**Project report on**

**Whatsapp Based Home Automation Using Raspberry Pi & Python**

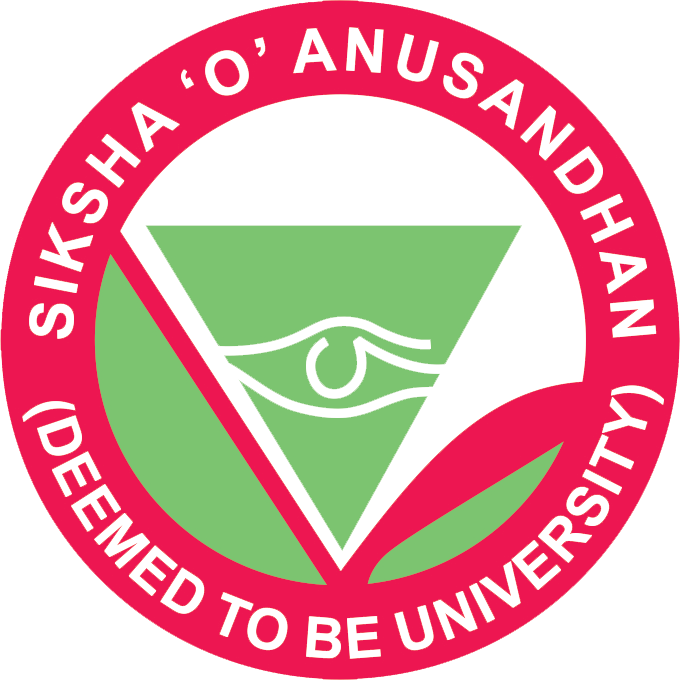
Submitted by

**Satyajit Panda (2341016091) (Group Leader)**

**Satya Swarup Nayak (2341014076)**

**Shruti Sahoo (2341019153)**

**B. Tech. (CSE(IOT) 4th Semester (Section–23412O3)**



**CENTRE FOR INTERNET OF THINGS**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

Institute of Technical Education and Research

**SIKSHA ‘O’ ANUSANDHAN**

**DEEMED TO BE UNIVERSITY**

Bhubaneswar, Odisha, India.

(May, 2025)

# Abstract

This project explores a practical and innovative approach to smart home automation using WhatsApp as the primary user interface, leveraging the capabilities of a Raspberry Pi Zero 2 W. Traditional home automation systems often rely on mobile applications or proprietary control hubs, which may limit accessibility and scalability. In contrast, this system empowers users to control home appliances—specifically a light (LED) and a fan (DC motor)—using simple WhatsApp messages from any location with internet connectivity.

The project integrates the **Twilio WhatsApp API**, a **Flask-based Python backend**, and the **gpiozero library** to interpret and respond to user commands. The Raspberry Pi acts as the central controller, receiving messages through a webhook, processing the command logic, and executing real-time GPIO operations to switch devices ON or OFF. Additionally, the system provides feedback to the user by sending confirmation messages, ensuring clear and interactive communication.

A major strength of this system lies in its **cost-effectiveness, portability, and ease of deployment**, using readily available hardware components like the L298N motor driver, 5V DC motor, and standard LEDs. It eliminates the need for dedicated apps, making it accessible to users familiar with WhatsApp and minimizing the learning curve. The project also demonstrates the foundational principles of the Internet of Things (IoT), showcasing how cloud APIs and microcontroller platforms can collaborate to create efficient automation systems.

Through structured phases of development—from requirements gathering and circuit design to code implementation and real-time testing—this project not only achieved its intended goals but also laid the groundwork for future enhancements such as multi-device control, sensor integration, and AI-based automation. This approach makes home automation more inclusive, adaptable, and intelligent.

