In The Name of Allah

جامعة بوليتكنك فلسطين

Palestine Polytechnic University
College of Information Technology and Computer Engineering
Computer Systems Engineering

Graduation Project

Touch and Tap Cooking System

Submitted by:

Reem Ihdeib (201066)

Mais Dababsa (201113)

Mariam Sayareh (201009)

Supervisor:

Dr. Eng. Amal Al-Dweik

To fulfill the requirements for a bachelor's degree in the field of Computer Systems Engineering

February, 2025

Certification and Anti-Plagiarism Declaration

We, the undersigned, hereby declare that the graduation project titled "Touch and Tap Cooking System", completed under the supervision of Dr. Amal Al-Dweik, was prepared by:

Mariam Sayareh, Mais Dababsa, and Reem Ihdeib

in partial fulfillment of the requirements for the degree of Bachelor in Computer Systems Engineering.

We certify that:

- No part of this work has been reproduced illegally or contains any plagiarized content.
- All referenced materials have been properly cited to support and justify the research.
- We commit to maintaining academic integrity, avoiding plagiarism, cheating, or any other violations.
- We accept full responsibility for any consequences if a violation of this declaration is proven.

Date: 3 February 2025

Graduation Project Group Members:

Name: Mariam Sayareh	Signature: ____________________
Name: Mais Dababsa	Signature: ____________________
Name: Reem Ihdeib	Signature: ____________________

Abstract

Traditional cooking systems pose challenges such as limited temperature control and safety risks like burns and gas leaks. These issues are critical in households, where accidents may lead to severe injuries or fatalities. Additionally, requiring users to stay near the stove restricts mobility and efficiency. A safer, more advanced cooking solution is essential.

The "Touch and Tap Cooking System" integrates modern technology to address these concerns. Using a temperature sensor, camera, and ESP32 microcontroller, the system enables real-time monitoring and remote control. Safety is enhanced with automatic pan detection, child locks, and diagnostic alerts via a mobile app, minimizing accident risks.

Experimental validation confirms that system meets all objectives, offering remote cooking management, reducing supervision needs, and improving operational flexibility. The automatic shutdown mechanism further increases reliability, ensuring a safer and more efficient cooking environment.

الملخص

تعاني أنظمة الطهي التقليدية من قيود تشمل التحكم المحدود في درجة الحرارة ومخاطر السلامة مثل الحروق وتسرب الغاز، مما يشكل تهديدًا خطيرًا في البيئات المنزلية. كما تتطلب هذه الأنظمة إشرافًا مستمرًا، مما يحد من المرونة للمستخدمين. ولذلك، هناك حاجة إلى نظام طهي متقدم يعزز الأمان والكفاءة.

يقدم نظام الطبخ باللمس والنقرْ حلاً تقنيًا لمعالجة هذه التحديات عبر توظيف تقنيات الاستشعار والتحكم. يستخدم النظام مستشعر حرارة، كاميرا، ووحدة تحكم لمراقبة العمليات وإدارتها عن بُعد. تم تحسين الأمان بميزات مثل كشف الأواني تلقائيًا، وأقفال أمان للأطفال، وإشعارات بالأعطال عبر التطبيق، مما يقلل من مخاطر الحوادث.

أثبتت التجارب أن النظام يحقق جميع الأهداف، مما يسمح للمستخدمين بالتحكم بالطهي عن بُعد، ويقلل الحاجة للإشراف المستمر، ويعزز المرونة التشغيلية. كما يضيف الإيقاف التلقائي في المشروع مستوىً إضافيًا من الأمان، مما يجعل بيئة الطهي أكثر أمانًا وكفاءةً

Acknowledgment

Every great achievement is a mosaic of dedication, support, and belief. This project is no exception, and we are privileged to express our deepest gratitude to those who have been part of this journey.

To Dr. Amal Al-Dweik, your guidance was more than supervision; it was a source of enlightenment that shaped our path. Your insights, patience, and unwavering encouragement turned challenges into stepping stones, and for that, we are profoundly thankful.

To our families, the silent strength behind every late night, every hurdle, and every breakthrough. Your unwavering support, unconditional love, and endless belief in us have been our foundation. This journey was ours to walk, but it was your encouragement that gave us the courage to keep going.

To our friends, the ones who celebrated the small victories and stood by us during the toughest moments. Your words of motivation and shared laughter made this experience all the more memorable.

This project is not just a technical accomplishment—it is a reflection of the incredible people who stood with us, believed in us, and fueled our ambition. For that, we are endlessly grateful.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Touch and Tap Cooking System is a next-generation cooking solution designed to optimize kitchen convenience and safety. It integrates smart technology to allow chefs and home cooks to manage their cooking remotely via a simple mobile app, enhancing control with automated settings and real-time feedback. This is all achieved while ensuring utmost safety with advanced features like auto-pan detection and child locks. This chapter presents the problem statement, describes the general idea of the project, and discusses the system requirements.

1.2 Problem Statement

Due to their poor timing and restricted temperature control, traditional cooking systems are difficult for many cooking lovers to use and can produce disappointing results like burnt or undercooked food.

As for security and safety, statistics from 2021 indicate that, according to international reports, the death rate due to cooking gas leakage ranges between 1,000 to 5,000 deaths annually worldwide [4]. Some surveys conducted in local communities show that 80% to 90% of burn-related deaths occur at home[5]. Children and women usually suffer from burns in the kitchen due to falling or burning cooking utensils containing hot liquids or the explosion of cooking stoves.

Users also face time and location restrictions; good cooking requires being close to the stove, which limits mobility and disrupts daily routines. Therefore, the goal of this project is to create a remotely controlled cooking system to overcome these problems.

By using temperature sensors and remote cooking time management capabilities, users may have a safe and convenient cooking experience. The system, operated with a timer via a mobile application, enhances their everyday cooking experience by providing more comfort and freedom. It also allows for simple, usual manual operation, ensuring users can cook with or without the automated features.

1.3 Motivation and Importance

The proposed system will save time and enhance convenience. We have observed that most chefs spend long hours in the kitchen monitoring the cooking process, preventing them from performing other tasks due to inflexible time constraints. This observation served as the inspiration for our project, highlighting its importance in providing comfort and luxury. It allows users to control cooking operations from anywhere and at any time, adding flexibility to the time required to complete cooking tasks.

Additionally, a significant percentage of mothers today are working, and young children are often left in the care of their older siblings. Since the kitchen is a favorite place for children, this can pose potential risks. The proposed system provides an effective solution to this issue by helping to reduce risks and ensure the safety of children through remote monitoring.

1.4 Aims and Objectives

Touch and Tap Cooking System is designed to be a multifunctional and safe addition to any kitchen, providing efficiency and convenience of usage. The main aim of this device is to simplify the cooking process by offering a clever, flexible cooking system that maximizes efficiency, assures safety, and improves the overall cooking experience for both professional chefs and home cooks. The following objectives align with the previous aim:

- Developing an intuitive user interface that includes a mobile application and LCD display for system configuration.

