ethernet, Camera Serial Interface (CSI), Display Serial Interface (DSI)
Dimensions	82mm x 56mm x 19.5mm, 50g
Operating Temperature	0°C to +50°C

3.4.2 HC-SR04 Ultrasonic sensor

The HC-SR04 is a cost-effective and straightforward sensor for measuring distance, with a range spanning from 2cm to 400cm, approximately one inch to 13 feet. This sensor incorporates two ultrasonic transducers - a transmitter that emits ultrasonic sound pulses and a receiver that listens for these pulses when they are reflected back, operating much like a SONAR system found in submarines for object detection under water.

In the context of using this sensor with a Raspberry Pi, the HC-SR04 has four pins. The VCC and GND pins are connected to the respective 5V and GND pins on the Raspberry

Pi[23]. The Trig and Echo pins can be connected to any available GPIO pins on the Raspberry Pi. The Trig pin is utilised to trigger the emission of the ultrasound wave from the transmitter, while the Echo pin is used to detect the returning reflected signal.

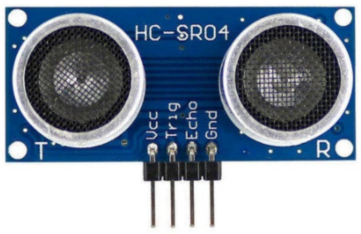


Figure 3.7 Pin of HC-SR04 Ultrasonic sensor[23]

The HC-SR04 ultrasonic distance sensor operates by emitting an ultrasound at 40,000 Hz, which travels in the air until it encounters an object or obstacle and bounces back to the module. By analysing the speed of sound and the time taken for the sound wave to return, the distance to the object can be calculated[23]. The ultrasound is generated by setting the Trig pin to a High State for 10 µs, initiating an 8-cycle ultrasonic burst that moves at the speed of sound.

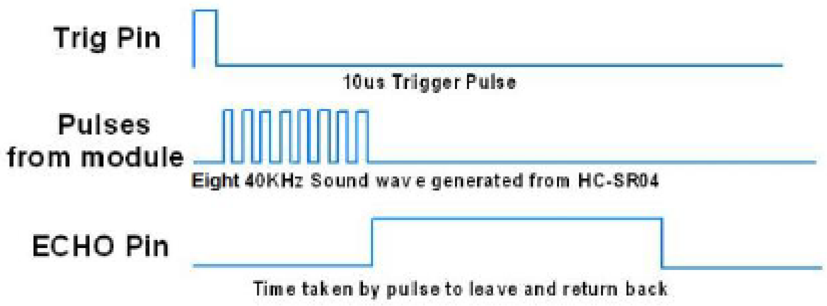

Trig Pin
10us Trigger Pulse
Pulses from module
Eight 40KHz Sound wave generated from HC-SR04
ECHO Pin
Time taken by pulse to leave and return back

Figure 3.8 HC-SR04 Ultrasonic sensor timing diagram[23]

The Echo pin then goes high and begins to listen for the reflected wave. If no reflection is detected, the Echo pin reverts to a low state after a 38ms timeout. However, if a reflection is detected, the Echo pin goes low before the 38ms, indicating a shorter travel time and therefore a closer object.

Distance = Speed x Time.

The basic formula for calculating distance, 'Distance = Speed x Time', is used, taking into account the time the Echo pin remained high and the speed of sound (340m/s). However, as we are measuring the time for the sound wave to travel to the object and bounce back, the resulting calculation is halved to provide the accurate distance.

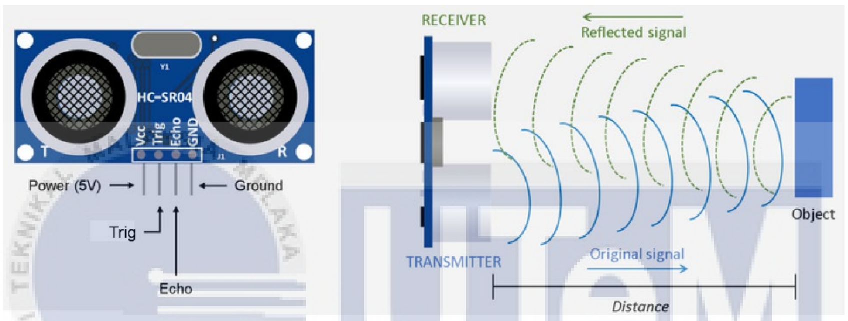


Figure 3.9 The detection of HC-SR04 Ultrasonic sensor on an object[23]

3.4.3 Camera Webcam

Figure 3.10 FULL HD 1080P Webcam Web Camera

Introducing the FULL HD 1080P Webcam Web Camera with Microphone. This high-quality webcam is designed to enhance video conferencing and online communication experiences. With its built-in sound absorption microphone, can enjoy crystal-clear audio even from a distance of 10 meters, eliminating the need for awkwardly leaning towards the camera. The webcam boasts high-definition pixels, delivering actual colour images that bring video quality to life. Its left, and right 360-degree adjustability allows to easily customize the camera angle to suit preferences. With a maximum resolution of 1920x1080, this webcam ensures sharp and clear visuals. Model number U01 guarantees its reliability, while the auto-focus feature ensures that image stays sharp even during movement. The webcam is USB interface-based, making it compatible with various devices such as computers, PCs, and laptops. It is equipped with a 2-megapixel CMOS image sensor, offering impressive image clarity. The webcam operates at a frame speed of 30FPS, delivering smooth video playback. With a 1.5-meter wire length, it provides flexibility in positioning. Notably, the webcam is driverless, eliminating the hassle of software installations. It effortlessly connects via USB 3.0, ensuring efficient data transfer. In addition to its remarkable video capabilities, this webcam also includes a built-in microphone, ensuring that voice is captured with precision. The FULL HD 1080P Webcam Web Camera with Microphone is the ideal companion for online meetings, virtual classrooms, and video chats, allowing to communicate seamlessly and effortlessly.

3.4.4 Camera ESP 32 Cam

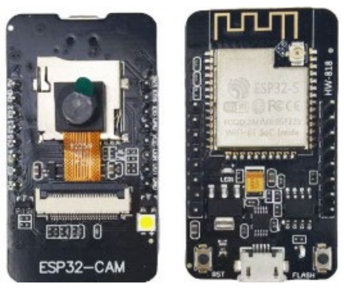


Figure 3.11 ESP32-Cam Camera

The ESP32-CAM is a small-size, low-power camera module based on the ESP32-S module. It comes with an OV2640 camera and provides an onboard TF card slot. This board has 4MB PSRAM which is used for buffering images from the camera into video streaming or other tasks and allows you to use higher quality in your pictures without crashing the ESP32. It also comes with an onboard LED for flash and several GPIOs to connect peripherals.

The ESP32-CAM is equipped with the ESP-32S WIFI module and supports WiFi + Bluetooth. It has a built-in flash of 32Mbit and RAM of internal 512KB + external 4M PSRAM. The WiFi protocol it supports is IEEE 802.11 b/g/n/e/i and it has Bluetooth 4.2 BR/EDR and BLE. The WIFI mode includes Station / SoftAP / SoftAP+Station and it has security features such as WPA/WPA2/WPA2-Enterprise/WPS.

The ESP32-CAM supports WiFi video monitoring and WiFi image upload. It supports multiple sleep modes, with deep sleep current as low as 6mA. The control interface is accessible via pin-header, making it easy to be integrated and embedded into user products. It also supports output image formats such as JPEG (OV2640 support only), BMP, and GRAYSCALE. The ESP32-CAM is suitable for IoT applications such as smart home devices and image upload.

3.5 Software Implementation

3.5.1 Raspberry Pi

Raspberry Pi is a versatile single-board computer widely used in IoT (Internet of Things) projects. Its software ecosystem offers various operating systems tailored for IoT, including Raspbian, Ubuntu Core, and Windows 10 IoT Core. Developers can program Raspberry Pi using Python, C/C++, or web technologies like JavaScript and Node.js. The board provides built-in Wi-Fi, Ethernet, Bluetooth, and USB interfaces for connectivity. Protocols like MQTT, HTTP, and CoAP enable seamless communication with IoT devices and platforms. The extensive collection of IoT frameworks and libraries simplifies development, such as Eclipse Paho MQTT, Adafruit, and Pi4J. Raspberry Pi integrates smoothly with cloud platforms like AWS IoT, Azure IoT Hub, and Google Cloud IoT Core for data management and analysis. With these software capabilities, Raspberry Pi empowers developers to create innovative and scalable IoT projects efficiently.

Figure 3.12 Raspberry Pi[17]

Raspberry Pi is a simple computer on a single board, it required many steps as compared to simple microcontroller. In order to initialize the Raspberry Pi, a specific operating system is required to be installed. The programming of Raspberry Pi will discuss in step by step as in below:

1) The 16GB micro-SD card was used and devoted specifically for Raspberry Pi Operating System.
2) OS software was selected and downloaded on the website of http://www.raspberrypi.org/downloads/ .
3) The SD card was formatted, and OS was installed on to the SD memory card using convenient methods.
4) The SD card after OS installation was inserted into the Raspberry Pi board.
5) The terminal like monitor, keyboard and mouse connected with Raspberry Pi.
6) The board with micro-USB connector was powered.
7) Once the power was switched on, the Raspberry Pi run on the OS installed in the memory card and started from boot.

8) Once all drivers were checked, the Raspberry Pi will ask for authorization, this was set by default and can be changed.
9) After authorization, a desktop popped up to start all application program development.

The application programs required can download on the Raspberry Pi and can directly install in the PC. Then, it is able to work on developing required program and let the Raspberry Pi operate the developed programs.

3.5.2 Telegram

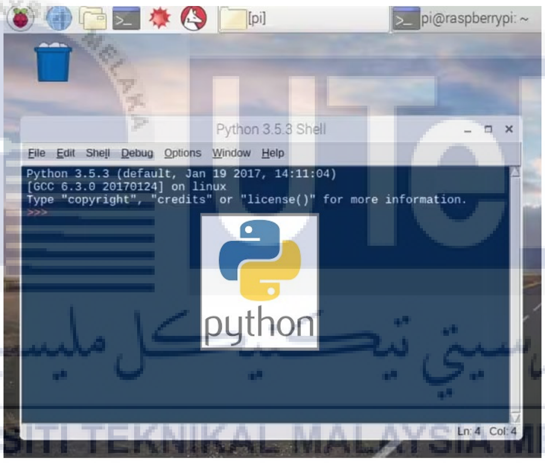
Telegram is a cloud-based mobile and desktop messaging app with a highlight on speed and security[24]. Although WhatsApp is the most popular app in term of messaging services, but Telegram offers cloud-based and heavily encrypted when compared to the WhatsApp app. It is a platform to access messages from different devices in the same time such as tablets and computers by sharing unlimited number of images, video and document. Telegram is free and stays free from the annoying advertisement and subscription fee. This home security system aimed to receive real time notification via smartphone or computer. A telegram bot is written to alert user when intrusion occurs. The homeowner is able to form a group in the Telegram application by inviting the telegram bot in the group. Thus, the other family members will the notification as well instead of the homeowner only.

Figure 3.13 Telegram Application[24]

3.5.3 Python

Python is an interpretive, high-level programming language that boasts remarkable readability and simplicity, which are pivotal for those who are just starting out with programming. However, it's not limited to beginners; Python's power, flexibility, and wide range of applications make it an invaluable tool for seasoned developers as well.

For our project, titled "Development of a Smart Mirror System Based on Raspberry Pi for Home Automation and Security (Google Assistance)", Python sits at the heart of our development strategy. With its extensive ecosystem of libraries, Python furnishes us with the essential tools required for various aspects of this innovative endeavour.



```
Python 3.5.3 (default, Jan 19 2017, 14:11:04)
[GCC 6.3.0 20170124] on linux
Type "copyright", "credits" or "license()" for more information.
>>>
```

Figure 3.14 Python Application[25]

To build an intuitive user interface for the smart mirror, we can make use of Python's GUI libraries like Tkinter or PyQt. These tools enable us to design an interface that's not only visually appealing but also simple for users to navigate and interact with[25].

When it comes to the security features of our system, including facial recognition and motion detection, Python's OpenCV library steps into the limelight. OpenCV supports us in carrying out complex image processing tasks which are integral for our system's security functionality.

Python's GPIO library will be of immense help in controlling the Raspberry Pi's GPIO pins, thereby playing a crucial role in hardware management.

For home automation features, we can write Python scripts that converse with smart devices using protocols like MQTT or HTTP. Python's prowess in handling networking protocols comes into play here, connecting various home devices seamlessly to our system. Integrating Google Assistant involves the Google Assistant SDK, which works beautifully with Python. This SDK allows our system to understand and respond to voice commands, enhancing the user experience.

As far as maintenance and updates are concerned, Python’s package management tools, such as pip, help keep the system up-to-date and secure.

Python’s robust support network, including comprehensive documentation and an active developer community, makes it an ideal choice for our project. It aids not only in the initial development phase but also ensures ongoing support and improvement of the smart mirror system.

3.5.4 Blynk

The Blynk project aims to showcase the integration of a microcontroller or development board with IoT capabilities, such as an Arduino or Raspberry Pi, and a connected LED. The purpose of this project is to provide a proof of concept for the communication between the IoT device and a central hub, typically through wireless protocols like Wi-Fi or Bluetooth.

The core idea behind the Blynk project is to establish a basic communication link between the IoT device and the central hub. The device is programmed to send a signal to the LED, instructing it to turn on for a specific duration and then turn off. This pattern of blinking is repeated continuously or for a predetermined number of cycles.

The Blynk project serves as an entry point for exploring more advanced IoT functionalities. It allows developers to verify the connectivity of their devices, ensure proper firmware or software configurations, and test the reliability of the communication protocols. By observing the LED blink, developers can gain confidence in the functioning of the IoT ecosystem they are building[26].

Figure 3.15 Blynk Application[26]

Moreover, Blynk can be expanded and customized to include additional features and functionalities. For example, developers can modify the blinking pattern to represent different states or conditions, such as different colours to indicate various events or sensor readings. This flexibility allows the Blynk project to be a starting point for more complex IoT applications, including home automation, environmental monitoring, or industrial control systems.

By starting with the Blynk project, developers can gradually build upon the foundational elements and extend their IoT solution to encompass a wide range of interconnected devices and sensors. This approach ensures a systematic and incremental development process, enabling the creation of robust and scalable IoT systems.

In conclusion, the Blynk project in an IoT context refers to a basic program that controls an LED to blink at regular intervals, serving as a starting point for testing and demonstrating the functionality of connected devices[26]. It represents the initial step in establishing communication and verifying the reliability of IoT devices and their integration with central hubs. The Blynk project can be customized and expanded to develop more complex IoT applications, providing a solid foundation for building innovative and interconnected systems.