# 

# Contents

[Abstract i](#_Toc132229941)

[Contents ii](#_Toc132229942)

[Chapter 01: Introduction 1](#_Toc132229943)

[1.1. Introduction 1](#_Toc132229944)

[1.2. Background 1](#_Toc132229945)

[1.3. Project Objectives 1](#_Toc132229946)

[1.4. Scope 1](#_Toc132229947)

[1.5. Project Management 1](#_Toc132229948)

[1.6. Overview and Benefits 2](#_Toc132229949)

[1.7. Organization of the Report 2](#_Toc132229950)

[Chapter 02: Background Review & Survey 3](#_Toc132229951)

[2.1. Related Works 3](#_Toc132229954)

[Chapter 03: Theoretical Aspects 4](#_Toc132229955)

[3.1. Internet of Things (IoT) 4](#_Toc132229958)

[3.2. Features of IoT 4](#_Toc132229959)

[3.3. Advantages of IoT 4](#_Toc132229960)

[3.4. Disadvantages of IoT 4](#_Toc132229961)

[3.5. Application areas of IoT 4](#_Toc132229962)

[3.6. IOT Technologies and Protocols 4](#_Toc132229963)

[3.7. Project Layout 5](#_Toc132229964)

[3.7.1. Brief Description 5](#_Toc132229965)

[Chapter 04: Hardware Requirements 6](#_Toc132229966)

[4.1. Raspberry Pi Zero 2W 6](#_Toc132229969)

[4.1.1. Features 6](#_Toc132229970)

[4.1.2. Pin Configuration 6](#_Toc132229971)

[4.2. L298N Motor Driver 6](#_Toc132229972)

[4.3. DC Motor &LED 6](#_Toc132229973)

[4.4.Adapter for power supply 6](#_Toc132229976)

[4.5. Block diagram of the proposed system 6](#_Toc132229977)

[4.5.1. Working of the system 6](#_Toc132229978)

[4.5.2. Circuit Diagram 6](#_Toc132229979)

[4.5.3. Components Required 6](#_Toc132229980)

[Chapter 05: Software Requirements 8](#_Toc132229981)

[5.1. NGROK,Flask(Python) 8](#_Toc132229984)

[5.2. Logic and Flowchart 8](#_Toc132229985)

[Chapter 06: Project development & Testing Aspects 9](#_Toc132229986)

[6.1. 9](#_Toc132229989)

[Chapter 07: Conclusion & Future Scope 10](#_Toc132229990)

[7.1. Result 10](#_Toc132229993)

[7.2. Conclusion 10](#_Toc132229994)

[7.3. Limitations 10](#_Toc132229995)

[7.4. Further Enhancement and Future Scope 10](#_Toc132229996)

[References 11](#_Toc132229997)

[Appendix 01 12](#_Toc132229998)

[A01.1. Code Listing 12](#_Toc132229999)

[A01.2. Main Code 12](#_Toc132230000)

[A01.3. Libraries 12](#_Toc132230001)

[Appendix 02 13](#_Toc132230002)

[A02.1. Project Proposal Form 13](#_Toc132230003)

[A02.2. Project Management 13](#_Toc132230004)

[A02.3. Bill of Material 13](#_Toc132230005)

[Appendix 03 14](#_Toc132230006)

[A03.1. Data Sheets 14](#_Toc132230007)

# Chapter 01: Introduction

## Introduction

This project focuses on integrating Internet of Things (IoT) principles with WhatsApp messaging for real-time control of home appliances.

## Background

Conventional home automation systems often rely on mobile apps or web interfaces. However, WhatsApp offers a universal and convenient platform for remote access, especially where users prefer simplicity and familiarity