- Offering a variety of cooking modes, both manual and remote, to accommodate the user's presence and location.

- Providing pre-programmed cooking schedules with varying power levels and an automated shut-off feature for convenience and security.
- Equipping the system with safety features including an auto-pan detection feature, a diagnostic error message system, and a child safety lock to prevent accidents and ensure safe operation.

1.5 System Description

Touch and Tap Cooking System is an easy way to add an extra burner to a small residential kitchen or a busy commercial one. The system will allow users to control their cooking operations remotely at any time, provide a camera that monitors the cooking environment, and offer feedback technology for the stove to enhance security, improve quality, and increase customer satisfaction by eliminating the need for hot coils or open flames.

All system settings can be adjusted remotely using the mobile application. The system operates in two modes:

- **Manual Mode:** Used when the user is near the system.
- **Remote Mode:** Used when the user is in another room or outside the house.

Figure 1.1: General View of the System

The system is equipped with:

- A child safety lock function to reduce risks by automatically shutting down the stove and turning on the buzzer if presence is detected near the device, ensuring enhanced safety.
- Timer settings to allow precise control over cooking durations for improved convenience.

- Temperature level selection functionality, offering four distinct settings: Warm, Low, Medium, and High, to accommodate various cooking needs.
- A diagnostic error message system in case of accidental failures, children approaching, or other safety concerns.
- An auto-pan detection unit that automatically shuts down the system immediately if no cookware is detected and turns it on again when the pan is placed correctly. If misplaced, the buzzer will ring, and the stove will turn off.
- When the stove is turned off due to the presence of a person, the system activates a pause feature that temporarily halts cooking. Cooking will automatically resume after a 2-minute delay, continuing with the previously configured settings.
- A camera that provides the user with a snapshot in case of a person's presence.
- A small and selected list of recipes stored in a database to help automatically adjust the system settings.

1.6 System Requirements

The system's functional and non-functional requirements are described as follows:

1.6.1 Functional Requirements

- The system will allow users to control cooking operations remotely when away using the mobile application, while also providing a manual mode for local control.
- Users will receive alerts when a preset cooking program is completed.
- The system will enable scheduling of cooking times for various dishes with adjustable settings via Firebase using the mobile application.
- Sensors will provide feedback on the cooking process, including temperature monitoring, motion detection, and limit switches.

- The system will alert users in real-time about potential safety hazards or malfunctions, with automatic shutdown in critical situations.
- A user-friendly mobile application interface will allow setup and monitoring of the system.

1.6.2 Non-Functional Requirements

- Accuracy: The system will precisely regulate stove temperature to achieve consistent cooking results.
- Cost-Effectiveness: The system will be designed with affordable components to ensure accessibility without compromising reliability.
- Energy Efficiency: The system will prevent unnecessary power consumption by automatically shutting off when cooking is complete and when pan detection confirms the cookware is completely removed from the stove.
- Usability: The system will have an intuitive user interface, ensuring ease of use for individuals with varying levels of technical proficiency.
- Scalability: Future upgrades will be supported, allowing integration with additional sensors, smart home ecosystems, and advanced cooking automation.
- Reliability: The system will operate continuously and respond in real time to sensor inputs, minimizing failures and improving operational stability.
- Safety Compliance: The system will adhere to safety standards by including auto shutoff mechanisms, child safety locks, and hazard detection features to reduce risks.

1.7 Constraints

- A continuous power source is required for operation.
- A Wi-Fi network must be available, and an internet connection is necessary for controlling in remote mode.
- The camera must be manually and regularly cleaned by the user.
- The user should exercise caution when handling water near the device.
- The load placed on the stove must not exceed 3 kg.
- The apparatus should have a maximum diameter of 17 cm and a minimum of 7 cm.

1.8 Project Schedule

The project schedule in **Table 1.1** outlines the timeline for developing the Touch and Tap Cooking System, ensuring systematic progress over two semesters.

Table 1.1: Project schedule in the first and second semester

Tasks	**Second Semester**			**First Semester**			
Weeks	1-4	5-10	11-15	1-5	6-9	10-13	14-15
Project Initialization and Planning							
System Design and Component Selection							
Hardware Development and Software Development							
System Integration and Testing							
User Interface and Experience Design							
Pilot Testing and Validation							
Documentation and Final Report Preparation							
Project Presentation and Delivery							

1.9 Report Outline

This report begins with Chapter 1, which introduces the Touch and Tap Cooking System, detailing the problem it addresses, its objectives, system requirements, constraints, and schedule. Chapter 2 provides the theoretical background, including key concepts, a review of related literature, and how this work differs from existing solutions. Chapter 3 focuses on system design, covering hardware and software components, design alternatives, and implementation methodologies. Chapter 4 discusses implementation, while Chapter 5 is for testing, validation, analyzing results, and evaluating system performance. Finally, Chapter 6 concludes the report with key findings and suggestions for future work, followed by references and appendices.

Chapter 2

Background

2.1 Preface

This chapter introduces the basic concepts and literature relevant to the project. It provides an overview of the theoretical background, reviews similar projects, highlights their main ideas, and concludes with a summary encapsulating the main points discussed.

2.2 Theoretical Background

Modern cooking technologies are continuously evolving, focusing on energy efficiency, safety, and ease of control. Different cooking methods require specific conditions and technologies to function effectively. **Table 2.1** compares various cooking technologies, their safety features, control mechanisms, and the requirements necessary for their operation.

is choosen Touch Controls and Smart Interfaces for its advanced energy efficiency and high level of control. Its integration with sensors and software enhances precision and usability, making it a cutting-edge and reliable cooking technology.

Table 2.1: Comparison of Cooking Technologies, Safety Features, and Requirements

Technology	Efficiency	Safety	Control	Requirements
Induction Technology	Very high, as heat is generated only within the pan	Surface remains cool, reducing burn risks	Very precise temperature control	Requires compatible induction cookware and continuous power supply
Ceramic Radiant Heating	Moderate due to surface heat loss	Hot surface remains after cooking, posing a burn risk	Reasonably precise temperature control	Requires heat-resistant glass or ceramic cooking surfaces
Halogen Heating	Moderate, as lamps are powerful but lose some heat	Can get very hot, increasing the burn risk	Quick temperature changes but less precise	Requires a high-power halogen heating element and durable cookware
Touch Controls and Smart Interfaces	Improves user control over energy use	Includes features like child locks and automatic shut-off	Highly accurate settings with remote control options	Requires sensors, software integration, and a smart interface

We chose Touch Controls and Smart Interfaces for its advanced energy efficiency and high level of control. Its integration with sensors and software enhances precision and usability, making it a cutting-edge and reliable cooking technology.