3.6 Creating a custom casing project

3.6.1 Project prototype 3D printing

Designing in Tinkercad involves creating an account, starting a new design, and adding components for the ESP32, Raspberry Pi 4B, and ESP-CAM enclosures. You can customize the designs, export them as STL files, and visualize the results in Figure 3.15 and 3.16.

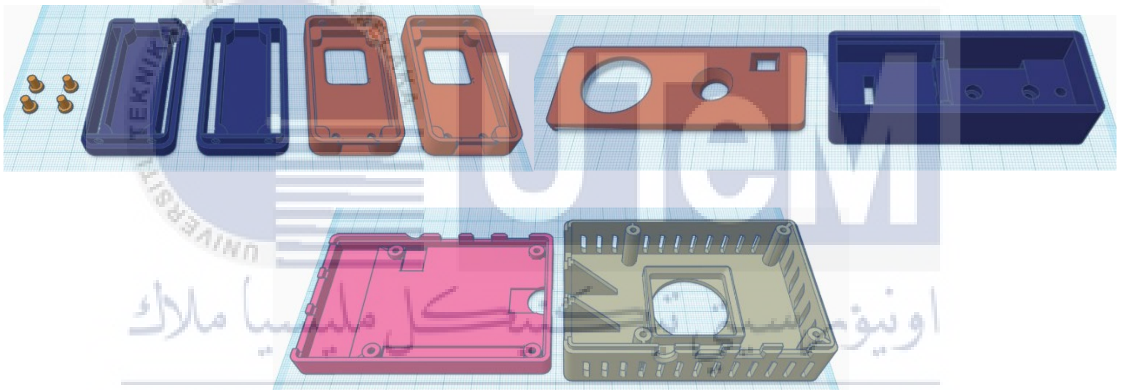


Figure 3.16 3D printing Model

Create an account on Tinkercad, visit the Tinkercad website and sign up or log in. Start a new design by clicking on "Create new design." Add components for the ESP32 design by dragging and dropping box components for the main body of the enclosure, adjusting dimensions, and adding openings for necessary features. Repeat this process for the Raspberry Pi 4B and ESP-CAM designs, adjusting dimensions and adding mounting features, wiring pathways, and openings for required components.

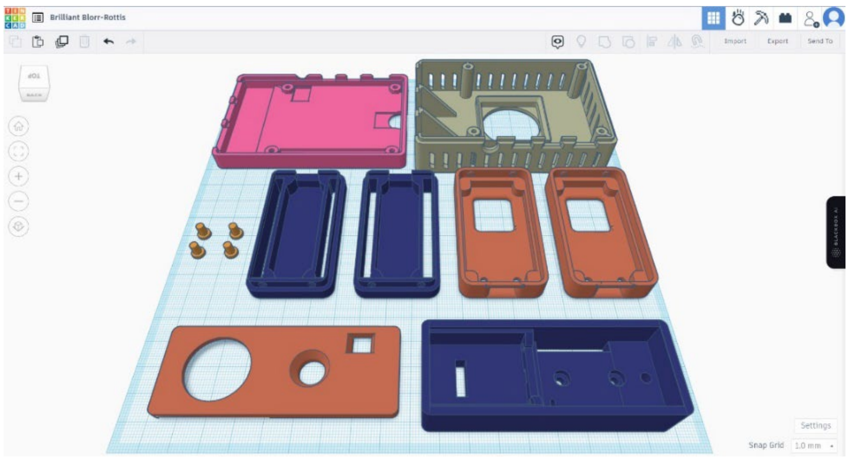


Figure 3.17 Printing Model Thinkercad

Customize the designs using the toolbar to add, resize, and customize shapes as needed. Combine shapes to create single enclosures or multiple parts, and group components together to simplify editing. Once satisfied with the design, click on "Design" and choose "Download for 3D Printing." Save the design as an STL file, which is compatible with most 3D printers.

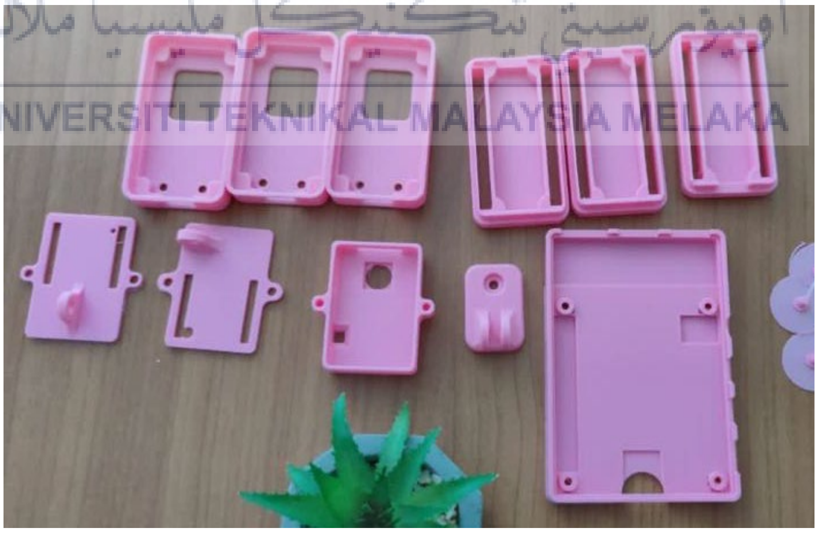


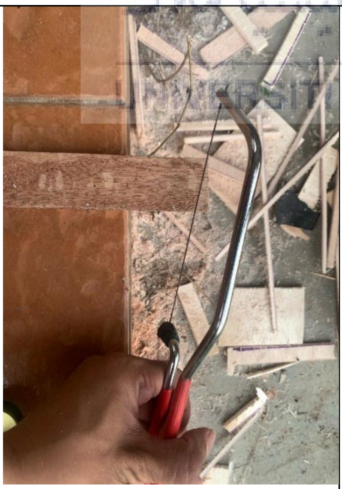
Figure 3.18 Design 3D after printing

Figure 3.16 (Design Stage) represents the design phase in Tinkercad, showing the arrangement of components for the ESP32, Raspberry Pi 4B, and ESP-CAM enclosures.

The figure should illustrate the placement of components, openings, and any unique features in the design. Figure 3.17 (3D Printed Result) illustrates the physical result after 3D printing the designs from Figure 3.16, showing the actual appearance of the printed enclosures for ESP32, Raspberry Pi 4B, and ESP-CAM. The printed result should reflect the accuracy of the design and the functionality of the enclosures.

3.6.2 Project prototype and connection

Designing a frame for an LCD screen is a great way to add a professional and polished look to your project. The process involves selecting the type of wood for the frame, measuring the dimensions of the LCD screen, cutting the wood boards to the desired lengths, sanding the edges, assembling the frame, and mounting the LCD screen. The materials required for this project include wood boards, measuring tape or ruler, pencil, miter saw or handsaw, sandpaper or a sanding block, wood glue, wood screws, screwdriver or drill, and spray paint.

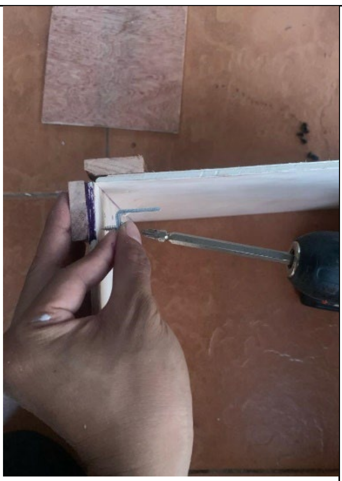
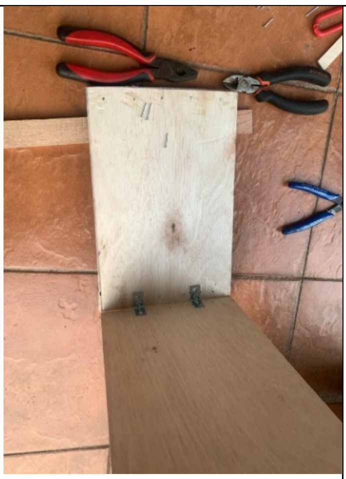
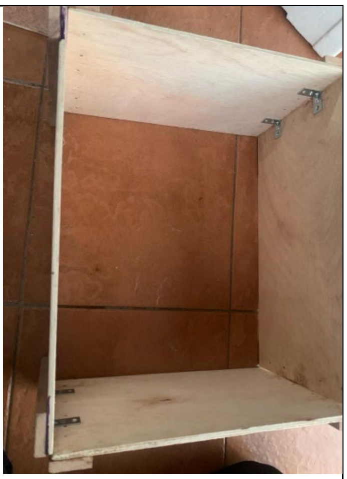
		

Figure 3.19 Designing Frame

The first step in designing a frame for an LCD screen is to measure the dimensions of the screen, including the width, height, and thickness. This will help determine the size of the wood boards needed for the frame. Next, select the type of wood for the frame. Common choices include pine, oak, or plywood. Purchase wood boards based on the dimensions of the LCD screen and your desired frame thickness. Once you have the wood boards, use a handsaw to cut the wood boards to the desired lengths. Cut four pieces to create the frame and make angle cuts at each end of the boards to create joints for a clean and professional look.

After cutting the wood boards, sand the cut edges of the wood to smooth any rough spots or splinters. This will ensure a polished finish. Assemble the frame by placing the cut pieces together in a dry fit. Check that the corners align correctly, and the frame fits the LCD screen. Apply wood glue to the edges of the frame using a small brush or your finger to spread the glue evenly. For added strength, you can reinforce the frame by drilling pilot holes and inserting wood screws through the corners. Be cautious not to split the wood. Allow the wood glue to fully dry according to the manufacturer's instructions. This typically takes several hours.

Once the frame is dry, sand the entire frame to smooth any imperfections. Apply spray paint if you want to enhance the appearance of the frame. Allow it to dry completely. Finally, place the LCD screen inside the frame and ensure a snug fit. The finished product will add a professional and polished look to your project.

In addition, Figure 3.18 is example results illustrates the step-by-step process of designing a wooden frame for an LCD screen. It guides users through the sequential steps required to create a professional-looking frame, showcasing the measurements, wood selection, cutting, sanding, assembly, and gluing stages. This visual representation provides a comprehensive overview of the DIY project, making it easier for users to understand and follow the outlined process.

On the other hand, Figure 4.18 in the search results showcases the result of the designed wooden frame for the LCD screen. It highlights the polished appearance, precise corners, and the snug fit around the LCD screen, serving as a visual representation of the successful completion of the DIY project. This figure emphasizes the professional and polished look achieved through the outlined process, providing users with a clear image of the final outcome.

3.7 Summary

In Chapter Three is dedicated to explaining in detail how we completed our project. This involves laying out the exact steps and designs we used to fulfil our project's goals. Imagine this chapter as a guidebook, providing a comprehensive view of the journey we embarked on to develop the project. It details every part of the process, from the initial planning stage right through to the final execution.

To further aid in this, we will use visual aids like a flowchart or block diagram. These are simple drawings that can help show how one step leads to another, and how all the different parts of the project are linked together. This visual guide will serve to make the complex process more digestible and straightforward.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 INTRODUCTION

This project focuses on developing a smart mirror system using Raspberry Pi to elevate home automation and security. The design and hardware implementation entail coding in Python for voice assistance, home automation, CCTV security, and display utilizing Thinkter. Additionally, the integration of ESP32 with Python through PyMkR is explored for home automation, while ESP32 Cam is linked to Telegram through Python for storing CCTV camera data. The project also incorporates face recognition using a webcam. The overall aim is to enhance home automation and security through the implementation of these interconnected technologies.

4.2 Project Prototype and Connection

A smart mirror system has been developed using a Raspberry Pi 4B as the main processor, an ESP Cam camera for monitoring, and two ESP 32 microcontrollers for home automation. The hardware setup is based on the schematic diagram in Figure 3.3, and the front view of the project model includes two ESP 32 microcontrollers, an ESP CAM camera, and a webcam as shown in Figure 4.1. The system allows for controlling home automation in two different places within the coverage range of the internet Wi-Fi connection. The ESP CAM camera captures images and video streaming in Blynk, and when motion is detected, it triggers and captures the image and send it to Telegram. The back view and front view of the project model, shown in Figure 4.3 and Figure 4.4, includes a connection USB microphone device, speaker, and webcam, which function to record audio and output audio for voice assistant when the program is running in Raspberry Pi 4B. The microphone and the speaker are located inside the frame model. The easiest way to power the Raspberry Pi and the display is by connecting this port with a power supply.