## Project Objectives

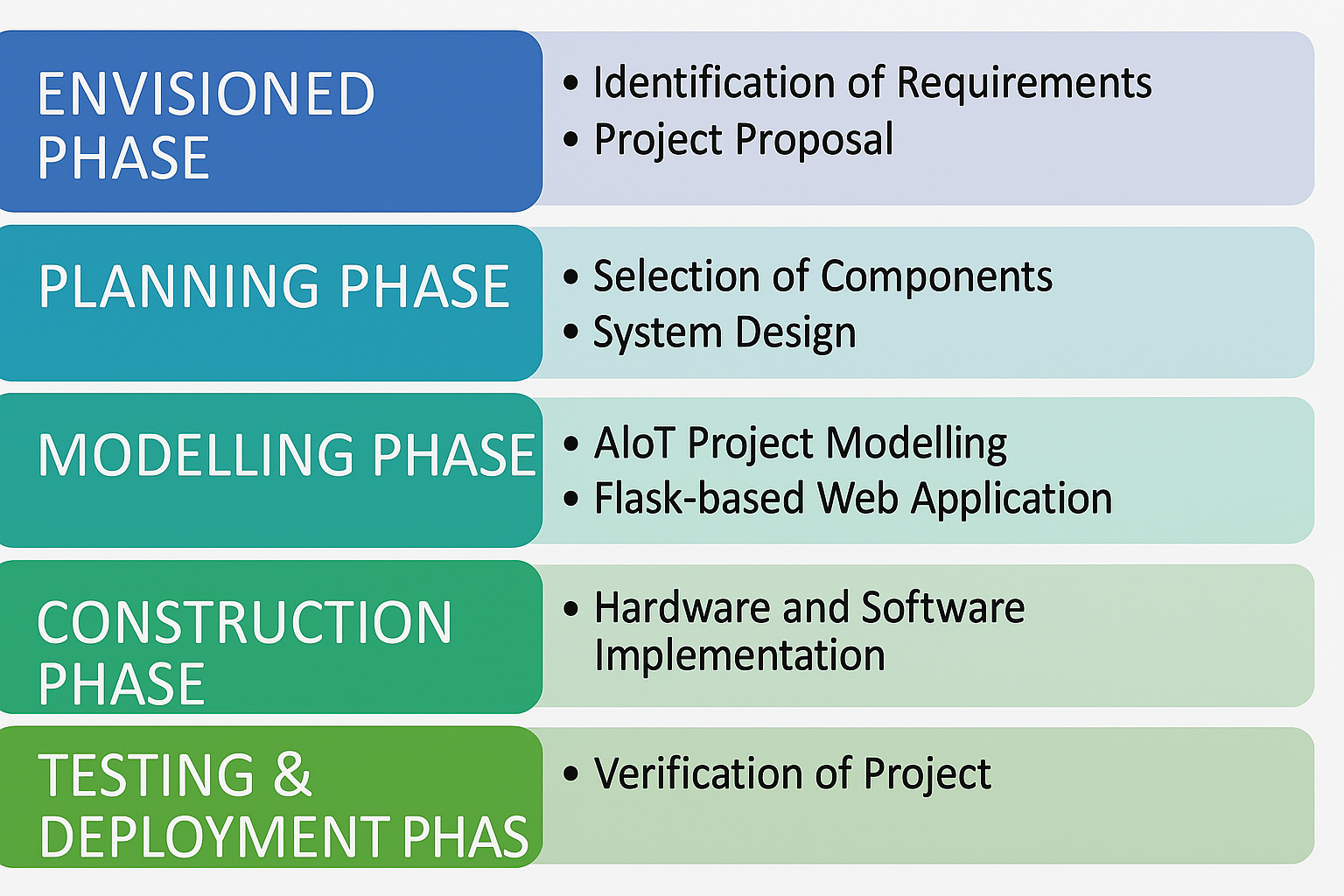
* To enable real-time control of light and fan through WhatsApp messages.
* To use a Raspberry Pi for device automation.
* To provide reliable acknowledgment via Twilio API.

## Scope

This project is designed for local or remote automation of small home appliances using text commands. It can be expanded to include sensors, relays, and additional actuators.

## Project Management

The project was divided into planning, development, testing, and documentation. Weekly milestones were set for hardware integration, API setup, and final testing.



**Figure 1. Model of phases in project management.**

## Overview and Benefits

The proposed system simplifies automation by using a universally accessible platform (WhatsApp). It eliminates the need for separate mobile apps, offering flexibility, affordability, and easy integration.

## Organization of the Report

Chapter 2 presents related works. Chapter 3 covers IoT fundamentals. Chapter 4 and 5 describe hardware and software. Chapter 6 outlines development and testing, followed by conclusion and future scope in Chapter 7.

# 

# Chapter 02: Background Review & Survey



## Related Works

In recent years, home automation has emerged as one of the most widely explored applications of the Internet of Things (IoT). Various projects and research studies have contributed to the development of IoT-based automation systems using platforms like mobile apps, web servers, voice assistants, and messaging services. However, the use of **WhatsApp** as a medium for home automation is relatively novel.

Key related works include:

* **Mobile App-Based Automation:** Projects using Blynk, MIT App Inventor, or Android Studio to control devices via custom apps. While powerful, these solutions often require installation and are platform-dependent.
* **Web Dashboard Systems:** Some systems use Flask or Django with a public IP to host a webpage for home control. This approach needs cloud hosting or port forwarding, which introduces security and setup challenges.
* **Voice Assistant Integration:** Integration with Amazon Alexa or Google Assistant allows hands-free control, but it requires additional APIs and compatible hardware. These systems are often expensive and complex to set up.
* **WhatsApp-Based Automation:** Few open-source projects and hobbyist solutions have experimented with Twilio’s WhatsApp API for sending commands. These typically control basic LEDs or relays but often lack feedback mechanisms, motor control, or error handling.