2.3 Literature Review

Several systems and projects relate to the system. The following sections introduce key examples:

2.3.1 Smart Stove Functional Specification [12]

The Thermopix Smart Stove provides convenience and efficiency by utilizing induction heating technology for rapid and efficient heating of magnetic pots. Features include advanced temperature control, temperature sensor units, and a high-definition LCD screen. Challenges include reliance on specific pots and the need for continuous power supply. Our project shares temperature control and tool detection features but adds functionalities like time control, remote access, and a built-in camera for monitoring.

2.3.2 A Smart Cooking Device for Assisting Cognitively Impaired Users [13]

This device employs advanced sensors for temperature, humidity, and motion tracking, alongside RFID and computer vision systems for real-time data capture. It uses a touchscreen or Android tablet for user interaction and features a comprehensive recipe database. While it shares similarities like remote control, recipe selection, and automatic shutdown, our project differs with added capabilities such as real-time camera monitoring.

2.3.3 Design of a Smart Electric Cooking Stove [14]

This design enhances safety and energy efficiency using an Arduino Uno microcontroller and sensors for pot detection. A relay switch shuts off the stove if unattended for five minutes. Although it shares pot detection functionality, our project provides additional features such as temperature and time specification, remote control, and integrated recipes, along with a camera for enhanced usability.

2.4 Summary

A comparison of our project with similar systems is presented in **Table 2.2**, highlighting the main features, advantages, and limitations of each system.

Table 2.2: Differences Between Our Project and Similar Projects

Specification	**Thermopix Smart Stove [12]**	**A Smart Cooking Device for Assisting Cognitively Impaired Users [13]**	**Smart Electric Cooking Stove [14]**	**Touch and Tap Cooking System**
Heating Elements	Induction heating	Induction heating	Traditional heating elements	Traditional electric hot plate
Temperature Control	Advanced digital control	Automated system ensuring safe cooking	Manual control using Arduino	Manual and remote control via mobile app
LCD Display	Integrated LCD display	User-friendly LCD display providing clear feedback	Not specified	LCD display for stove status
Feedback	Visual and audible alerts	Visual and auditory feedback for users	Automatic feedback through microcontroller	Integrated feedback with real-time cooking updates and safety notifications
Pan Detection	Automatic cookware detection	Built-in pan detection	Spring-loaded pot contactor	Auto-pan detection with automatic shutdown
Safety Features	Overheat protection, auto shut-off	Auto shut-off, overheating prevention, and malfunction alerts	Auto turn-off if no pot is detected	Child safety locks, automatic shutdown, diagnostic error messaging

Specification	**Thermopix Smart Stove**	**Cognitively Impaired Device**	**Smart Electric Stove**	**Touch & Tap Cooking System**
Mobile Application	No dedicated app	Remote control for caregivers	No dedicated app	Remote control and monitoring via mobile app
System Control (Remote)	No remote control	Caregivers or family members can control remotely	No remote control	Remote operation mode via mobile app
System Control (Local)	Touch-sensitive buttons integrated with LCD display	Large buttons and safeguards to prevent accidental changes	Manual control via microcontroller	Manual mode for local operation
Camera	No	No	No	Yes
Recipe Database	No	Yes	No	Yes
Challenges	Requires magnetic pots, continuous power supply	Sensor accuracy affected by environment	Sensor contact and environmental impact issues	Ensuring reliable remote operation and sensor accuracy
Similarities with Our Project	Temperature control, cooking tool detection, LCD features	Remote control, recipe selection, automatic shutdown	Auto-pot detection, safety mechanisms	N/A (Base System)
Differences from Our Project	Lacks remote control, camera	No camera, lacks time control	No remote control, no camera	Includes all listed features

Chapter 3

System Design

3.1 Preface

This chapter presents the defined software and hardware components, explores design alternatives, and conceptualizes the system architecture.

3.2 System Components

The system consists of both hardware and software components.

3.2.1 Hardware Components

The hardware components include:

Microcontroller

A microcontroller is needed as it acts as the central brain of the system, controlling all components, processing sensor data, enabling automation, and ensuring real-time adjustments for optimal cooking. It also handles communication with external devices, such as mobile apps, and provides a cost-effective, energy-efficient solution for smart functionality.

Table 3.1 compares the Arduino ATmega328P, ESP32, and Raspberry Pi4 microcontrollers.

Table 3.1: Microcontroller Comparison

Feature	**Arduino ATmega328P** [3]	**ESP32** [8]	**RaspberryPi 4** [7]
Speed	16 MHz	240 MHz	1.8 GHz
Memory	256KB Flash, 8KB SRAM	520KB SRAM, 16MB Flash	1GB-8GB LPDDR4
Wi-Fi/Bluetooth	No	Yes	Yes
GPIO Pins	54	34	40
Cost	$25-$40	$5-$15	$35-$75

ESP32 is selected due to its built-in Wi-Fi, suitable memory size, and ample interfacing pins.

Stove

The stove is an essential component in the project. Table 3.2 compares electric stoves and induction cooktops.

Table 3.2: Option Stove Comparison

Type	**Electric Stove**	**Induction Cooktop**
Heating Method	Uses resistance heating elements to transfer heat from the surface to cookware.	Directly heats cookware via electromagnetic induction, requiring magnetic materials.
Technology Complexity	Simple design, relying on basic wiring, insulation, and safety features.	Advanced control, heating only the cookware for precise temperature adjustments.
Compatibility with Cookware	Works with various cookware types, including aluminum and glass.	Requires magnetic cookware but ensures a safer, cooler surface with auto shutoff.
Availablility	Available	Not available
Cost	More affordable	Generally more expensive due to advanced technology and efficiency.

The hob shown in **Figure 3.1** is chosen because it is low-cost, made of stainless steel, has fast heating (1600 W), and is compatible with cookware up to 17 cm. It also features an LED indicator that shows when the stove is on and a rotary encoder with five temperature levels, providing precise control.

Figure 3.1: Selected Stove for the Project

Camera

A camera is included to capture images when motion is detected near the stove. An ESP32-CAM module, shown in **Figure 3.2**, is used for this purpose.

Figure 3.2: ESP32-CAM Module Used for Motion Detection

Wi-Fi Module

Wi-Fi connectivity allows remote control via a smartphone app, enabling users to manage cooking operations from a distance.

Buzzer

A buzzer notifies the user when cooking is complete or alerts them if a hazard is detected.

Power Supply

An AC/DC power supply converts wall outlet voltage to the required level for the system components.

Sensors

The system uses different sensors :

- **Temperature Sensor:** A thermocouple was selected for its wide temperature range and fast response time, making it suitable for precise cooking control. As shown in **Table 3.3**, it outperforms RTDs in sensitivity and cost-effectiveness. .