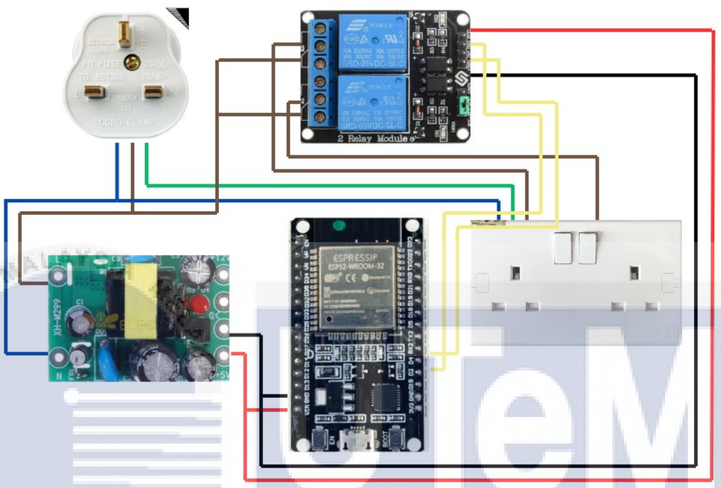


Figure 4.1 ESP 32 Microcontroller Home Automation Circuit

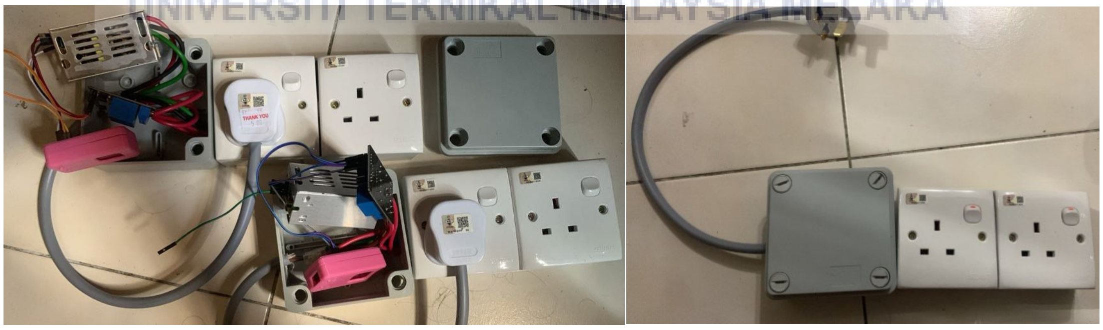


Figure 4.2 ESP 32 Microcontroller Home Automation Prototype

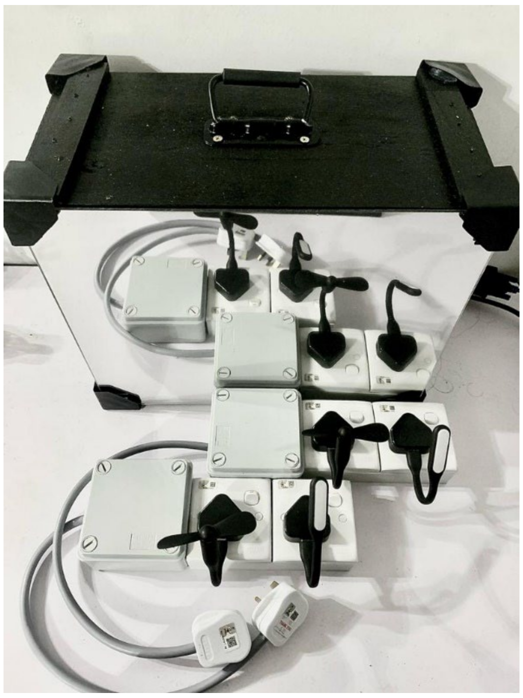


Figure 4.3 Front View of Project Prototype

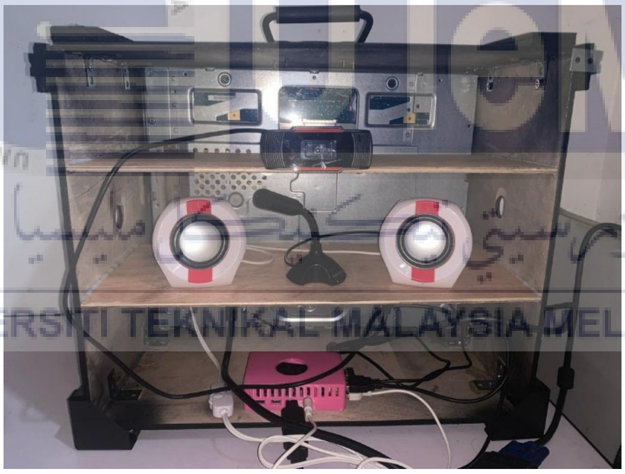


Figure 4.4 Back View of Project Prototype

4.3 Raspberry Pi 4B Setup (display)

Setting up a Raspberry Pi for its first boot and remotely accessing it using Fing to obtain its IP address and VNC to establish a connection. Here's a step-by-step guide: Before starting, ensure have the following:

1) Raspberry Pi (with Raspbian OS pre-loaded on an SD card).

2) A device to control Raspberry Pi (e.g., a laptop or smartphone).

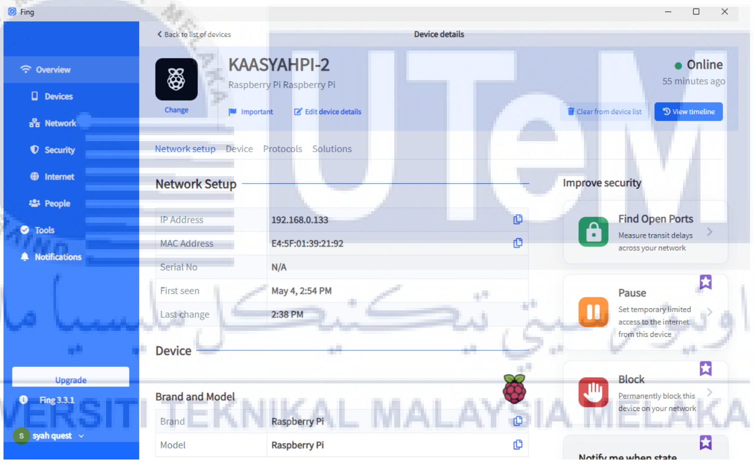
3) Power supply for Raspberry Pi.

4) Fing app installed on device.

5) VNC Viewer installed on device.

First, power up the Raspberry Pi by connecting it to a power source. It will automatically boot up once powered.

Next, connect Raspberry Pi to the local network. Connecting an Ethernet cable from router to the Raspberry Pi or by using a monitor, keyboard, and mouse to connect it to a Wi-Fi network.



KAASYAHPI-2
Raspberry Pi Raspberry Pi
Online
55 minutes ago

Network Setup

IP Address	192.168.0.133
MAC Address	E4:5F:01:39:21:92
Serial No	N/A
First seen	May 4, 2:54 PM
Last change	2:38 PM

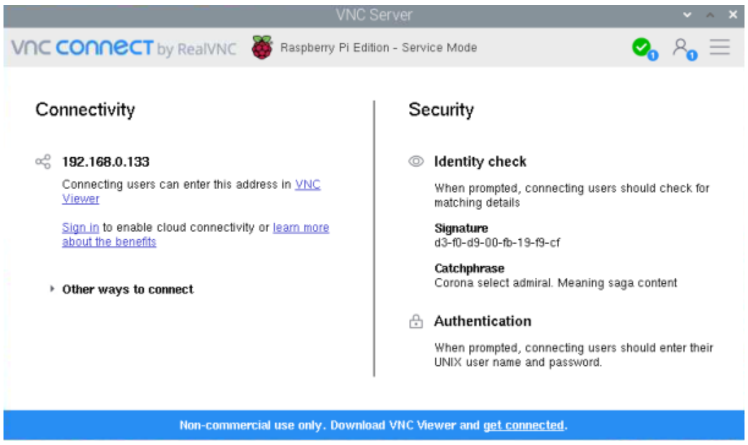
Device

Brand and Model

Brand	Raspberry Pi
Model	Raspberry Pi

Figure 4.5 Fing app (Ip address finder)

The IP address of Raspberry Pi, ensure controlling device (laptop, smartphone, etc.) is connected to the same network. Open the Fing app and perform a network scan. Look for the Raspberry Pi device in the scan results, which should be listed as 'Raspberry Pi'. Make note of the IP address associated with Raspberry Pi.



Connectivity

192.168.0.133

Connecting users can enter this address in VNC Viewer

Sign in to enable cloud connectivity or learn more about the benefits

Other ways to connect

Security

Identity check

When prompted, connecting users should check for matching details

Signature
d3-f0-d9-00-fb-19-f9-cf

Catchphrase
Corona select admiral. Meaning saga content

Authentication

When prompted, connecting users should enter their UNIX user name and password.

Non-commercial use only. Download VNC Viewer and get connected.

Figure 4.6 VNC Ip address raspberry pi.

Enable VNC on the Raspberry Pi. If have access to a monitor and keyboard, open the terminal on the Raspberry Pi and enter 'sudo raspi-config' to launch the configuration tool. Navigate to 'Interfacing Options' > 'VNC' and select 'Yes' to enable VNC.

If using SSH to connect to the Raspberry Pi, SSH into it using the IP address. Once connected, enter 'sudo raspi-config' to open the configuration tool. Navigate to 'Interfacing Options' > 'VNC' and select 'Yes' to enable VNC.

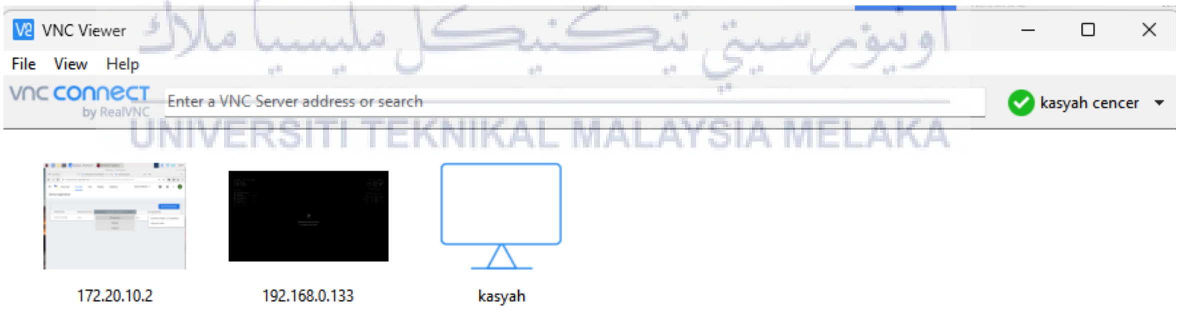
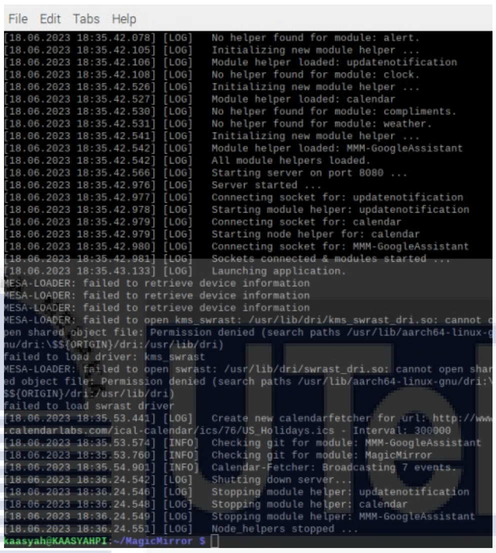


Figure 4.7 VNC viewer

Connect to Raspberry Pi using VNC Viewer, open the VNC Viewer app on device. Click on 'File' > 'New Connection' and enter the IP address of Raspberry Pi in the 'VNC Server' field. Click 'OK' and then double-click on the created connection. Enter the username and password for Raspberry Pi (the default username is 'pi' and the default password is 'raspberry'). Click 'OK' or 'Connect'. remotely control Raspberry Pi using VNC.

Install MagicMirror2 on Raspberry Pi, start by connecting a USB keyboard and a display screen to Raspberry Pi using an HDMI cable. Ensure that Raspberry Pi is connected to the internet via Wi-Fi or an Ethernet cable. Open the programming terminal on Raspberry Pi interface.



```
[18.06.2023 18:35.42.078] [LOG]   No helper found for module: alert.
[18.06.2023 18:35.42.105] [LOG]   Initializing new module helper ...
[18.06.2023 18:35.42.106] [LOG]   Module helper loaded: updatenotification
[18.06.2023 18:35.42.108] [LOG]   No helper found for module: clock.
[18.06.2023 18:35.42.526] [LOG]   Initializing new module helper ...
[18.06.2023 18:35.42.527] [LOG]   Module helper loaded: calendar
[18.06.2023 18:35.42.530] [LOG]   No helper found for module: compliments.
[18.06.2023 18:35.42.531] [LOG]   No helper found for module: weather.
[18.06.2023 18:35.42.541] [LOG]   Initializing new module helper ...
[18.06.2023 18:35.42.542] [LOG]   Module helper loaded: MMM-GoogleAssistant
[18.06.2023 18:35.42.542] [LOG]   All module helpers loaded.
[18.06.2023 18:35.42.566] [LOG]   Starting server on port 8080 ...
[18.06.2023 18:35.42.976] [LOG]   Server started ...
[18.06.2023 18:35.42.977] [LOG]   Connecting socket for: updatenotification
[18.06.2023 18:35.42.978] [LOG]   Starting module helper: updatenotification
[18.06.2023 18:35.42.979] [LOG]   Connecting socket for: calendar
[18.06.2023 18:35.42.979] [LOG]   Starting node helper for: calendar
[18.06.2023 18:35.42.980] [LOG]   Connecting socket for: MMM-GoogleAssistant
[18.06.2023 18:35.42.981] [LOG]   Sockets connected & modules started ...
[18.06.2023 18:35.43.133] [LOG]   Launching application.
MESA-LOADER: failed to retrieve device information
MESA-LOADER: failed to retrieve device information
MESA-LOADER: failed to retrieve device information
MESA-LOADER: failed to open kms_swrast: /usr/lib/dri/kms_swrast_dri.so: cannot o
pen shared object file: Permission denied (search paths /usr/lib/aarch64-linux-g
nu/dri:\$${ORIGIN}/dri:/usr/lib/dri)
failed to load driver: kms_swrast
MESA-LOADER: failed to open swrast: /usr/lib/dri/swrast_dri.so: cannot open shar
ed object file: Permission denied (search paths /usr/lib/aarch64-linux-gnu/dri:\
$${ORIGIN}/dri:/usr/lib/dri)
failed to load swrast driver
[18.06.2023 18:35.53.441] [LOG]   Create new calendarfetcher for url: http://www
.calendarlabs.com/ical-calendar/ics/76/US_Holidays.ics - Interval: 300000
[18.06.2023 18:35.53.574] [INFO]  Checking git for module: MMM-GoogleAssistant
[18.06.2023 18:35.53.760] [INFO]  Checking git for module: MagicMirror
[18.06.2023 18:35.54.901] [INFO]  Calendar-Fetcher: Broadcasting 7 events.
[18.06.2023 18:36.24.542] [LOG]   Shutting down server...
[18.06.2023 18:36.24.546] [LOG]   Stopping module helper: updatenotification
[18.06.2023 18:36.24.548] [LOG]   Stopping module helper: calendar
[18.06.2023 18:36.24.549] [LOG]   Stopping module helper: MMM-GoogleAssistant
[18.06.2023 18:36.24.551] [LOG]   Node_helpers stopped ...
kaasyah@KAASYAHPI:~/MagicMirror $ 
```

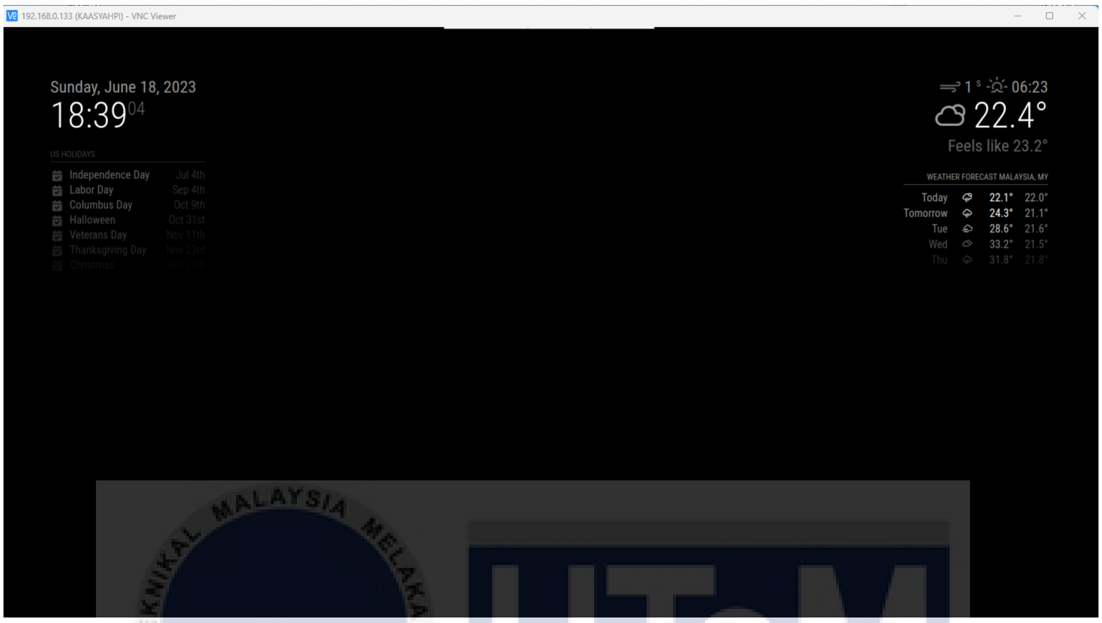
Figure 4.8 Command prompt raspberry pi

In the terminal, enter the following command to download Node.js, a programming environment for the Pi: "curl -sL https://deb.nodesource.com/setup_14.x | sudo -E bash - " and "sudo apt install -y nodejs". This will download Node.js successfully onto Raspberry Pi.