**Our project builds on these foundations by:**

* Using **Twilio’s WhatsApp API** for a widely accessible, platform-independent interface.
* Controlling **both an LED (light)** and **a DC motor (fan)** via Raspberry Pi GPIO.
* Providing **real-time acknowledgment messages** to users.
* Using **Flask server + Ngrok** for easy and secure webhook exposure without static IP.

This approach is user-friendly, doesn't require app installation, and is scalable for real-world IoT applications.

# 

# Chapter 03: Theoretical Aspects



## Internet of Things (IoT)

The Internet of Things (IoT) refers to the network of physical devices embedded with sensors, software, and connectivity capabilities to collect, exchange, and act upon data. IoT enables seamless integration of the physical world with digital systems, allowing real-time monitoring and control.

In this project, IoT is applied by integrating hardware components (Raspberry Pi, sensors, and actuators) with cloud-based services (Twilio API and WhatsApp). The Raspberry Pi acts as an IoT gateway, bridging communication between the user and connected devices. Commands sent over WhatsApp are processed by the Flask server, which then triggers GPIO operations on the Pi to turn devices ON or OFF.

IoT not only enhances automation but also offers scalability, remote accessibility, and improved user interaction. This project demonstrates a foundational example of IoT by connecting common household devices to a popular messaging platform without the need for custom mobile applications

## Features of IoT

1. Intelligence: IoT systems integrate data processing, decision-making logic, and responsiveness into devices, enabling smart interactions and autonomous operation.
2. **Connectivity:** IoT devices are interconnected through networks, allowing communication between devices and cloud-based platforms in real time.
3. **Dynamic Nature:** IoT systems adapt dynamically to changes in the environment or user input. They support real-time updates, status monitoring, and automatic actions.
4. **Enormous Scale:** IoT can scale to billions of devices globally, supporting vast networks of sensors, actuators, and users across different industries.
5. **Sensing:** IoT systems utilize sensors to collect accurate data from the environment, such as temperature, motion, light, and humidity, for informed decision-making.
6. **Heterogeneity:** IoT includes diverse devices, protocols, and platforms. Interoperability is essential to ensure that all components can communicate and function together.
7. g) **Security:** IoT devices require strong authentication, encryption, and access control to protect data privacy and prevent unauthorized usage.

## Advantages of IoT

1. **Communication:** IoT enables devices to communicate with each other and with centralized systems, improving coordination and real-time decision-making.
2. **Automation and Control:** IoT systems reduce the need for human intervention through intelligent automation and programmable behaviour.
3. **Information:** Continuous data collection and analysis help users gain better insights into system performance and environmental conditions.
4. **Monitoring:** IoT supports real-time tracking and monitoring of assets, devices, or spaces for safety, efficiency, and productivity.
5. **Efficiency:** Automation and data-driven decision-making enhance system performance, optimize resource usage, and reduce operational costs.

## Disadvantages of IoT

1. **Compatibility:** IoT includes a wide variety of devices and platforms, making standardization and integration challenging.
2. **Complexity:** Setting up and managing IoT systems can be technically complex and require expertise in hardware, networking, and software.
3. **Privacy/Security:** With devices connected to the internet, data security and user privacy become significant concerns.
4. **Safety:** Faults or vulnerabilities in IoT systems can lead to risks or damage in critical environments.

## Application areas of IoT

* **Smart Homes:** IoT enables automation of lighting, appliances, climate control, and security systems in residential spaces.
* **Healthcare:** Devices monitor patient vitals and medication schedules, enabling remote care and emergency alerts.
* **Agriculture:** IoT is used in precision farming for soil monitoring, irrigation management, and crop health tracking.
* **Industrial Automation:** Factories implement IoT for predictive maintenance, equipment monitoring, and production control.
* **Smart Cities:** IoT contributes to traffic management, waste control, street lighting, and environmental monitoring.
* **Retail:** IoT helps in inventory tracking, smart shelves, and personalized customer experiences.
* **Transportation:** Fleet tracking, smart logistics, and autonomous vehicle systems rely on IoT integration.