Table 3.3: Comparison of Temperature Sensor Options

Feature	**Thermocouple [11]**	**Resistance Temperature Detector (RTD) [11]**
Temperature Range	-200°C to 2000°C	-200°C to 600°C
Response Time	0.1 to 10 seconds	0.5 to 5 seconds
Self-Heating	Negligible	Can self-heat
Sensitivity	More sensitive	Less sensitive
Availability	Online	Online
Cost	$10 (up to 600°C)	$14 (up to 500°C)

Figure 3.3 illustrates the selected thermocouple used in the system

Figure 3.3: Thermocouple Sensor

- **Motion Sensor:** An ultrasonic sensor was selected over PIR sensors due to its stable detection range and resistance to environmental changes, as shown in **Table 3.4**.

Table 3.4: Comparison of Motion Sensor Options

Feature	**PIR Motion Sensor [1]**	**Ultrasonic Motion Sensor [1]**
Detection Principle	Detects infrared radiation emitted by objects	Emits high-frequency sound waves and measures reflection
Detection Range	4.5 to 9 meters	Can cover a larger area
Detection Method	Responds to heat and movement of warm objects	Measures time for sound waves to bounce off objects
Sensitivity	Highly sensitive to heat and movement	Less sensitive to small movements but detects solid objects
Response Time	Fast but affected by environmental conditions	More stable but with slight delay
Cost	Lower cost	Slightly more expensive
Environmental Impact	Affected by temperature changes	Less affected by environmental factors

- **Pan Detection:** Limit switches were chosen over weight, infrared, and CNN-based sensors due to their simplicity, reliability, and low cost, according to **Table 3.5**.

Table 3.5: Comparison of Sensors for Detecting Pots on an Electric Stove

Feature	**Weight Sensors**	**Limit Switches**	**Infrared Sensors**	**Convolutional Neural Network (CNN)**
Detection Principle	Measures weight/pressure of the pot	Mechanical contact triggers the switch	Detects heat radiation from cookware	Uses image recognition to detect pot placement
Working Mechanism	Uses strain gauge technology	Activates when cookware is in contact	Detects changes in infrared radiation	Trained on dataset images to recognize patterns
Accuracy	High	High	High	High
Sensitivity	Detects small weight changes	Moderate, adjustable based on force applied	Detects presence but not object type	Can differentiate between objects
Response Time	Fast	Fast	Fast	Depends on processing power
Cost	Moderate to high	Low to moderate	Moderate to low	High

Voltage Regulator

A voltage regulator (L7805C) ensures stable voltage for system operation.

Human-Machine Interface (HMI)

The system provides an OLED display for user feedback, including temperature levels and pan status.

Relays

Relays control the power state of the stove, allowing both manual and remote operation.

3.2.2 Software Components

The system contains the following software components:

Software Platforms and Database

Efficient data management is essential for real-time monitoring and remote control capabilities in the system.**Table 3.6** compares different database options considered for the system.

Table 3.6: Comparison of Database Platforms

Feature	**Firebase [10]**	**Supabase [10]**	**AWS Amplify [10]**
Type	NoSQL (JSON-based)	SQL (PostgreSQL)	NoSQL (DynamoDB)
Ease of Use	Beginner-friendly with real-time capabilities	Requires SQL knowledge, more structured	Complex, requires AWS expertise
Integration	Optimized for Google services and mobile apps	Open-source, flexible with third-party tools	Deep integration with AWS ecosystem
Scalability	Highly scalable, designed for large real-time applications	Supports complex queries but requires optimization for large-scale use	Enterprise-grade, scalable but requires configuration
Offline Support	Yes, allows local data caching	Limited built-in offline support	Requires additional configuration for offline usage
Security	Built-in authentication (Google, Email, etc.)	Role-based access control, requires setup	Strong security with AWS IAM integration

After evaluating these options, Firebase was chosen as the database platform due to its real-time synchronization, ease of integration with mobile applications, and built-in authentication features.

Mobile Application

The system requires a mobile application to allow users to remotely control the stove, monitor real-time status, and receive alerts.**Table 3.7** presents a comparison of the mobile development frameworks considered for this project.

Table 3.7: Comparison of Mobile Development Frameworks

Feature	**React Native** [9]	**Flutter** [6]	**Android SDK** [2]
Language	JavaScript	Dart	Java/Kotlin
Platform Support	Cross-platform (Android, iOS)	Cross-platform (Android, iOS)	Android only
UI Performance	Uses native components, may require bridging for some features	Fast performance with custom rendering engine (Skia)	Best performance, as it runs natively
Development Speed	Fast, but requires extra work for native modules	Faster due to **hot reload** and single UI codebase	Slower due to platform-specific development
Community	Large community, backed by Meta	Growing community, backed by Google	Strong Android support, but limited to Android ecosystem
Integration	Well-supported but sometimes requires third-party libraries for native access	Supports native features and provides high UI flexibility	Full access to Android features

After evaluating these frameworks, Flutter was chosen for mobile development due to its fast development capabilities, cross-platform support, and efficient UI rendering.

3.3 System Diagrams

To visualize the architecture, workflow, and connectivity of the syste*, various diagrams are provided. These diagrams illustrate the interaction between hardware and software components, data flow, and overall system functionality.

3.3.1 Block Diagram

The block diagram, shown in **Figure 3.4**, represents the high-level system architecture, depicting the connection between key hardware components and how they communicate.

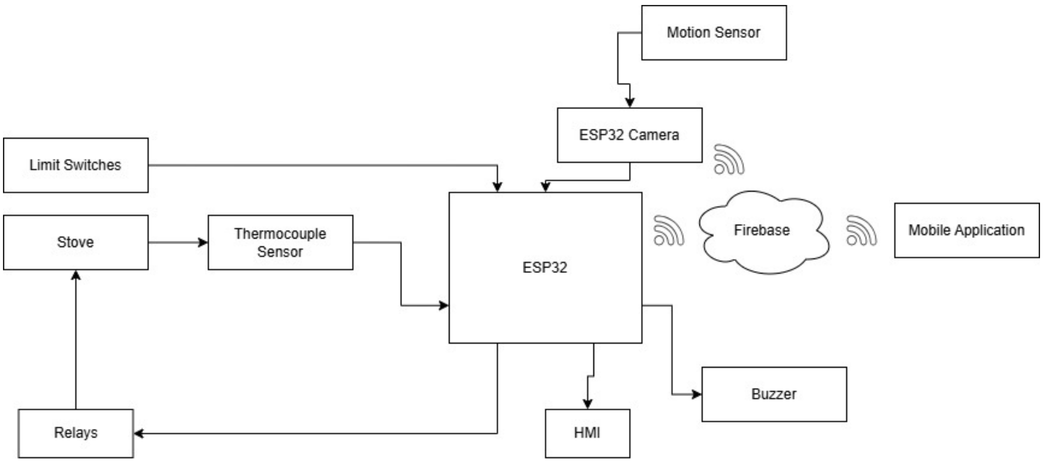

Figure 3.4: Block Diagram of the System

3.3.2 Conceptual Diagram

Figure 3.5 illustrates the conceptual overview of the system, showing the flow of information between sensors, the microcontroller (ESP32) and the mobile application.