Next, clone the MagicMirror2 repository from GitHub using the command: "git clone https://github.com/MichMich/MagicMirror". This will create a local copy of the MagicMirror2 code on Raspberry Pi.

Navigate to the MagicMirror directory by running the command: "cd MagicMirror/". Once inside the directory, install the MagicMirror application by running "npm install". This will install the required dependencies for MagicMirror2.

To customize MagicMirror configuration, copy the sample configuration file using the command: "cp config/config.js.sample config/config.js". This will create a new configuration file that can modify to suit needs.



Sunday, June 18, 2023
18:39 04

US HOLIDAYS

Independence Day	Jul 4th
Labor Day	Sep 4th
Columbus Day	Oct 9th
Halloween	Oct 31st
Veterans Day	Nov 11th
Thanksgiving Day	Nov 23rd
Christmas	Dec 25th

1 S 06:23
22.4°
Feels like 23.2°

WEATHER FORECAST MALAYSIA, MY

Today	22.1°	22.0°
Tomorrow	24.3°	21.1°
Tue	28.6°	21.6°
Wed	33.2°	21.5°
Thu	31.8°	21.8°

Figure 4.9 Interface of Smart Mirror

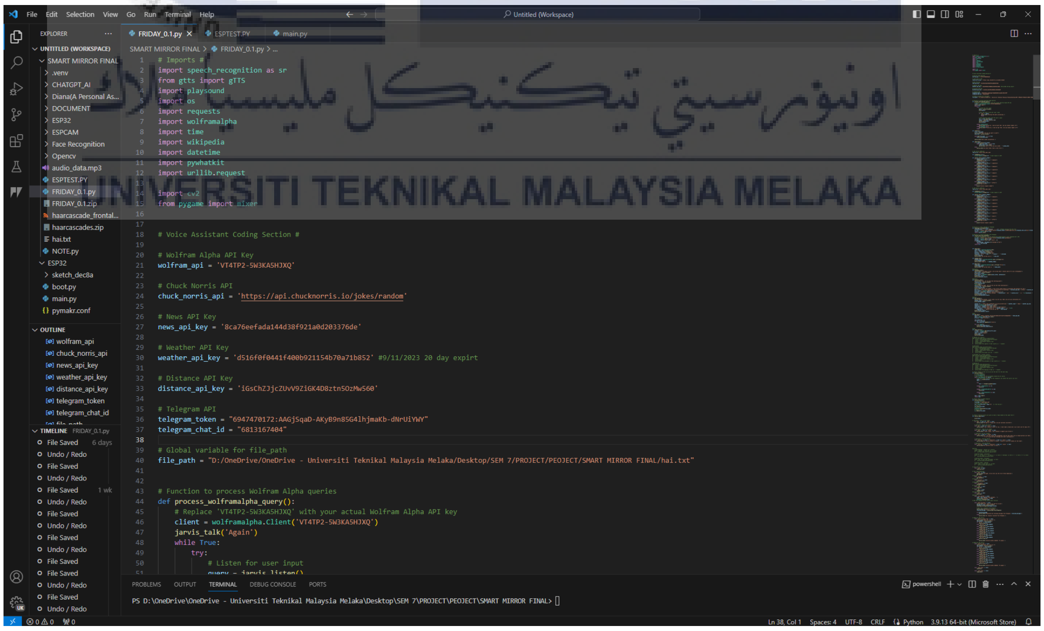
Finally, start the MagicMirror2 application by running the command: "npm run start". This will launch the MagicMirror interface on display screen.

4.4 Raspberry Pi 4B Setup (voice assistance)

The provided Python script is a comprehensive implementation of a voice-controlled assistant named "FRIDAY." The code begins by importing several libraries and APIs, including speech recognition, text-to-speech, Wolfram Alpha, Wikipedia, HTTP requests, computer vision, and sound mixing. The script initializes various API keys and URLs required for services like Wolfram Alpha, Chuck Norris jokes, News, Weather, and Distance calculation. It also sets up the Telegram API credentials and a global variable for the file path used in the note-taking feature.

The script defines several functions to perform specific tasks, such as processing user queries using the Wolfram Alpha API, saving and reading notes, controlling LEDs in different rooms using ESP32 microcontrollers, interacting with the Telegram API, providing time, date, weather, and news information, fetching Chuck Norris jokes, retrieving Wikipedia information, calculating distances, and handling speech-to-text and text-to-speech conversion. The "execute_assistant" function initiates the assistant, greets the user, listens for commands, and responds accordingly.

The code is well-structured and uses functions to modularize different functionalities. It demonstrates the integration of various APIs for Wolfram Alpha, Chuck Norris jokes, News, Weather, and more. The voice-controlled assistant interacts with the user through speech recognition and synthesis. The script effectively combines several APIs, hardware control (ESP32), and natural language processing to create a comprehensive voice-controlled assistant with a variety of functionalities.



```python
# Imports #
import speech_recognition as sr
from gtts import gTTS
import playsound
import os
import requests
import wolframalpha
import time
import wikipedia
import datetime
import pywhatkit
import urllib.request

import cv2
from pygame import mixer


# Voice Assistant Coding Section #

# Wolfram Alpha API Key
wolfram_api = 'VT4TP2-5W3KA5HJXQ'

# Chuck Norris API
chuck_norris_api = 'https://api.chucknorris.io/jokes/random'

# News API Key
news_api_key = '8ca76eefada144d38f921a0d203376de'

# Weather API Key
weather_api_key = 'd516f0f0441f400b921154b70a71b852' #9/11/2023 20 day expirt

# Distance API Key
distance_api_key = 'iGsChZJjcZUvV9ZiGK4D8ztn5OzMw560'

# Telegram API
telegram_token = "6947470172:AAGjSqaD-AKyB9n85G4lhjmaKb-dNrUiYWY"
telegram_chat_id = "6813167404"

# Global variable for file_path
file_path = "D:/OneDrive/OneDrive - Universiti Teknikal Malaysia Melaka/Desktop/SEM 7/PROJECT/PEOJECT/SMART MIRROR FINAL/hai.txt"


# Function to process Wolfram Alpha queries
def process_wolframalpha_query():
    # Replace 'VT4TP2-5W3KA5HJXQ' with your actual Wolfram Alpha API key
    client = wolframalpha.Client('VT4TP2-5W3KA5HJXQ')
    jarvis_talk('Again')
    while True:
        try:
            # Listen for user input
            query = jarvis.listen()
```

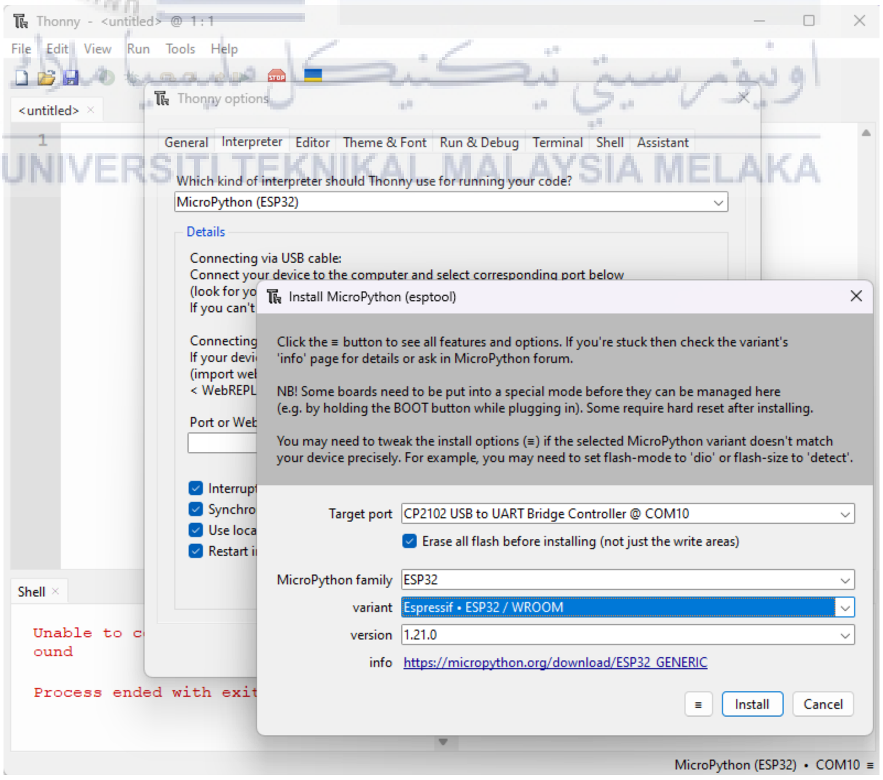
```
PS D:\OneDrive\OneDrive - Universiti Teknikal Malaysia Melaka\Desktop\SEM 7\PROJECT\PEOJECT\SMART MIRROR FINAL>
```

Figure 4.10 VSCODE Voice Assistance Code

Figure 4.8 is likely a reference to an example code snippet. Unfortunately, the code snippet itself is not provided, but it might represent a visual representation of the coding structure or a specific implementation detail. In conclusion, the code showcases the development of a voice-controlled assistant with diverse functionalities, integrating APIs and libraries for a comprehensive user experience. It demonstrates the capabilities of handling natural language queries, controlling devices, and fetching information from external sources.

4.5 Raspberry Pi 4B Setup (home assistance)

PyMakr in VSCode, the first is to flash the ESP32 board with the MicroPython firmware, which is typically done using a tool like Thonny. The MicroPython firmware allows the ESP32 to run Python scripts. Once the ESP32 is flashed, PyMakr can be used to interact with the board in VSCode. PyMakr provides a REPL (Read-Eval-Print Loop) console for running MicroPython code interactively, and it allows uploading and managing files on the ESP32.


Thonny - <untitled> @ 1 : 1
File Edit View Run Tools Help
<untitled>
Thonny options
General Interpreter Editor Theme & Font Run & Debug Terminal Shell Assistant
Shell
MicroPython (ESP32) • COM10 ≡

Which kind of interpreter should Thonny use for running your code?
MicroPython (ESP32)

Details
Connecting via USB cable:
Connect your device to the computer and select corresponding port below
(look for yo
If you can't

Connecting
If your devi
(import wel
< WebREPL

Port or Web

Interrupt
Synchro
Use loca
Restart i

```
Unable to c
ound

Process ended with exit
```

Install MicroPython (esptool)

Click the ≡ button to see all features and options. If you're stuck then check the variant's 'info' page for details or ask in MicroPython forum.

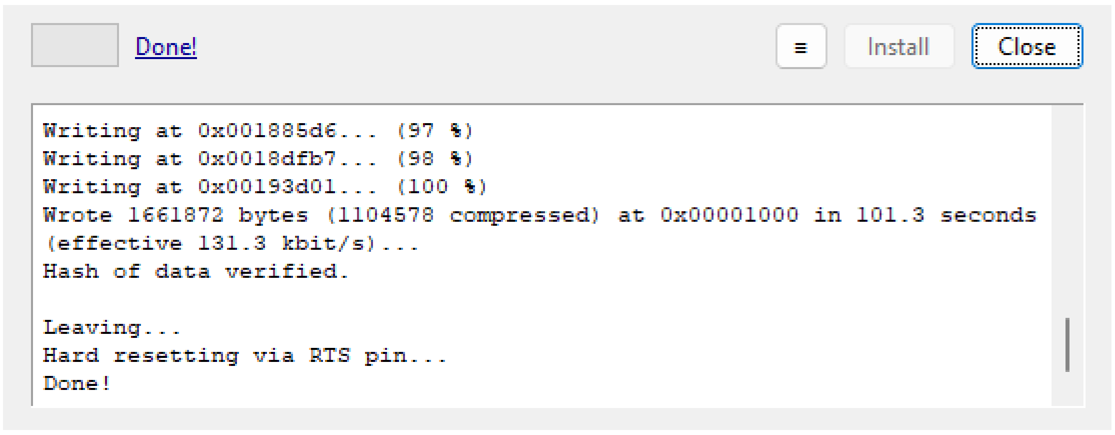
NB! Some boards need to be put into a special mode before they can be managed here (e.g. by holding the BOOT button while plugging in). Some require hard reset after installing.

You may need to tweak the install options (≡) if the selected MicroPython variant doesn't match your device precisely. For example, you may need to set flash-mode to 'dio' or flash-size to 'detect'.

Target port: CP2102 USB to UART Bridge Controller @ COM10
✓ Erase all flash before installing (not just the write areas)
MicroPython family: ESP32
variant: Espressif • ESP32 / WROOM
version: 1.21.0
info: https://micropython.org/download/ESP32_GENERIC

≡ Install Cancel

Figure 4.11 Thonny (Flashing ESP 32 Micropython)

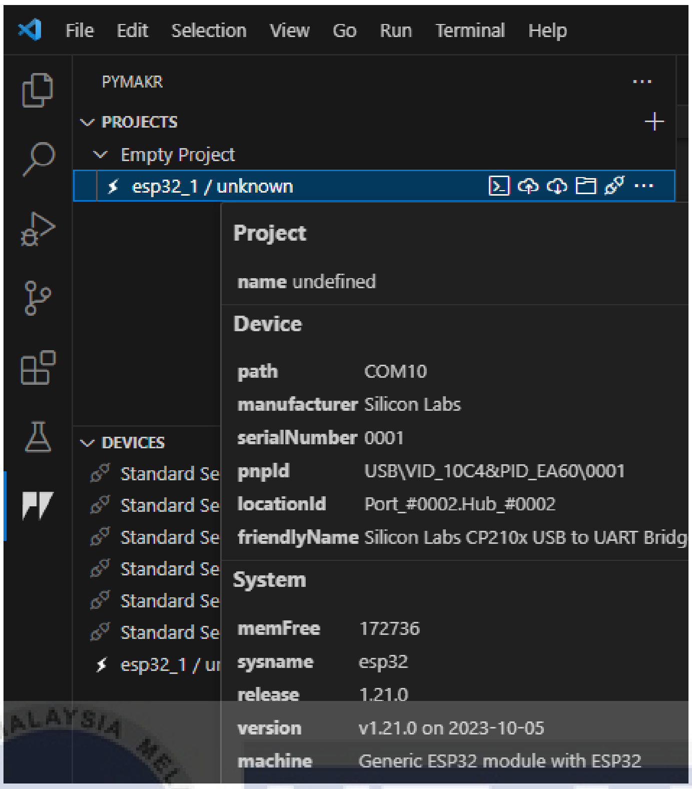


```
Writing at 0x001885d6... (97 %)
Writing at 0x0018dfb7... (98 %)
Writing at 0x00193d01... (100 %)
Wrote 1661872 bytes (1104578 compressed) at 0x00001000 in 101.3 seconds
(effective 131.3 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done!
```

Figure 4.12 Uploading and Verification Flashing ESP 32

The provided code is a MicroPython script designed to run on an ESP32 board, creating a simple web server that controls three LEDs connected to GPIO pins on the ESP32. The code first connects the ESP32 to a Wi-Fi network, sets up a socket server to listen for incoming connections, and then handles LED control based on incoming HTTP requests. After flashing the ESP32 with the MicroPython firmware, PyMakr can be used in VSCode to interact with the ESP32, providing a REPL console for running MicroPython code interactively and allowing file management on the ESP32.