## IOT Technologies and Protocols

* **Wi-Fi (IEEE 802.11):** The Raspberry Pi connects to the internet through a Wi-Fi network (e.g., mobile hotspot) to receive and send commands.
* **HTTP Protocol:** Used by the Flask web server running on the Raspberry Pi to handle incoming HTTP POST requests from Twilio.
* **Twilio WhatsApp API:** Enables message communication between the user and Raspberry Pi. It acts as the gateway for receiving commands over WhatsApp.
* **Ngrok:** Tunnels the Flask server running locally on the Raspberry Pi to a public internet address, enabling webhook access by Twilio.
* **GPIO Protocol:** Raspberry Pi uses its GPIO pins to control electrical devices like LEDs and motors.
* **gpiozero Library:** Python library used for high-level GPIO pin control in the Raspberry Pi, simplifying control of devices.

## Project Layout

**Figure 2. Layout of project module**

### Brief Description

The block diagram illustrates the high-level architecture of the WhatsApp-Based Home Automation System. It is divided into two main modules:

* **Hardware Module:**
  + **Raspberry Pi Zero 2 W:** Acts as the central IoT controller.
  + **L298N Motor Driver:** Interfaces the Raspberry Pi with the DC motor (fan) to enable motor control.
  + **DC Motor (Fan):** Represents a home appliance controlled through WhatsApp commands.
  + **LED & Resistor:** Simulates a light device, connected to GPIO for switching ON/OFF.
  + **5V 2A Adapter:** Powers the Raspberry Pi and motor driver.
  + **Breadboard & Wires:** Used for prototyping and electrical connections.
* **Software Module:**
  + **Flask Web Server:** Runs a lightweight server on Raspberry Pi to listen for incoming HTTP POST requests from Twilio.
  + **Twilio WhatsApp API:** Acts as the interface between the user’s WhatsApp and the Flask server, delivering and receiving commands.
  + **Ngrok:** Establishes a secure tunnel to expose the local Flask server to the public internet.
  + **Python Code (whatsapp\_bot.py):** Handles GPIO control and command parsing.

This structured diagram ensures a clear understanding of how hardware components interact with software layers to complete the automation process.