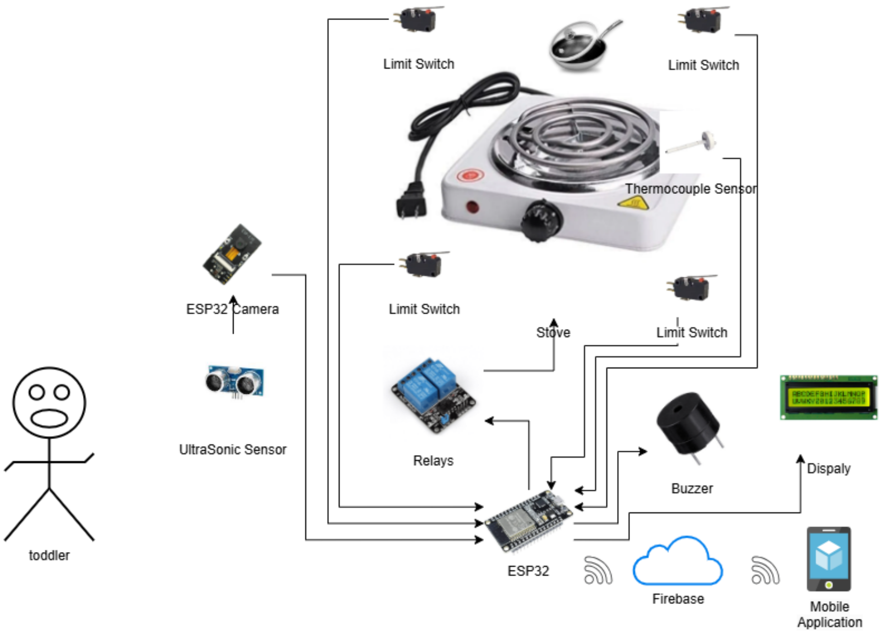

Figure 3.5: Conceptual Diagram

3.4 Conceptual System Description

The Touch and Tap Cooking System is designed to enhance safety, convenience, and efficiency in cooking.

3.4.1 How the System Works

- User Control: The user can turn the stove on/off and adjust temperature via the mobile application (Flutter-based). Commands are sent to the Firebase database, which the ESP32 retrieves in real-time.
- Temperature Regulation: A thermocouple sensor continuously monitors the stove's temperature. If the temperature exceeds a predefined threshold, the ESP32 triggers the relay to power off.
- Pan Detection: Limit switches verify if cookware is placed on the stove. If no pan is detected, the system automatically turns the stove off for efficiency.
- Motion Detection & Safety Alerts: An ultrasonic motion sensor detects movement near the stove. If unauthorized motion is detected while the

stove is on, an alert is sent to the mobile app, and a buzzer sounds to notify the user.

- Visual Monitoring: The ESP32-CAM module captures images if motion using ultrasonic sensor is detected. These images can be viewed in the mobile application.
- Power Control: A relay module controls the stove's power, allowing remote ON/OFF switching. The L7805C voltage regulator ensures stable power for all components.

3.4.2 System Communication and Data Flow

- The ESP32 fetches data from Firebase, updating the stove's status based on user input.
- Sensor readings are processed to trigger automated actions.
- The mobile app continuously syncs with Firebase to display real-time cooking status.

3.5 Software illustration and Explanation

The system is designed to provide users with an intuitive and interactive way to control their cooking process. The system integrates Firebase for real-time data synchronization and remote control.

3.5.1 Algorithm

The next figure 3.6 shows the algorithm of the software:

Figure 3.6: Software Algorithm

3.5.2 Pseudocode

Here is the presentation of the pseudocode for the software:

```
BEGIN
  Initialize Firebase
  Setup Firebase Messaging and Notifications
  Initialize App UI

  FUNCTION LoadSplashScreen()
    DISPLAY splash animation
    RETRIEVE Firebase token
    SET global.hasOperations from Firebase
    SET global.status from Firebase
    NAVIGATE TO HomeScreen

  FUNCTION HomeScreen()
    DISPLAY category options: ["Foods", "Add New", "Smart Set", "Operations"]
    IF User Selects "Foods"
      NAVIGATE TO FoodSelectionScreen
    ELSE IF User Selects "Add New"
      NAVIGATE TO AddNewFoodScreen
    ELSE IF User Selects "Smart Set"
      NAVIGATE TO SmartSetScreen
    ELSE IF User Selects "Operations"
      NAVIGATE TO OperationScreen

  FUNCTION FoodSelectionScreen()
    RETRIEVE food list from Firebase
    DISPLAY list of saved foods
    IF User Selects a Food Item
      NAVIGATE TO SmartSetScreen WITH Selected Food
  FUNCTION AddNewFoodScreen()
    INPUT food name
    SET temperature using slider
    SET cooking time using timer
    SAVE food data to Firebase
    DISPLAY success message
    RETURN TO HomeScreen
  FUNCTION SmartSetScreen(Food)
    DISPLAY temperature and time controls
    IF User Starts Cooking
```

```
      UPDATE Firebase: Set hasOperation = TRUE
      NAVIGATE TO OperationScreen
  FUNCTION OperationScreen()
    RETRIEVE real-time stove status from Firebase
    IF Motion Detected
      ALERT User and SHOW Image
    IF User Presses Stop
      SET hasOperation = FALSE in Firebase
      RETURN TO HomeScreen
    ELSE IF User Presses Pause
      SET isPaused = TRUE in Firebase
    ELSE IF User Presses Continue
      SET isPaused = FALSE in Firebase
END
```

3.5.3 Function Explanations

Here is the part, which is a detailed breakdown of each function of the pseudocode involved in the process.

1. LoadSplashScreen()

This function initializes the application by displaying the splash screen. It retrieves the Firebase token, updates global variables such as the operational status of the system, and then navigates to the Home Screen.

2. HomeScreen()

This function acts as the main menu, providing users with different category options: Foods, Add New, Smart Set, and Operations. Based on the user's selection, the app navigates to the corresponding screen.

3. AddNewFoodScreen()

This function allows users to input a new food item, specifying its name, temperature, and cooking time. The entered details are saved in Firebase, and a success message is displayed before the app returns to the Home Screen.

4. FoodSelectionScreen()

The function retrieves a list of saved food items from Firebase and displays them. When a user selects a food item, the app navigates to the Smart Set screen, carrying the selected food's details.

5. SmartSetScreen(Food)

This function enables users to set temperature and time for cooking. If the user initiates the cooking process, the system updates Firebase to reflect that an operation has started and navigates to the Operation Screen.