Project

name undefined

Device

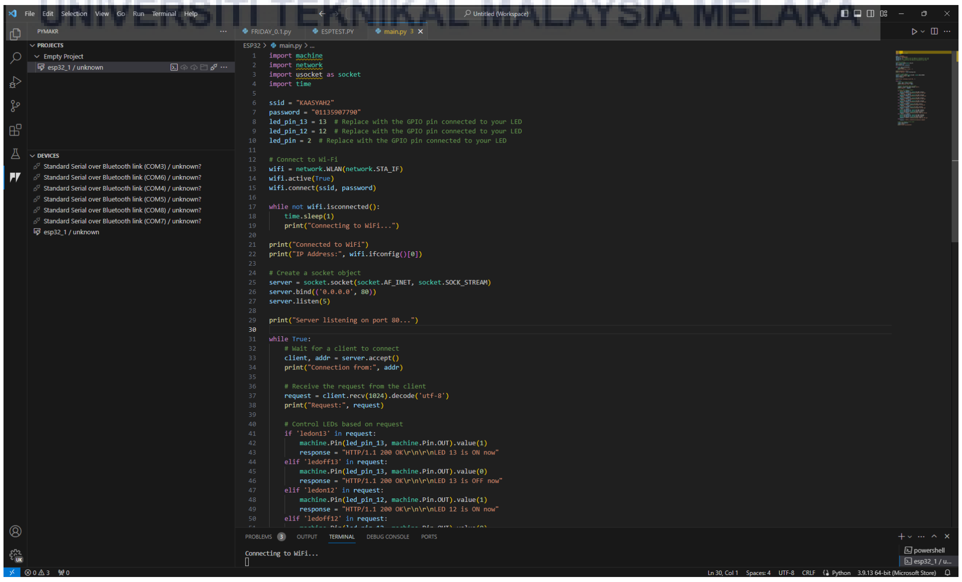
path	COM10
manufacturer	Silicon Labs
serialNumber	0001
pnpId	USB\VID_10C4&PID_EA60\0001
locationId	Port_#0002.Hub_#0002
friendlyName	Silicon Labs CP210x USB to UART Bridg

System

memFree	172736
sysname	esp32
release	1.21.0
version	v1.21.0 on 2023-10-05
machine	Generic ESP32 module with ESP32

Figure 4.13 Identify port for PYMAKR.

Figure 4.12 likely represents a screenshot or an example code snippet in VSCode. Unfortunately, the figure is not provided, but it might show the ESP32 connected, the REPL console, or the code editor in VSCode with the MicroPython script loaded.



```python
import machine
import network
import usocket as socket
import time

ssid = "KAASYAH2"
password = "01135907790"
led_pin_13 = 13  # Replace with the GPIO pin connected to your LED
led_pin_12 = 12  # Replace with the GPIO pin connected to your LED
led_pin = 2  # Replace with the GPIO pin connected to your LED

# Connect to Wi-Fi
wifi = network.WLAN(network.STA_IF)
wifi.active(True)
wifi.connect(ssid, password)

while not wifi.isconnected():
    time.sleep(1)
    print("Connecting to WiFi...")

print("Connected to WiFi")
print("IP Address:", wifi.ifconfig()[0])

# Create a socket object
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(('0.0.0.0', 80))
server.listen(5)

print("Server listening on port 80...")

while True:
    # Wait for a client to connect
    client, addr = server.accept()
    print("Connection from:", addr)

    # Receive the request from the client
    request = client.recv(1024).decode('utf-8')
    print("Request:", request)

    # Control LEDs based on request
    if 'ledon13' in request:
        machine.Pin(led_pin_13, machine.Pin.OUT).value(1)
        response = "HTTP/1.1 200 OK\r\n\r\nLED 13 is ON now"
    elif 'ledoff13' in request:
        machine.Pin(led_pin_13, machine.Pin.OUT).value(0)
        response = "HTTP/1.1 200 OK\r\n\r\nLED 13 is OFF now"
    elif 'ledon12' in request:
        machine.Pin(led_pin_12, machine.Pin.OUT).value(1)
        response = "HTTP/1.1 200 OK\r\n\r\nLED 12 is ON now"
    elif 'ledoff12' in request:
```

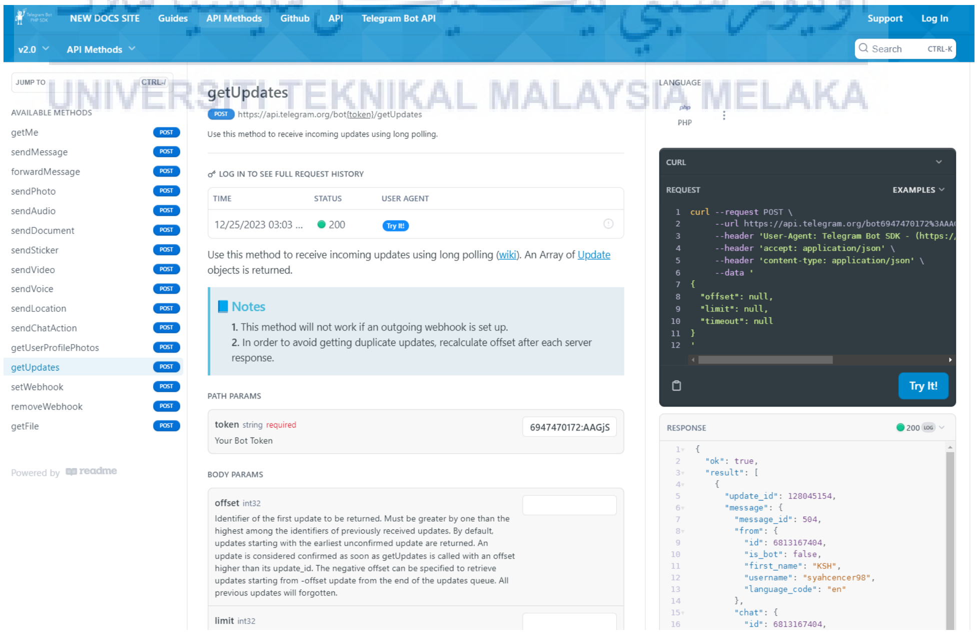
```
Connecting to WiFi...
```

Figure 4.14 Code Home Automation ESP32

The code demonstrates the implementation of a simple web server on an ESP32, allowing control of LEDs through HTTP requests. It leverages MicroPython to provide a lightweight and efficient solution for IoT applications. The use of PyMakr in VSCode facilitates the development, testing, and debugging of MicroPython code on the ESP32.

4.6 Feature of CCTV Notification System

The provided Arduino code is a comprehensive program designed for an ESP32-CAM module that offers a wide range of functionalities. The code establishes a connection to a Wi-Fi network using the ESP32 module, initializes the OV2640 camera, and sets up a web server to handle HTTP requests. The camera is configured with various settings, including frame size, JPEG quality, and other parameters, using the configInitCamera() function. The code also supports MJPEG streaming for video functionality, which can be accessed by navigating to the specified URL.



getUpdates

POST https://api.telegram.org/bot{token}/getUpdates

Use this method to receive incoming updates using long polling.

LOG IN TO SEE FULL REQUEST HISTORY

TIME	STATUS	USER AGENT
12/25/2023 03:03 ...	● 200	Try It!

Use this method to receive incoming updates using long polling (wiki). An Array of Update objects is returned.

Notes
1. This method will not work if an outgoing webhook is set up.
2. In order to avoid getting duplicate updates, recalculate offset after each server response.

PATH PARAMS

token string required
Your Bot Token
6947470172:AAGjS

BODY PARAMS

offset int32
Identifier of the first update to be returned. Must be greater by one than the highest among the identifiers of previously received updates. By default, updates starting with the earliest unconfirmed update are returned. An update is considered confirmed as soon as getUpdates is called with an offset higher than its update_id. The negative offset can be specified to retrieve updates starting from -offset update from the end of the updates queue. All previous updates will forgotten.

limit int32

CURL

REQUEST

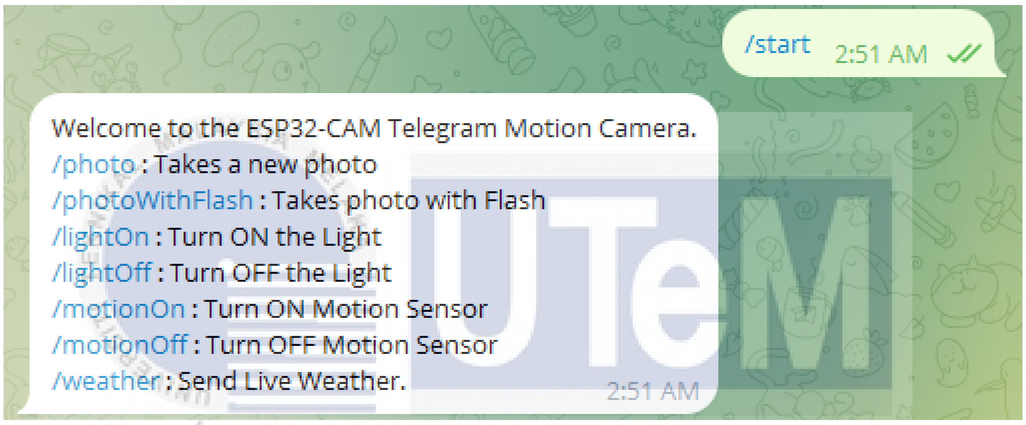
```
curl --request POST \
     --url https://api.telegram.org/bot6947470172%3AAA
     --header 'User-Agent: Telegram Bot SDK - (https:/
     --header 'accept: application/json' \
     --header 'content-type: application/json' \
     --data '
{
  "offset": null,
  "limit": null,
  "timeout": null
}
'
```

RESPONSE ● 200

```
{
  "ok": true,
  "result": [
    {
      "update_id": 128045154,
      "message": {
        "message_id": 504,
        "from": {
          "id": 6813167404,
          "is_bot": false,
          "first_name": "KSH",
          "username": "syahcencer98",
          "language_code": "en"
        },
        "chat": {
          "id": 6813167404,
```

Figure 4.15 Identify telegram id and token API Telegram

One of the most notable features of the code is its integration with a Telegram bot, which allows users to interact with the ESP32-CAM module remotely. Users can send commands via Telegram to trigger actions like capturing photos, turning on/off lights, and retrieving weather information. The sendPhotoTelegram() function captures a photo using the camera and sends it to the specified Telegram chat using the Telegram API. The handleNewMessages() function processes incoming Telegram messages, interprets user commands, and triggers corresponding actions.

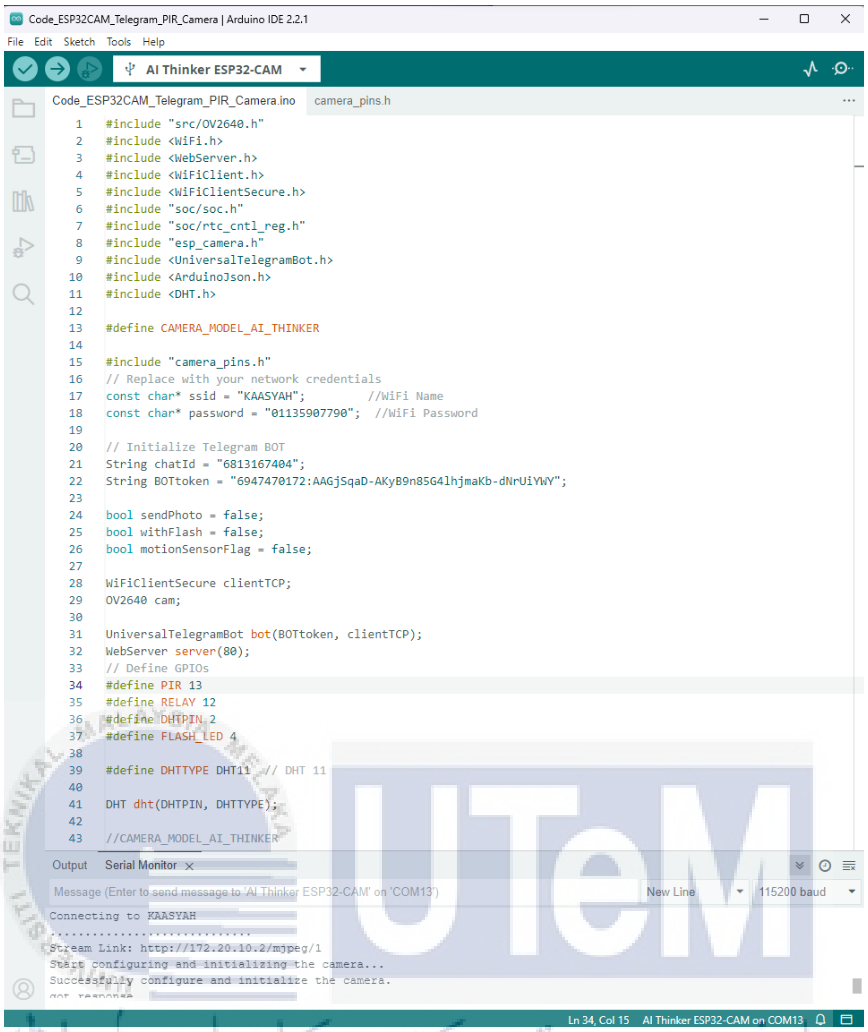


```
/start 2:51 AM

Welcome to the ESP32-CAM Telegram Motion Camera.
/photo : Takes a new photo
/photoWithFlash : Takes photo with Flash
/lightOn : Turn ON the Light
/lightOff : Turn OFF the Light
/motionOn : Turn ON Motion Sensor
/motionOff : Turn OFF Motion Sensor
/weather : Send Live Weather.
2:51 AM
```

Figure 4.16 Generates controlling system in telegram.