# 

# Chapter 04: Hardware Requirements



## Raspberry Pi Zero 2W

### Features

###  Quad-core 1GHz CPU, 512MB RAM

 **Built-in Wi-Fi & Bluetooth**

 40-pin **GPIO header** for hardware control

 **MicroSD, mini HDMI, micro USB OTG + power**

### Pin Configuration of jumper wire

 **GPIO17** – LED (Light control)

 **GPIO18** – ENA (Motor enable)

 **GPIO22, GPIO23** – Motor control (IN1, IN2)

 **GND, 5V** – Power & ground for components

## L298N Motor Driver

## LED

## 5V DC Motor and Resistor

## Jumper Wires

## 5V 2A Adapter

## Block diagram of the proposed system

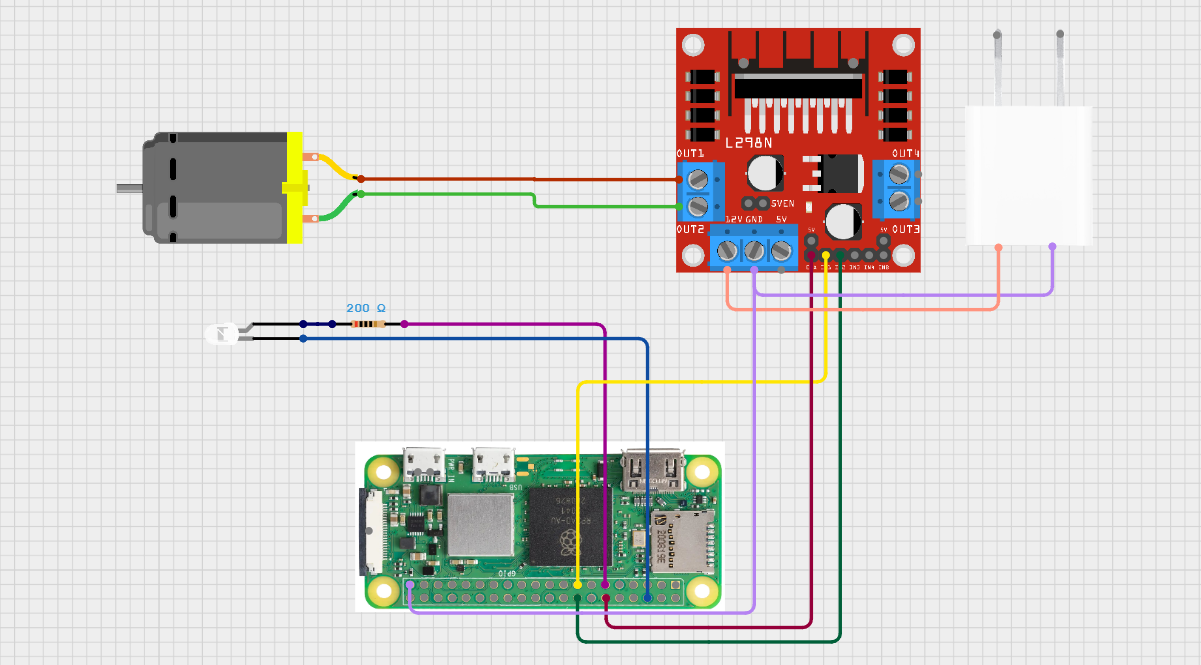


### Working of the system

The system works by integrating **WhatsApp communication with IoT hardware** using a Raspberry Pi. Here's how it functions:

1. **User Interaction:**
   * The user sends a message like **"light on"** or **"fan off"** through WhatsApp.
2. **Message Handling:**
   * The message is received by **Twilio’s WhatsApp API**, which forwards it to the **Flask server** running on the Raspberry Pi.
3. **Command Processing:**
   * The Python script (whatsapp\_bot.py) reads the message.
   * Based on the content, it activates or deactivates specific **GPIO pins**.
4. **Hardware Response:**
   * **GPIO17** controls an LED (light).
   * **GPIO22, GPIO23**, and **ENA (GPIO18)** control the L298N motor driver, which operates the **DC motor (fan)**.
5. **Acknowledgment:**
   * The system sends a **confirmation message back** to the user via WhatsApp (e.g., "✅ Light is ON").

### Circuit Diagram



### Components Required

**Table 1. Component listing.**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Component and Specification** | **Quantity** |
|  | Raspberry Pi Zero 2W | 1 |
|  | L298N Motor Driver | 1 |
|  | DC Motor (5v) with Fan | 1 |
|  | LED | 1 |
|  | Resistor (220 ohm) | 1 |
|  | Jumper Wires | APR |
|  | 5V 2AMP Adapter with Female Power Connector | 1 |

# Chapter 05: Software Requirements



The software requirements for the WhatsApp-Based Home Automation project are centered around Python development and IoT integration using APIs and libraries compatible with the Raspberry Pi platform.

## Operating System

* **Raspberry Pi OS (Lite or Full)** – A Debian-based Linux OS used to run the project on Raspberry Pi Zero 2 W.

## Programming Language

* **Python 3** – Main programming language used to control GPIO pins and create the WhatsApp automation bot.

## Required Software Tools and Libraries

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Software/Library** | **Purpose** |
| 1. | Python 3.x | Core programming for automation and GPIO interaction |
| 2. | Flask | Web server framework to create a local server for receiving messages |
| 3. | Twilio API | To send/receive WhatsApp messages and connect to WhatsApp Sandbox |
| 4. | Ngrok | Used to expose the Flask server to the internet using a public URL |
| 5. | gpiozero | Python library to control Raspberry Pi GPIO pins easily |
| 6. | pip | Python package manager used to install libraries |

## Setup and Configuration Summary

* Flask server is created using Python on the Raspberry Pi.
* Twilio is configured with a webhook pointing to the Ngrok URL.
* GPIO pins are mapped in Python code using gpiozero to control LED and motor.
* The Pi must be connected to the internet via Wi-Fi or hotspot for full functionality.

This combination of tools ensures seamless integration between messaging and device control, enabling real-time remote automation.