6. OperationScreen()

The Operation Screen continuously monitors the stove's status using real-time Firebase updates. If motion is detected, an alert is triggered with an image display. Users have options to pause, continue, or stop the cooking process, with Firebase updating accordingly to ensure real-time control.

This structured implementation ensures that users can manage their cooking remotely, with real-time synchronization, automated alerts, and interactive controls for a seamless cooking experience.

3.6 System Sequence Diagrams

This section presents the sequence diagrams that illustrate the interactions between the system components during different operational scenarios.

Operational Mode

Figure 3.7 illustrates the operational mode. The user sets cooking parameters via the mobile app, and the ESP initializes the stove. Motion detection triggers the camera, sending an image to the user for verification. If no response is received within two minutes, the system shuts down automatically for safety.

Operational Mode for Pan Detection

Figure 3.8 illustrates the pan detection process. The system verifies pan placement using four limit switches. If correctly positioned, the stove activates; otherwise, the user receives a notification. If partial placement is detected, the system reassesses before activation. A buzzer may alert the user in case of misplacement.

Startup Mode

Figure 3.9 shows the system startup. The ESP initializes, checks Wi-Fi, and connects. The stove runs in manual or remote mode. The ultrasonic sensor detects motion, activating the camera if needed. If no motion is detected, the system stays in standby.