The code also includes functionality for motion sensor handling using a Passive Infrared Sensor (PIR). The code monitors the PIR and captures a photo when motion is detected if the motion sensor functionality is enabled. Additionally, the getWeather() function uses a DHT11 sensor to read temperature and humidity data, which is then sent back to the user upon request.The code also includes functions like lightOn() and lightOff() that enable users to remotely control a relay-connected light. These functions allow users to turn the light on or off based on their commands.


Code_ESP32CAM_Telegram_PIR_Camera | Arduino IDE 2.2.1
File Edit Sketch Tools Help
AI Thinker ESP32-CAM
Code_ESP32CAM_Telegram_PIR_Camera.ino
camera_pins.h
Output
Serial Monitor
Message (Enter to send message to 'AI Thinker ESP32-CAM' on 'COM13')
Ln 34, Col 15
AI Thinker ESP32-CAM on COM13

```
#include "src/OV2640.h"
#include <WiFi.h>
#include <WebServer.h>
#include <WiFiClient.h>
#include <WiFiClientSecure.h>
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"
#include "esp_camera.h"
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>
#include <DHT.h>

#define CAMERA_MODEL_AI_THINKER

#include "camera_pins.h"
// Replace with your network credentials
const char* ssid = "KAASYAH";          //WiFi Name
const char* password = "01135907790";  //WiFi Password

// Initialize Telegram BOT
String chatId = "6813167404";
String BOTtoken = "6947470172:AAGjSqaD-AKyB9n85G4lhjmaKb-dNrUiYWY";

bool sendPhoto = false;
bool withFlash = false;
bool motionSensorFlag = false;

WiFiClientSecure clientTCP;
OV2640 cam;

UniversalTelegramBot bot(BOTtoken, clientTCP);
WebServer server(80);
// Define GPIOs
#define PIR 13
#define RELAY 12
#define DHTPIN 2
#define FLASH_LED 4

#define DHTTYPE DHT11  // DHT 11

DHT dht(DHTPIN, DHTTYPE);

//CAMERA_MODEL_AI_THINKER
```

New Line   115200 baud

```
Connecting to KAASYAH
.............................
Stream Link: http://172.20.10.2/mjpeg/1
Start configuring and initializing the camera...
Successfully configure and initialize the camera.
got response
```

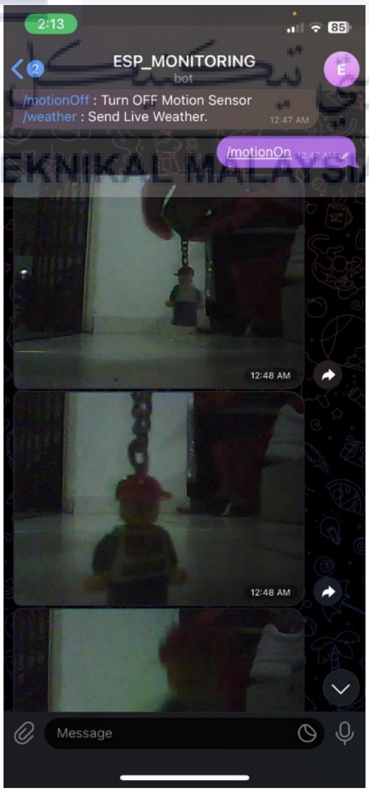
Figure 4.17 Code ESP32-CAM

The figure 4.8 in the search results exemplifies a typical ESP32-CAM program with Telegram integration, camera setup, motion sensing, and additional features. This figure showcases the versatility of ESP32-CAM for remote monitoring and control applications. Users can interact with the ESP32-CAM module in real-time, making it suitable for applications like surveillance, home automation, and environmental monitoring.

In conclusion, the provided Arduino code for ESP32-CAM module offers a wide range of functionalities, including image capture, video streaming, and remote control through Telegram. The code combines several features to create a versatile IoT device capable of performing various actions like capturing and sending photos, controlling lights, monitoring motion sensors, and fetching live weather data. The code's integration with Telegram bot makes it easy for users to interact with the ESP32-CAM module remotely, making it suitable for various applications like surveillance, home automation, and environmental monitoring.

4.6.1 Image Capturing

The image capture functionality in the provided Arduino code for the ESP32-CAM module involves capturing a photo using the camera module and sending the captured image to a specified Telegram chat using the Telegram Bot API. This process is facilitated by the sendPhotoTelegram() function, which is triggered by specific Telegram commands such as /photo and /photoWithFlash. Upon receiving these commands, the ESP32-CAM captures a photo and interacts with the Telegram API to send the image to the specified chat, providing feedback to the user regarding the success or failure of the operation.



/motionOff : Turn OFF Motion Sensor
/weather : Send Live Weather. 12:47 AM

/motionOn 12:47 AM

Figure 4.18 Receiving image from ESP32-CAM

The integration with Telegram allows for remote control and monitoring of the ESP32-CAM's camera capabilities, enabling users to trigger photo capture and receive real-time feedback on the status of the image transmission. Additionally, the optional /photoWithFlash command includes the use of a flash LED during photo capture, providing users with the flexibility to adjust the photo capture process based on their specific requirements. This seamless integration with Telegram enhances the versatility of the ESP32-CAM module, making it suitable for various applications such as surveillance, home automation, and environmental monitoring.

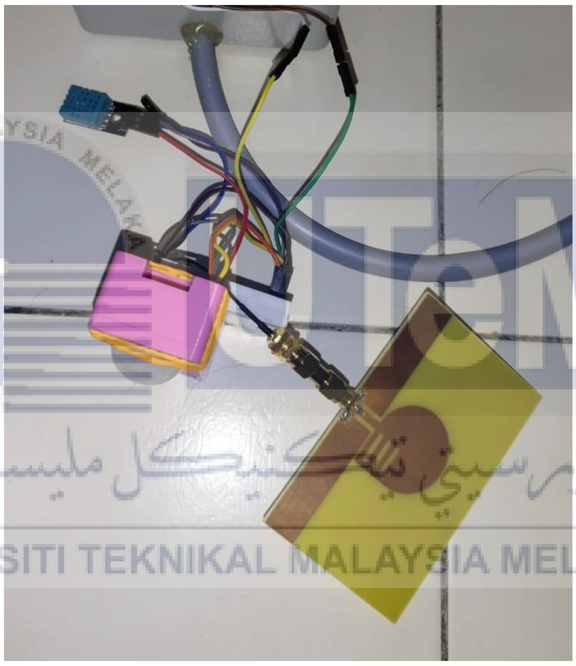


Figure 4.19 Connection Esp32, Temperature, Pir and Antenna

In summary, the image capture functionality in the provided Arduino code for the ESP32-CAM module offers a user-friendly and interactive way to capture and transmit images, leveraging the capabilities of the Telegram Bot API for seamless remote control and monitoring. This feature, combined with the optional flash usage, enhances the adaptability of the ESP32-CAM for diverse use cases, ranging from basic photo capture to more advanced applications that require real-time image transmission and user interaction.

4.6.2 Video Streaming

The MJPEG streaming functionality in the provided Arduino code for the ESP32-CAM module enables continuous video streaming from the camera module. The streaming involves sending a series of JPEG images in a stream to create a video feed. The code sets up an HTTP server to handle MJPEG streaming requests at the /mjpeg/1 endpoint, and the handle_jpg_stream() function is responsible for handling MJPEG streaming. The integration with the Blynk IoT platform allows users to incorporate the video feed into a customized IoT application with other Blynk widgets, making it suitable for surveillance, monitoring, or remote viewing applications.

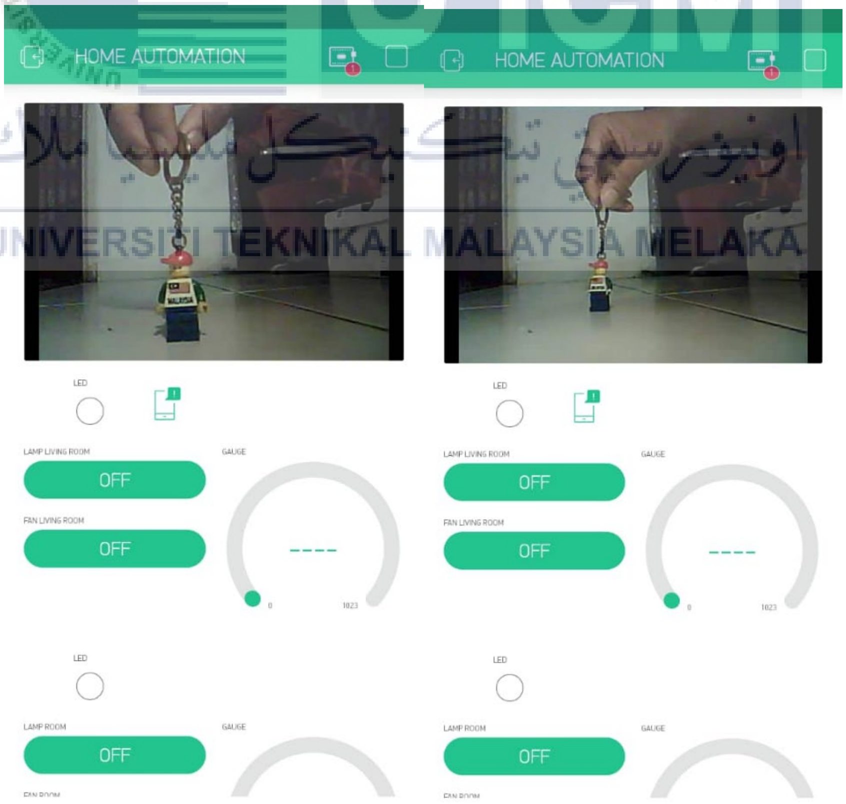


Figure 4.20 Streaming video ESP32-CAM from blynk

The Blynk Video Streaming widget continuously fetches frames from the MJPEG stream and displays them in real-time. Users can view the live video feed from the ESP32-CAM on the Blynk app, making it suitable for various applications. However, MJPEG streaming may have latency, and the quality may depend on the network conditions. The Blynk app may require a stable internet connection for smooth video streaming. In summary, the MJPEG streaming functionality in the provided code allows for continuous video streaming from the ESP32-CAM's camera. When integrated with the Blynk app using the Video Streaming widget, users can remotely view the live video feed on their smartphones or tablets, adding a visual monitoring aspect to their IoT projects.

4.7 Result Analysis

The test range of motion detection distance for each PIR sensor in the CCTV system using ESPCAM was conducted. The results are recorded and analyzed in this section. The time response of the PIR sensor notification system has been analyzed and illustrated in the plotted graph.

4.7.1 Test Range of Motion Detection Distance

The PIR sensor measures distance based on the transmitting and receiving signals and is capable of accurately measuring a distance up to 100cm. Due to the extensive coverage range, this project limits the distance range for motion detection from 3cm to 90cm to prevent the detection of undesired motion. Testing has been conducted to determine the time taken for the PIR sensor's output signal (Out) to remain in a high state at different distances. The time taken for the Out pin to transition from a low state to a high state corresponds to the distance travelled . When a detected motion is closer to the PIR sensor, the time taken for the Out pin to stay at a high status is shorter, resulting in faster response times for detecting closer objects.

Table 4.1 Test Range of Motion Detection Distance for Motion Sensor (PIR Sensor)

Test Range	Distance of Motion Detection (cm)	Time taken for output trigger stays high status (s)
1	5	0.5
2	10	0.6
3	15	0.7
4	20	0.8
5	25	0.9
6	30	1.0

Based on the Table 4.1, the time taken for the output of motion sensor to stay high, on average, increases by 0.1s for every 5cm increase in the distance of motion detection. The longest recorded time for the output to stay at a high status was 1.0s when the distance of motion detection was 30cm. Conversely, the shortest time for the output to stay high was 0.5s when the distance of motion detection was 5cm.

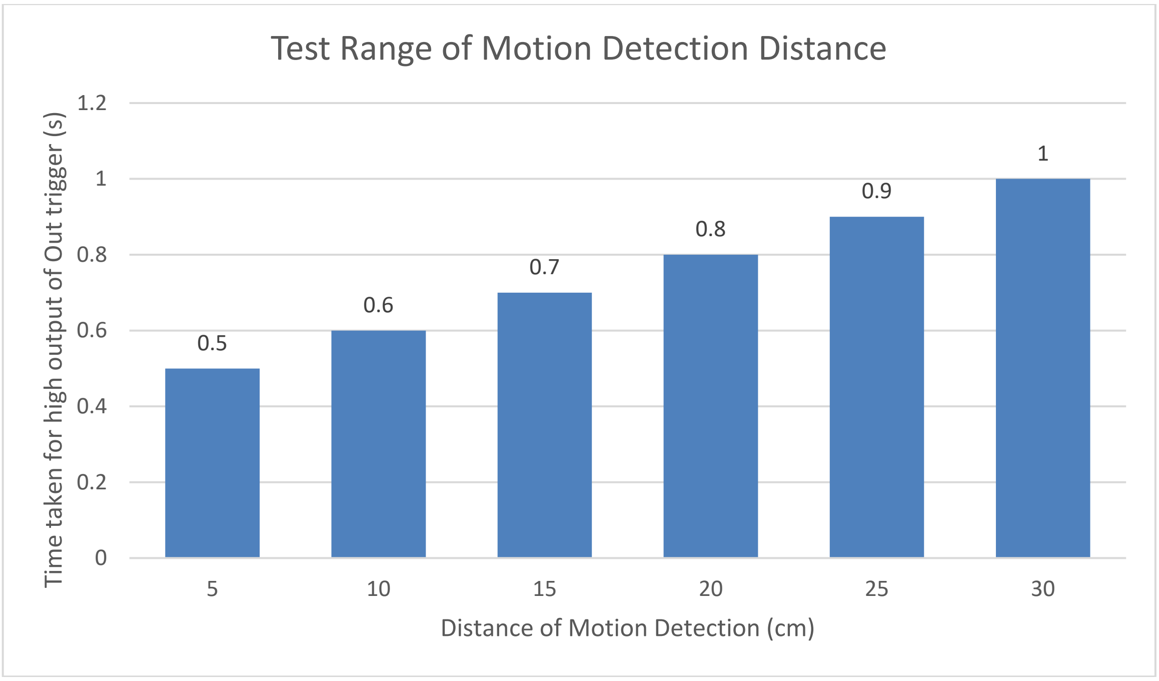


Figure 4.21 Graph of test range of motion detection distance for Motion Sensor (PIR Sensor)

4.7.2 Response Time for CCTV Notification System

The results of response time for the CCTV using ESPCAM and PIR sensor were gathered based on the time taken to receive notifications in the Telegram Bot once motion is detected. In this scenario, motion detection occurs when the PIR sensor measures a distance within the pre-set motion detection range. A graph depicting the time response for motion detection versus coverage distance has been generated, as illustrated in Figure 4.20. The time response shows a slight increase with the distance for motion detection. According to the trend line in the graph, the time response tends to rise as the distance of motion detection increases from 3cm to 90cm. A small difference in the distance of motion detection does not significantly affect the time response of the system.