# 

# Chapter 06: Project development & Testing Aspects



## Development Stages

The project was developed through the following stages:

1. **Component Collection & Assembly**
   * All required hardware (Raspberry Pi, L298N driver, LED, motor, power supply) was gathered and mounted on a breadboard.
2. **Environment Setup**
   * Raspberry Pi OS was installed on a 32GB microSD card.
   * Python, Flask, Twilio library, and gpiozero were installed using pip.
3. **Code Development**
   * A Flask server was created to receive WhatsApp messages.
   * GPIO pin control was implemented using the gpiozero library.
   * Python logic was written to parse and respond to “light on/off” and “fan on/off” commands.
4. **Ngrok & Twilio Configuration**
   * Ngrok was used to tunnel the local Flask server.
   * Twilio’s WhatsApp Sandbox was configured to forward incoming messages to the Ngrok URL.
5. **Integration Testing**
   * The system was tested by sending messages from WhatsApp.
   * GPIO responses were verified by checking LED and motor behaviour.

## Testing Outcomes

* **LED Test:** Successful ON/OFF switching using WhatsApp messages “light on” and “light off.”
* **Motor Test:** DC fan turned ON and OFF correctly using “fan on” and “fan off.”
* **Twilio Webhook:** Proper message delivery and acknowledgment.
* **Power Supply Test:** Stable operation observed with a 5V 2A adapter.

## Issues Faced & Solutions

|  |  |
| --- | --- |
| **Issue** | **Solution** |
| Motor vibrating but not rotating | Used external 5V 2A adapter for sufficient current |
|  |  |
| Twilio not responding | Fixed webhook and verified Twilio sandbox setup |
| ENA pin not working | Soldered the ENA connection securely to GPIO18 |
|  |  |

This iterative development approach ensured a reliable, functioning system with responsive control and feedback through WhatsApp.

# Chapter 07: Conclusion & Future Scope



## Result

The project successfully achieved its goal of creating a WhatsApp-controlled home automation system. The LED (light) and DC motor (fan) could be turned ON and OFF using simple WhatsApp messages. The Raspberry Pi responded instantly and accurately via GPIO, and the system provided acknowledgment replies through Twilio's WhatsApp API.

## Conclusion

This project demonstrates a practical and low-cost solution for remote home automation using IoT. It integrates familiar tools like WhatsApp and Python to provide user-friendly control of basic appliances. The modularity of the system makes it scalable and adaptable to more complex automation scenarios.

## Limitations

* Dependent on stable internet connectivity for both the Raspberry Pi and mobile device.
* Twilio’s free sandbox plan may have message limitations or delays.
* Limited to only two devices (light and fan) in the current implementation.
* Security mechanisms like user authentication are not included.

## Further Enhancement and Future Scope

* Extend the system to control multiple appliances like smart plugs, bulbs, and sensors.
* Integrate sensor-based automation (e.g., motion, temperature).
* Develop a custom mobile or web dashboard for more intuitive control.
* Add user authentication and logging for better security and monitoring.
* Upgrade from Twilio sandbox to full API for production-level performance.

# References

1. Twilio WhatsApp API Documentation – <https://www.twilio.com/docs/whatsapp>
2. Raspberry Pi Official Website – <https://www.raspberrypi.com/>
3. Gpiozero Python Library – <https://gpiozero.readthedocs.io/en/stable/>
4. Flask Documentation – <https://flask.palletsprojects.com/>
5. Ngrok Documentation – <https://ngrok.com/docs>
6. L298N Motor Driver Datasheet – <https://components101.com/motor-driver/l298n-motor-driver-module>
7. Raspberry Pi GPIO Pinout – <https://pinout.xyz/>
8. Various IoT forums and GitHub repositories related to WhatsApp automation and home automation systems
9. <https://search.app/qzKmjbtUBauCtHC46>
10. <https://youtu.be/2bganVdLg5Q?feature=shared>

# 

# Appendix 01

## A01.1. Code Listing

The project is powered by a Python script that handles incoming WhatsApp messages and controls GPIO pins on the Raspberry Pi. This appendix contains the full source code and descriptions.