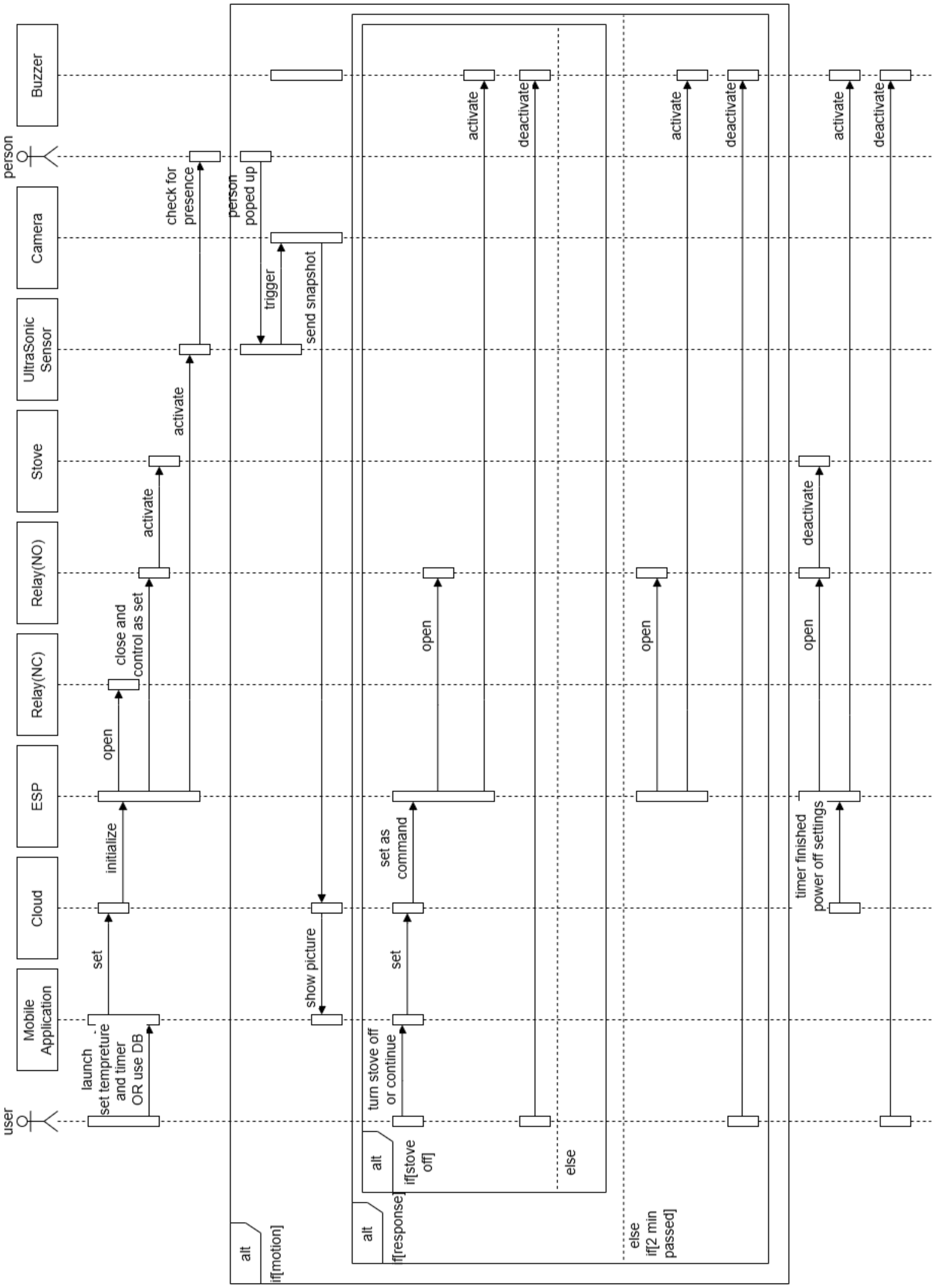

Figure 3.7: Operational Mode Sequence Diagram

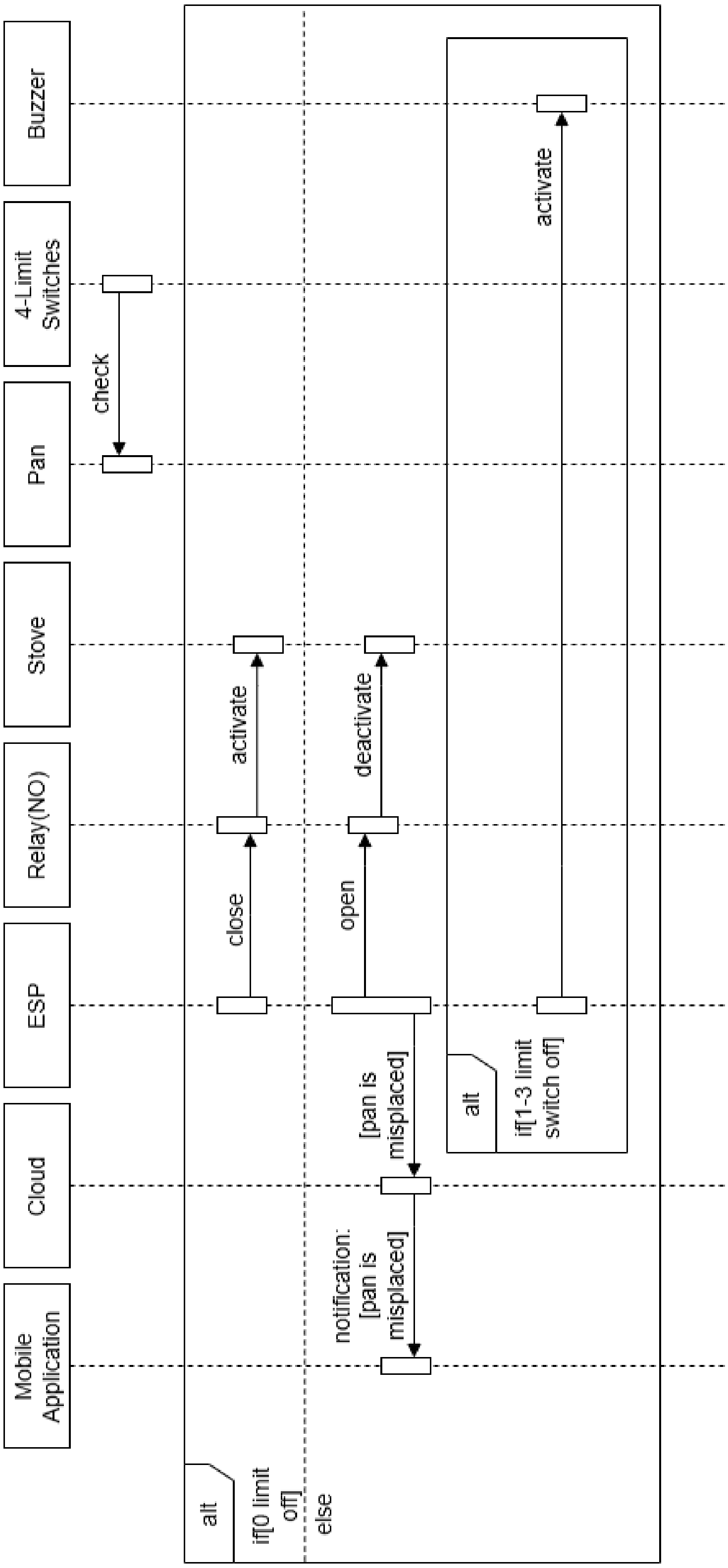

Figure 3.8: Pan Detection Sequence Diagram

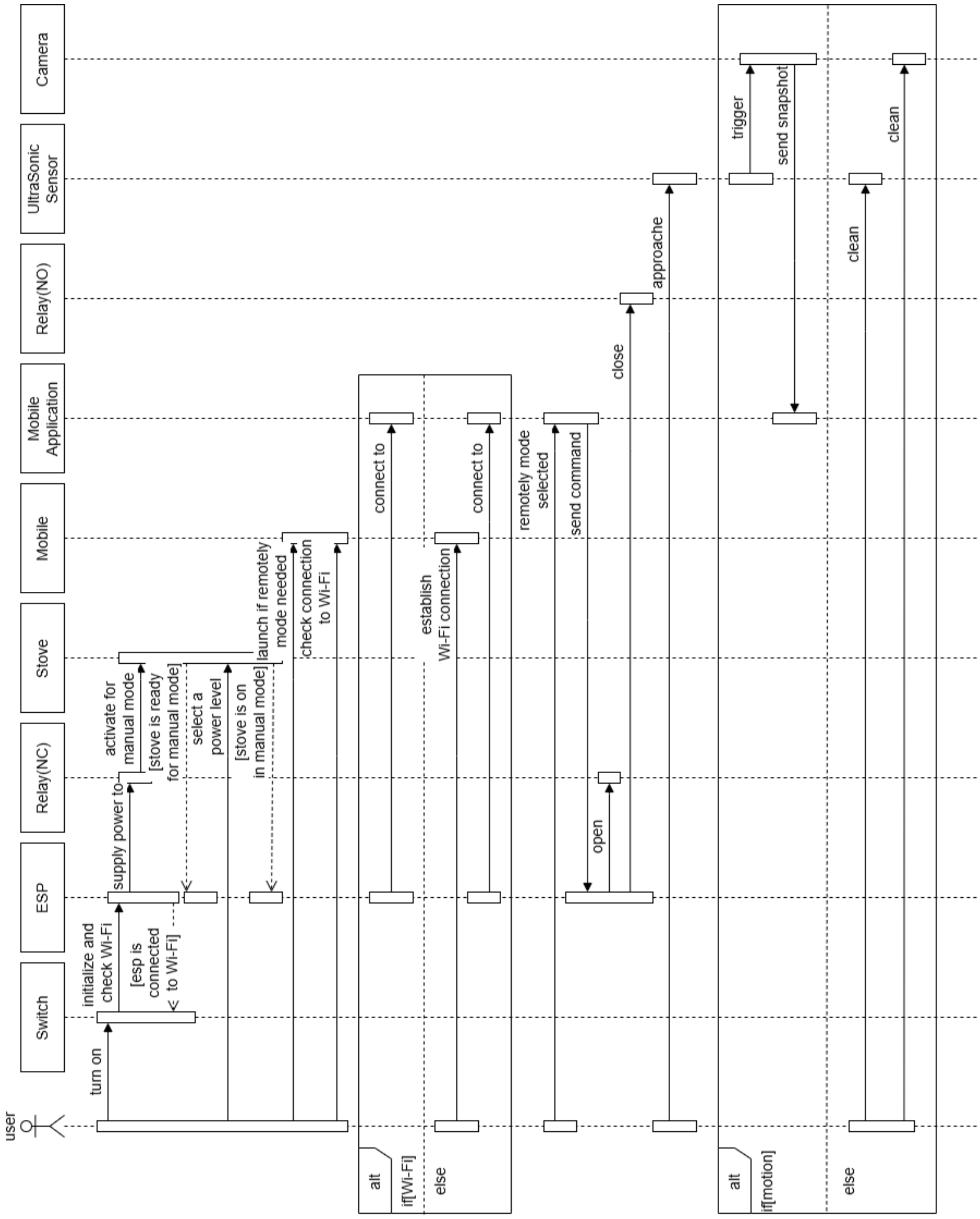

Figure 3.9: Startup Sequence Diagram

3.6.1 Schematic Diagram

Figure 3.10 illustrates the system's hardware connections, including the ESP32 microcontroller, sensors, relays, and stove. It shows how components interact to ensure safe and efficient cooking operations.