For instance, it took only 1.96s to send a notification when motion was detected at a distance of 3cm, while it took 2.14s to deliver a notification when motion was detected at a distance of 90cm. The average time response for the CCTV using ESPCAM and PIR sensor system was between 1.92s and 2.14s. There was an abrupt increase in the time taken to receive a notification at distances of 45cm and 48cm. This may be due to the possibility of human error, as the time taken to receive notifications was recorded manually. To obtain accurate results, it is suggested to calculate the average time response by recording it three times at a specific distance.

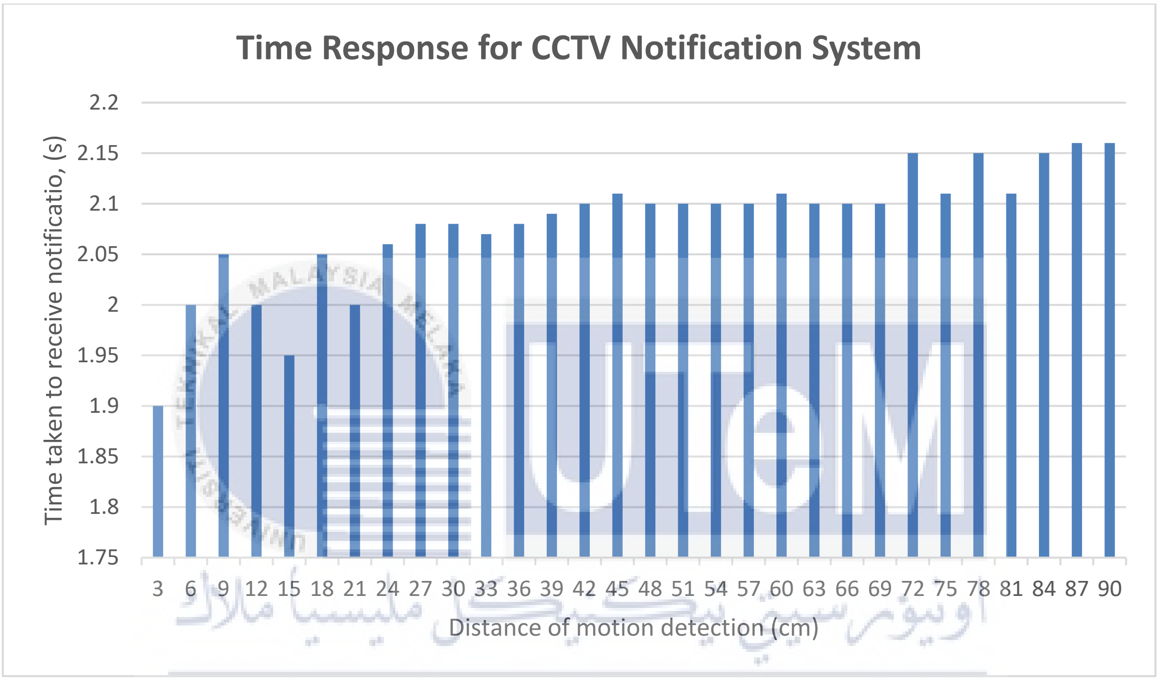


Figure 4.22 Time Response of CCTV Notification System

Table 4.2 illustrates the overall time response of the CCTV using ESPCAM and PIR sensor system based on a range of distances. The data collected was not accurate within the first 15cm. The overall time response roughly increased from 0.03s to 0.04s in the categorized distance range. The expected average time response for further distance range can be 2.04s to 2.05s.

Table 4.2 Average Response of CCTV Notification System

Distance range (cm)	Average Response
< 15	Data collected not accurate
18-30	Average time response about 2.06s
33-48	Average time response about 2.10s
51-69	Average time response about 2.10s
72-90	Average time response about 2.16s

4.8 Summary

In conclusion, this project has effectively developed a smart mirror system utilizing a Raspberry Pi 4B as the primary processor, ESP32 microcontrollers for home automation, and an ESP32-CAM camera for CCTV security. The system seamlessly integrates voice assistance, home automation, and face recognition, taking home security and automation to new heights. The comprehensive setup instructions for the Raspberry Pi 4B, covering display and voice assistance configurations, serve as a detailed guide for users to replicate the project.

The project's adaptability is showcased through the incorporation of various technologies, such as Telegram for remote monitoring, Blynk for IoT applications, and the utilization of MJPEG streaming for a continuous video feed from the ESP32-CAM. The voice-controlled assistant, named "FRIDAY," features a well-structured Python script that integrates multiple APIs and libraries, making it a robust component of the smart mirror system.

Measures the time taken for the PIR sensor to trigger motion, the duration after motion is triggered, and subsequently sending this information to Telegram. This modification enhances the project's focus on real-time motion detection and notification. The implementation of a CCTV notification system using the ESP32-CAM extends the project's capabilities, enabling users to capture images, stream video, and remotely control devices through Telegram commands. The detailed breakdown of functionalities, including image capturing and video recording, provides a clear understanding of the project's capabilities.

In summary, this smart mirror system successfully integrates hardware components, software scripts, and real-time motion detection to create a comprehensive home automation and security solution. The step-by-step instructions and detailed explanations make it accessible for users to replicate and customize the project based on their specific needs. Overall, the project significantly contributes to advancing interconnected technologies for enhanced home automation and security.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 INTRIDUCTION

This chapter discusses the conclusion and the future work for the Smart Mirror System through Raspberry Pi. The performance of Smart Mirror System through Raspberry Pi has been determined and recommendation for this project was suggested in this chapter.

5.2 CONCLUSION

In this project, a substantial effort has been dedicated to exploring the realm of IoT and utilizing it in the development of a Smart Mirror System using Raspberry Pi. The primary goal of the system is to revolutionize home automation and security through voice assistance, fundamentally transforming the way individuals interact with their living spaces. The envisioned outcome is a centralized platform, facilitated by the smart mirror, enabling users to effortlessly monitor and control various aspects of their living environment.

The research and proposed implementation of the IoT-based Smart Mirror System underscore the increasing potential of digital innovations in enhancing our daily lives. This system seamlessly integrates essential day-to-day information, such as the date, time, and weather, while concurrently addressing the critical aspects of home security and automation.

Moreover, the project has made significant strides in integrating advanced home automation and security features. Through the development of both software and hardware interfaces that interact with a variety of smart devices, the smart mirror system lays the foundation for a holistic and interconnected ecosystem within a household.

The addition of a voice assistance module further elevates the user experience of the smart mirror system. By enabling voice commands, it improves accessibility, ensuring convenience and user-friendliness, particularly for those who may face challenges with traditional user interfaces.

To broaden the project's horizons, considering its commercialization is vital. Breaking down the total implementation cost, which includes components such as ESP32, ESP CAM, sensors, speakers, microphone, webcam, LCD screen, frame, regulator, and socket, totaling almost RM 300 (excluding Raspberry Pi, which costs almost RM 300), provides valuable insights into the project's financial aspects. It is crucial to highlight that this implementation is more cost-effective than existing products in the market, which typically range from RM 1000 to RM 4500. Additionally, the unique feature of upgradeability sets this Smart Mirror System apart, offering users the potential for continuous improvement and feature enhancement, unlike many existing products that lack flexibility for upgrades.

In conclusion, this project not only works towards creating a revolutionary home automation and security system but also challenges the pricing models of existing products, making it a more accessible and adaptable solution for users. It marks a significant step forward in the future of home automation systems, setting a precedent for subsequent technological innovations.

.

5.3 RECOMMENDATION AND FUTURE WORK

This section determines the suggestions for developing this Smart Mirror System through Raspberry Pi.

5.3.1 Enhanced User Personalisation

In the future, the level of customisation available to users could be increased. This could include allowing users to select the types of information displayed on the mirror, changing the layout of the interface, or choosing from a variety of voice assistants.

5.3.2 Expanded Smart Home Compatibility

While the system is currently planned to integrate with certain common smart devices, in the future, the range of devices the system is compatible with could be broadened. This will ensure the system can be adapted to as many home setups as possible.

5.3.3 Improved Voice Recognition

While the initial model will include a voice assistance module, future iterations could focus on refining this technology. This could involve expanding the range of recognised commands, improving voice recognition accuracy, or introducing multilingual support.

5.3.4 Energy Efficiency

Efforts could be made to reduce the power consumption of the Smart Mirror System. This could involve integrating sensors to detect when someone is in front of the mirror, turning off the display when not in use, or improving the system's overall energy efficiency.

5.3.5 Enhanced Security Features

Future work could focus on further enhancing the system's security features. This expansion might include integrating additional security systems, such as alarm or smoke detection, to provide a more comprehensive safety net for users. Additionally, exploring the incorporation of facial recognition technology could offer personalized user experiences and elevate the overall security posture of the system.

Furthermore, to reinforce the security measures, a consideration for future development could involve implementing a feature where the system can automatically notify law enforcement agencies, such as the police, in the event of a security breach or emergency. This enhancement would provide an extra layer of protection and support, ensuring a swift response to critical situations and contributing to the overall robustness of the security infrastructure.

Future endeavors in advancing the security features of the system could encompass integrating alarm and smoke detection, exploring facial recognition for enhanced personalization and security, and adding the capability to notify the police in case of emergencies. These improvements would collectively contribute to a more sophisticated and proactive security system.

REFERENCES

[1] Suraj, I. Kool, D. Kumar, and S. Barma, “Visual Machine Intelligence for Home Automation,” in *2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU)*, IEEE, Feb. 2018, pp. 1–6. doi: 10.1109/IoT-SIU.2018.8519915.

[2] N. C. Noruwana, P. A. Owolawi, and T. Mapayi, “Interactive IoT-based Speech-Controlled Home Automation System,” in *2020 2nd International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*, IEEE, Nov. 2020, pp. 1–8. doi: 10.1109/IMITEC50163.2020.9334081.

[3] S. M. Shabber, M. Bansal, P. M. Devi, and P. Jain, “iHAS: An Intelligent Home Automation Based System for Smart City,” in *2021 IEEE International Symposium on Smart Electronic Systems (iSES)*, IEEE, Dec. 2021, pp. 48–52. doi: 10.1109/iSES52644.2021.00023.

[4] B. K. Akhmetzhanov, O. A. Gazizuly, Z. Nurlan, and N. Zhakiyev, “Integration of a Video Surveillance System Into a Smart Home Using the Home Assistant Platform,” in *2022 International Conference on Smart Information Systems and Technologies (SIST)*, IEEE, Apr. 2022, pp. 1–5. doi: 10.1109/SIST54437.2022.9945718.

[5] S. K. Vishwakarma, P. Upadhyaya, B. Kumari, and A. K. Mishra, “Smart Energy Efficient Home Automation System Using IoT,” in *2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, IEEE, Apr. 2019, pp. 1–4. doi: 10.1109/IoT-SIU.2019.8777607.

[6] S. K. Sooraj, E. Sundaravel, B. Shreesh, and K. Sireesha, “IoT Smart Home Assistant for Physically Challenged and Elderly People,” in *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, IEEE, Sep. 2020, pp. 809–814. doi: 10.1109/ICOSEC49089.2020.9215389.

[7] S. Mohan, N. A. J, and R. Sitharthan, “Enhanced Home Automation and Security Using IoT Architecture,” in *2022 International Conference on Power, Energy, Control and Transmission Systems (ICPECTS)*, IEEE, Dec. 2022, pp. 1–6. doi: 10.1109/ICPECTS56089.2022.10047164.

[8] P. Karthik, U. G. Keerthiv Sanjay, S. Abhiram, and V. Dinesh Kumar, “Implementation of Home Automation using Smart Mirror,” in *2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA)*, IEEE, Oct. 2021, pp. 1–6. doi: 10.1109/ICAECA52838.2021.9675660.

[9] S. Somani, P. Solunke, S. Oke, P. Medhi, and P. P. Laturkar, “IoT Based Smart Security and Home Automation,” in *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, IEEE, Aug. 2018, pp. 1–4. doi: 10.1109/ICCUBEA.2018.8697610.

[10] H. V. Tomar, A. Anand, H. L. Harsha, A. Deshwal, and B. N. K, “‘Smart Home Automation Device’ Using Raspberry Pie and Arduino Uno,” in *2022 IEEE 1st International Conference on Data, Decision and Systems (ICDDS)*, IEEE, Dec. 2022, pp. 01–06. doi: 10.1109/ICDDS56399.2022.10037544.

[11] S. Amit, A. S. Koshy, S. Samprita, S. Joshi, and N. Ranjitha, “Internet of Things (IoT) enabled Sustainable Home Automation along with Security using Solar Energy,” in *2019 International Conference on Communication and Electronics Systems (ICCES)*, IEEE, Jul. 2019, pp. 1026–1029. doi: 10.1109/ICCES45898.2019.9002526.

[12] S. Dash and P. Choudekar, “Home Automation using Smart Devices and IoT,” in *2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, IEEE, Sep. 2021, pp. 1–5. doi: 10.1109/ICRITO51393.2021.9596533.

[13] M. Prasad, A. P. Singh, and Y. Kumar, “Home Automation including Switching of appliance speed control of ac fan and brightness control of incandescent bulb,” in *2022 IEEE 10th Power India International Conference (PIICON)*, IEEE, Nov. 2022, pp. 1–5. doi: 10.1109/PIICON56320.2022.10045125.

[14] “peppe8o, peppe8o (2019) Raspberry pi 3 Model A+ datasheet, peppe8o. Available at: https://peppe8o.com/raspberry-pi-3-model-a-datasheet/ (Accessed: 10 June 2023). .”

[15] “Pololu - Raspberry Pi 3 model B+ (no date) Pololu Robotics &amp; Electronics. Available at: https://www.pololu.com/product/2797 (Accessed: 10 June 2023). .”

[16] “Log in (no date) Raspberry Pi Zero W from Raspberry Pi. Available at: https://www.electronicsdatasheets.com/manufacturers/raspberry-pi/parts/raspberry-pi-zero-w (Accessed: 10 June 2023). .”

[17] “Raspberry Pi (no date) Raspberry pi 4 model B specifications, Raspberry Pi. Available at: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/ (Accessed: 10 June 2023). .”

[18] “Westover, B. (2021a) Raspberry pi 4 model B review, Tom’s Guide. Available at: https://www.tomsguide.com/reviews/raspberry-pi-4-model-b (Accessed: 10 June 2023). .”

[19] “Raspberry pi camera board - night vision &amp; adjustable-focus lens (5MP) (no date) Wiki. Available at: http://wiki.sunfounder.cc/index.php?title=Raspberry_Pi_Camera_Board_-_Night_Vision_%26_Adjustable-Focus_Lens_%285MP%29 (Accessed: 10 June 2023). .”