## A01.2. Main Code

|  |
| --- |
| from flask import Flask, request  from gpiozero import LED, OutputDevice  from twilio.twiml.messaging\_response import MessagingResponse  app = Flask(\_name\_)  # Devices  led = LED(17)  in1 = OutputDevice(22)  in2 = OutputDevice(23)  ena = OutputDevice(18)  @app.route("/bot", methods=["POST"])  def bot():  msg = request.values.get('Body', '').lower()  resp = MessagingResponse()  if 'light on' in msg:  led.on()  resp.message("✅ Light is ON")  elif 'light off' in msg:  led.off()  resp.message("✅ Light is OFF")  elif 'fan on' in msg:  ena.on()  in1.on()  in2.off()  resp.message("✅ Fan is ON")  elif 'fan off' in msg:  ena.off()  in1.off()  in2.off()  resp.message("✅ Fan is OFF")  else:  resp.message("Send:\nlight on / light off\nfan on / fan off")    return str(resp)  if \_name\_ == "\_main\_":  app.run(debug=True) |

## A01.3. Libraries

The following Python libraries are used in the project:

* **Flask** – Lightweight Python web framework to handle webhook requests
* **Twilio** – For interacting with the WhatsApp API
* **gpiozero** – High-level interface to GPIO pins on the Raspberry Pi
* **request** – To fetch form data from HTTP POST requests
* **twilio.twiml.messaging\_response** – To generate WhatsApp-compatible responses
* **os / pip / system utilities** – To install and manage required packages

Libraries were installed using pip: pip install flask twilio gpiozero

# 

# Appendix 02

## A02.1. Project Proposal Form

The project proposal form was prepared and duly signed from our Faculty-in-Charge Dr. Biswaranjan Swain. The same is attached at the last of this report.

## 

## A02.2. Project Management

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Component** | **Individual Contributions in %** | | | **Total** |
| **Satyajit Panda**  **(Group Leader)** | **Satya Swarup Nayak** | **Shruti Sahoo** | Contribution |
|  | Planning | 35% | 35% | 30% | 100% |
|  | Background Research and Analysis | 40% | 40% | 20% | 100% |
|  | Hardware design | 60% | 30% | 10% | 100% |
|  | Software design | 45% | 45% | 10% | 100% |
|  | Testing | 50% | 30% | 20% | 100% |
|  | Final Assembling | 33.33% | 33.33% | 33.33% | 100% |
|  | Project report writing | 50% | 25% | 25% | 100% |
|  | Presentation | 10% | 50% | 40% | 100% |
|  | Logistics | 30% | 40% | 30% | 100% |

## 

## A02.3. Bill of Material

**Table 1. Component listing.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Component** | **Specification** | **Unit Cost** | **Quantity** | **Total** |
|  | Raspberry Pi | Zero 2W | 1560 | 1 | 1560 |
|  | power supply adapter –branded | 5v 3amp | 550 | 1 | 550 |
|  | USB to micro USB cable (1.5meter) with on/off switch power control for raspberry pi | 5V 3A | 185 | 1 | 185 |
|  | official raspberry pi microSD/SDHC memory card | 32GB | 395 | 1 | 395 |
|  | micro sd card reader |  | 50 | 1 | 50 |
|  | DC motor | 5V | 20 | 2 | 40 |
|  | Soldering iron | 25W | 130 | 1 | 130 |
|  | miscellaneous | LED, glue, resistor etc. | 200 | Multiple | 200 |
|  | Adapter with F connector | 5V 2A | 275 | 1 | 275 |
|  | Jumper wires | ( M to F) | 40 | 40 cables | 40 |
|  | Motor Driver | L298N | 120 | 1 | 120 |
|  | Bread board |  | 50 | 1 | 50 |
| **Grand Total** | | | | | 3595/- |

# Appendix 03

## A03.1. Data Sheets

Below are the key hardware components used in the project along with links to their official or reference datasheets:

1. **Raspberry Pi Zero 2 W**
   * <https://www.raspberrypi.com/products/raspberry-pi-zero-2-w/>
   * Features: Quad-core ARM Cortex-A53, 512MB RAM, 2.4GHz Wi-Fi, 40 GPIO pins
2. **L298N Motor Driver Module**
   * <https://components101.com/motor-driver/l298n-motor-driver-module>
   * Features: Dual H-Bridge driver, 5V-35V support, built-in heat sink and power LED
3. **5V DC Motor**
   * Generic motor used for fan simulation. Specs vary by model (usually 100–300 RPM, 5V–9V range).
4. **LED**
   * Standard 5mm LED with 220Ω resistor to limit current
5. **Micro SD Card (32GB Class 10)**
   * Used to load Raspberry Pi OS and store project files

These datasheets and specifications help in understanding voltage, pin configuration, and safe operation for all connected components.