[20] “Raspberry pi camera v1.3 (no date) Maker Portal. Available at: https://makersportal.com/shop/raspberry-pi-camera (Accessed: 10 June 2023). .”

[21] “HC-SR501 Pir Sensor (2021) Components101. Available at: https://components101.com/sensors/hc-sr501-pir-sensor (Accessed: 15 June 2023). .”

[22] “Partners, D. and Instructables (2018) How pir sensor work, Instructables. Available at: https://www.instructables.com/How-PIR-Sensor-Work/ (Accessed: 15 June 2023). .”

[23] “How HC-SR04 ultrasonic sensor works &amp; how to interface it with Arduino (2023) Last Minute Engineers. Available at: https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/ (Accessed: 10 June 2023). .”

[24] “What is telegram and why should I use it? (2022) Android Authority. Available at: https://www.androidauthority.com/what-is-telegram-messenger-979357/ (Accessed: 10 June 2023). .”

[25] “What is python? executive summary (no date) Python.org. Available at: https://www.python.org/doc/essays/blurb/ (Accessed: 10 June 2023). .”

[26] “Documentation for blynk, the most popular IOT platform for businesses. (no date) docs.blynk.cc. Available at: http://docs.blynk.cc/ (Accessed: 10 June 2023).”

APPENDICES

Appendix A Gantt chart for PSM 2

NO	Project Activity	Week													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Meeting with supervisor	🟨													
2	Meeting with JK PSM	🟨													
3	Result of 3d printing		🟨												
4	New 3d design (final design)			🟨											
5	Claiming process			🟨	🟨										
6	Finishing process of Casing				🟨										
7	Complete Chapter 2: Adding new statistic					🟨									
8	Complete Chapter 3: Methodology						🟨	🟨	🟨						
9	Collecting data							🟨	🟨	🟨					
10	Complete Chapter 4: Result								🟨	🟨	🟨				
11	Submit report											🟨	🟨	🟨	
12	Preparation for presentation										🟨	🟨	🟨	🟨	🟨

Appendix B Coding Rasberry Pi

```
# Imports #
import speech_recognition as sr
from gtts import gTTS
import playsound
import os
import requests
import time
import wikipedia
import datetime
import pywhatkit
import urllib.request

# Voice Assistant Coding Section #

# Chuck Norris API
chuck_norris_api = 'https://api.chucknorris.io/jokes/random'

# News API Key
news_api_key = '8ca76eefada144d38f921a0d203376de'

# Weather API Key
weather_api_key = 'd516f0f0441f400b921154b70a71b852' #9/11/2023 20 day expirt

# Distance API Key
distance_api_key = 'iGsChZJjcZUvV9ZiGK4D8ztn5OzMw560'

# Telegram API
telegram_token = "6947470172:AAGjSqaD-AKyB9n85G4lhjmaKb-dNrUiYWY"
telegram_chat_id = "6947470172"

# Global variable for file_path
file_path = "D:/OneDrive/OneDrive - Universiti Teknikal Malaysia
Melaka/Desktop/SEM 7/PROJECT/PEOJECT/SMART MIRROR FINAL/hai.txt"

def saveFile():
   global file_path
   jarvis_talk('What note do you want to save?')
   note_text = jarvis_listen()

   with open(file_path, 'w') as file:
      file.write(note_text)

   jarvis_talk('Note saved successfully.')

def read_note():
   if os.path.exists(file_path):
      with open(file_path, 'r') as file:
         saved_note = file.read()
```

```
        jarvis_talk('Here is the note you saved: ' + saved_note)
    else:
        jarvis_talk('No notes found. Save a note first.')


# LED Control ESP32_02
root_url_2= "http://192.168.0.102"

def sendRequest_2(url):
    urllib.request.urlopen(url)  # Send request to ESP

def control_led_2(request_number):
    if request_number == 1:
        sendRequest_2(root_url_2 + "/ledon13")
        print("Led is on")
    elif request_number == 2:
        sendRequest_2(root_url_2 + "/ledoff13")
        print("Led is off")
    elif request_number == 3:
        sendRequest_2(root_url_2 + "/ledon12")
        print("Led is on")
    elif request_number == 4:
        sendRequest_2(root_url_2 + "/ledoff12")
        print("Led is off")
    elif request_number == 5:
        sendRequest_2(root_url_2 + "/ledon2")
        print("Led is on")
    elif request_number == 6:
        sendRequest_2(root_url_2 + "/ledoff2")
        print("Led is off")
    elif request_number == 7:
        sendRequest_2(root_url_2 + "/onall")
        print("Led is on")
    elif request_number == 8:
        sendRequest_2(root_url_2 + "/offall")
        print("Led is off")
    else:
        print("Invalid request number")

# LED Control ESP32_01
root_url_1= "http://192.168.0.101"

def sendRequest_1(url):
    urllib.request.urlopen(url)  # Send request to ESP

def control_led_1(request_number):
    if request_number == 1:
        sendRequest_1(root_url_1 + "/ledon13")
        print("Led is on")
    elif request_number == 2:
        sendRequest_1(root_url_1 + "/ledoff13")
        print("Led is off")
    elif request_number == 3:
        sendRequest_1(root_url_1 + "/ledon12")
        print("Led is on")
    elif request_number == 4:
        sendRequest_1(root_url_1 + "/ledoff12")
        print("Led is off")
    elif request_number == 5:
        sendRequest_1(root_url_1 + "/ledon2")
        print("Led is on")
    elif request_number == 6:
        sendRequest_1(root_url_1 + "/ledoff2")
        print("Led is off")
    elif request_number == 7:
        sendRequest_1(root_url_1 + "/onall")
        print("Led is on")
    elif request_number == 8:
        sendRequest_1(root_url_1 + "/offall")
        print("Led is off")
    else:
        print("Invalid request number")

# Function to send a message to Telegram
def send_msg_to_telegram(text):
    formatted_text = f"/{text.replace(' ', '_')}"  # Remove spacing from the text
    url_req =
f"https://api.telegram.org/bot{telegram_token}/sendMessage?chat_id={telegram_c
hat_id}&text={formatted_text}"
    results = requests.get(url_req)
    return results.json()

# Function to receive messages from Telegram
def receive_msg_from_telegram():
    # Replace 'getUpdates' with the appropriate method based on your Telegram
bot's API
    url_req = f"https://api.telegram.org/bot{telegram_token}/getUpdates"
    results = requests.get(url_req)
    messages = results.json().get('result', [])
    if messages:
        last_message = messages[-1]['message']['text']
        return last_message
    return None



def time_now():
    today_date = datetime.datetime.now()
    hour = today_date.strftime("%I")
    minute = today_date.strftime("%M")
    meridiem = today_date.strftime("%p")
    time_now = 'It is ' + hour + ':' + minute + ' ' + meridiem
    print(time_now)
    jarvis_talk('the current now is ' + time_now)

def weekday_now():
    weekday_today = datetime.datetime.now().strftime("%A")
    print(weekday_today)
    jarvis_talk('today is ' + weekday_today)

def date_now():
    today_date = datetime.datetime.now().strftime("%Y-%m-%d")
    print(today_date)
    jarvis_talk('the current date is ' + today_date)

#wikipwdia
def wikipedia_info():
    jarvis_talk('I am happy to help. Let me know what I should search for you on
Wikipedia?')
    wiki_listen = jarvis_listen()
    print(wiki_listen)
    wiki_result = wikipedia.summary(wiki_listen, sentences=3)
    print(wiki_result)
    jarvis_talk(wiki_result)

#distance
def distance_info():
    jarvis_talk('Sure, let me know the starting point')
    location_one = jarvis_listen()
    print(location_one)
    time.sleep(1)
    jarvis_talk('Allright, and now tell me the final destination')
    location_two = jarvis_listen()
    print(location_two)
    jarvis_talk('Give me one moment, I will use my smart brain to calculate the
distance for you')
    dist_url = 'http://www.mapquestapi.com/directions/v2/route?key=' +
distance_api_key + '&from=' + location_one +'&to=' +location_two + '&unit=k'
    dist_request = requests.get(dist_url).json()
    distance = round(dist_request['route']['distance'],2)
    distance_result = 'The distance between ' + location_one +' and ' + location_two
+ ' is ' + str(distance) + ' kilometers'
    print(distance_result)
    jarvis_talk(distance_result)

#weather
def get_weather():
    jarvis_talk('No problem, I will look it up for you. What city are you interested
in?')
    weather_input = jarvis_listen()
```

```
   print(weather_input)

   weather_url = 'https://api.weatherbit.io/v2.0/current?city=' + weather_input +
'&key=' + weather_api_key
   weather = requests.get(weather_url).json()
   temperature = weather['data'][0]['temp']
   description = weather['data'][0]['weather']['description']
   weather_result = 'The temperature in ' + weather_input + ' is ' + str(
      temperature) + ' degrees and you can see ' + description
   print(weather_result)
   jarvis_talk(weather_result)

#news
def get_news():
   news_url = 'https://newsapi.org/v2/top-headlines?country=my&apiKey=' +
news_api_key
   news = requests.get(news_url).json()
   articles = news['articles']

   news_headlines = []
   for art in articles:
      news_headlines.append(art['title'])

   for i in range(3):
      print(news_headlines[i])
      jarvis_talk(news_headlines[i])

#joke
def chuck_norris():
   cn_data = requests.get(chuck_norris_api)
   cn_json = cn_data.json()
   joke = cn_json['value']
   print(joke)
   jarvis_talk(joke)

# Calculator

# convert speech to text so we can use the text for the next step
def jarvis_listen():
   # create recognizer
   r = sr.Recognizer()
   # what we speak into the microphone should be our source
   with sr.Microphone() as source:
      # use the listen function so the recognizer can cathch the source (our mic)
      audio = r.listen(source)
      text = ''

      try:
         text = r.recognize_google(audio)
```

```
        except sr.RequestError as re:
            print(re)

        except sr.UnknownValueError as uve:
            print(uve)

        except sr.WaitTimeoutError as wte:
            print(wte)

    text = text.lower()
    return text

# convert text to speech english
def jarvis_talk(text):
    # create audio data
    file_name = 'audio_data.mp3'
    # convert text to speech
    tts = gTTS(text=text, lang='en')   #, tld='com.au')
    # save file
    tts.save(file_name)
    # play file
    playsound.playsound(file_name)
    # remove file
    os.remove(file_name)

# create a function which will give us back a reply based on the input text #
def jarvis_reply(text):

    print(text)

# smalltalk - what is your name?
    if 'what' in text and 'name' in text:
        jarvis_talk('My name is friday and I am your personal assistant')

    # smalltalk - why do you exist?
    elif 'why' in text and 'exist' in text:
        jarvis_talk('I was created to work for you. I dont need a break and I will never
ask for days off')

    # smalltalk - when do you sleep?
    elif 'when' in text and 'sleep' in text:
        jarvis_talk('I never sleep. I was created to support you 24 hours')

    # smalltalk- are you stupid?
    elif 'you' in text and 'stupid' in text:
        jarvis_talk('No, I am not stupid. My grandmother told me that there are no
stupid persons out there. '
                + 'I try to give my best everyday and learn continuously')

    # Smalltalk - Favorite Movie?
    elif 'favorite' in text or 'favourite' in text and 'movie' in text:
        jarvis_talk("My favorite movie is Titanic. I watch it with my friends all the
time")

    # Smalltalk - Favorite Movie?
    elif 'favorite' in text or 'favourite' in text and 'colour' in text:
        jarvis_talk("My favorite colour is blue.")

# Chuck Norris Jokes
    elif 'joke' in text:
        chuck_norris()

# Top 3 News-Headlines
    elif 'news' in text:
        jarvis_talk('Allright, let me tell you the first three headlines')
        get_news()
# Weather
    elif 'weather' in text:
        get_weather()
# Distance
    elif 'distance' in text:
        distance_info()
# Wikipedia
    elif 'wikipedia' in text:
        wikipedia_info()
# current time
    elif 'time' in text:
        time_now()
# current day
    elif 'day' in text:
        weekday_now()
#current date
    elif 'date' in text:
        date_now()
#youtube
    elif 'play youtube' in text:
        song = text.replace('play youtube', '')
        jarvis_talk('playing ' + song)
        pywhatkit.playonyt(song)
# Add this command to the jarvis_reply function

    elif 'home assistant' in text:
        jarvis_talk('Sure, what do you want me to do?')
        message_to_send = jarvis_listen()
        send_msg_to_telegram(message_to_send)

        # Wait for a response from Telegram
        jarvis_talk('Waiting for response...')
        received_message = receive_msg_from_telegram()
        if received_message:
            jarvis_talk(f"Received the following message from Telegram:
{received_message}")
        else:
            jarvis_talk('No response received from Telegram.')

# ESP32_01 LED Control
    elif 'control living room' in text:
        jarvis_talk("what can i do for u")
        led_command = jarvis_listen()
        if 'fan on' in led_command:
            control_led_1(1)
        elif 'fan off' in led_command:
            control_led_1(2)
        elif 'light on' in led_command:
            control_led_1(3)
        elif 'light off' in led_command:
            control_led_1(4)
        elif 'system on' in led_command:
            control_led_1(5)
        elif 'system off' in led_command:
            control_led_1(6)
        elif 'all on' in led_command:
            control_led_1(7)
        elif 'all off' in led_command:
            control_led_1(8)
        else:
            jarvis_talk('Invalid system command. Try again.')

    elif 'safe notes' in text:
        saveFile()

    elif 'read notes' in text:
        read_note()



    elif 'stop' in text:
        jarvis_talk('It was a pleasure to help you, I wish you a wonderful day')

    else:
        jarvis_talk('Excuse me, I did not get that. Can you please repeat it?')

# Voice Assistant Execution Section #
def execute_assistant():
    # personalize name
    jarvis_talk('Hi, I am FRIDAY I am here to support you. Can you please tell me
your name?')
    listen_name = jarvis_listen()
    jarvis_talk('Hi ' + listen_name + ', HAPPY TO SEE U')
    jarvis_talk('if u need me just call me')
    print(listen_name)


    while True:
        listen_jarvis = jarvis_listen()
        print(listen_jarvis)


        # Check if the wake word "Jarvis" is detected
        if 'friday' in listen_jarvis:
            jarvis_talk('Yes, how can I assist you?')

            # Continue listening for the user's command
            user_command = jarvis_listen()

            # Process the user's command
            jarvis_reply(user_command)

            # Check if the user wants to stop the assistant
            if 'stop' in user_command:
                jarvis_talk('It was a pleasure to help you. Have a wonderful day!')
                break

        # Check if the wake word "stop" is detected
        elif 'stop' in listen_jarvis:
            jarvis_talk('It was a pleasure to help you. Have a wonderful day!')
            break

# Call the function
execute_assistant()
```