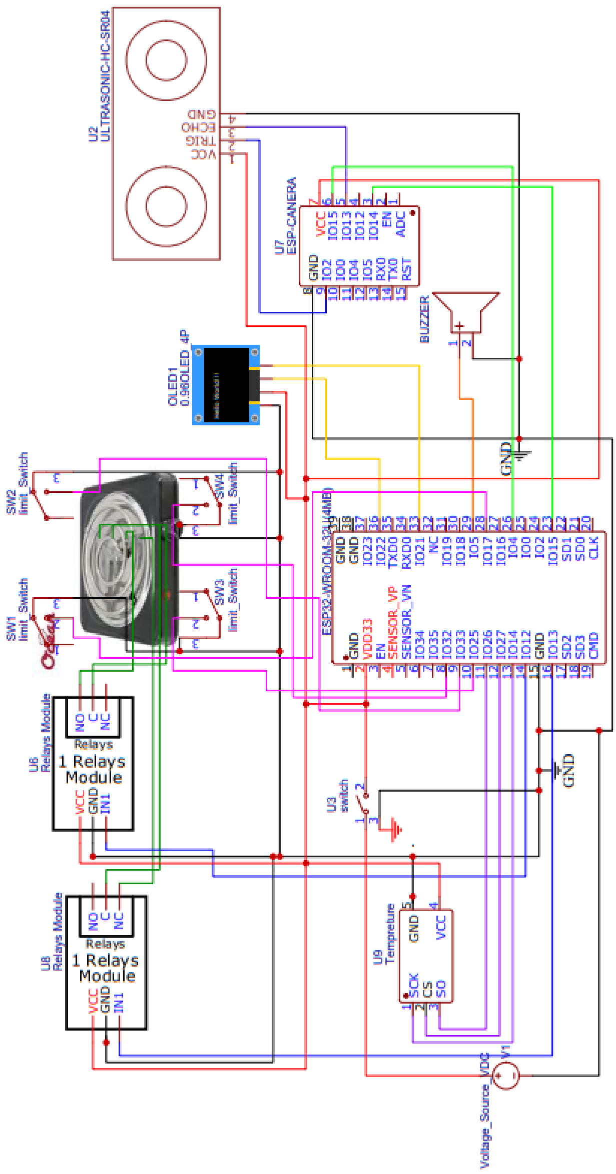

Figure 3.10: System Schematic Diagram

3.7 Summary

This chapter covered the hardware and software components, including sensor selection, database management, and mobile development choices. It explained the system's operation through sequence diagrams and concluded with the schematic diagram, providing an overview of the system's integration and functionality.

Chapter 4

Implementation of the System

4.1 Preface

This chapter covers the hardware and software implementation, along with the challenges encountered and their solutions.

4.2 Hardware Implementation

The hardware components were carefully selected and integrated to ensure reliable operation. The main hardware elements include:

4.2.1 Custom-Designed Wooden Stand for Electric Stove

This custom-designed wooden stand, shown in Figure 4.1, supports the electric stove with a 20 cm × 20 cm base with a 25 cm × 25 cm top surface for limit switches. The top panel includes a circular cutout for airflow and wiring, while the middle layer (5 cm) acts as a protective barrier with wiring holes. The bottom compartment houses electronic components, featuring a 15 cm for hand access.

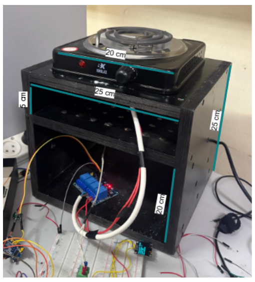

Figure 4.1: Custom-Designed Wooden Stand

4.2.2 Relay

Figure **Figure 4.2**, illustrates the core hardware components of the Touch and Tap Cooking System, highlighting the functions of the relays in different modes. In manual mode, the normally closed (NC) relay allows temperature adjustments via the rotary encoder. In remote mode, the NC relay opens and the normally open (NO) relay closes, enabling remote temperature control. The NO relay manages heating levels based on commands from the system's software, ensuring precise remote control. This dual relay set-up ensures smooth switching between modes, enhancing safety and convenience.

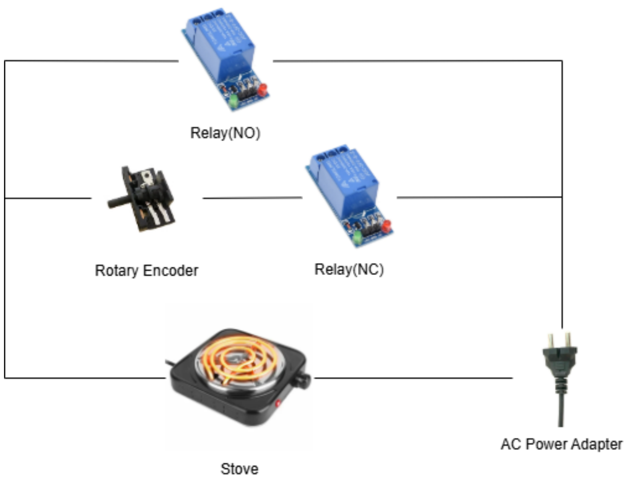

Figure 4.2: Relays and Stove Connections

As illustrated in the previous figure, Figure 4.3 provides a detailed view of the stove's internal setup, including the thermocouple connection for precise temperature monitoring and control, ensuring efficient and safe cooking.

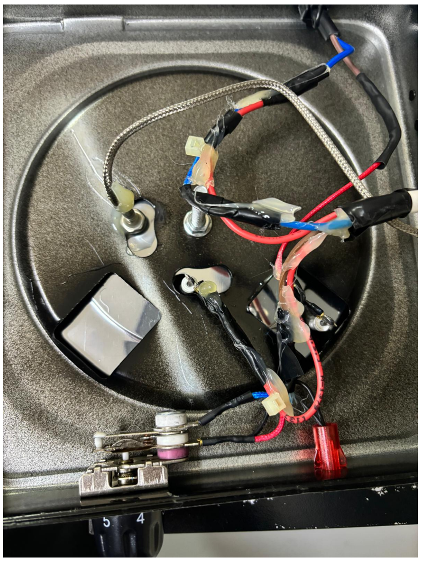

Figure 4.3: The Real Stove Connection

4.2.3 Sensors

The project includes temperature, motion, and pan detection sensors, ensuring safety and automation.

Temperature Sensor

A thermocouple sensor was installed to monitor the stove's temperature in real time. It is strategically placed to directly monitor the heat of the burner, ensuring accurate temperature readings, as shown in Figure 4.4.

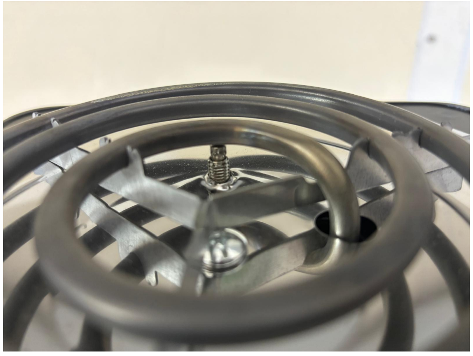

Figure 4.4: Thermocouple Sensor Installation

Motion Detection and Alert System with ESP32-CAM

The ultrasonic sensor is connected to the ESP32-CAM to detect motion, the system captures an image and sends it to the user via a mobile app. The camera communicates with the ESP32 via hardware serial, and a buzzer is used for alerts. As shown in 4.5.

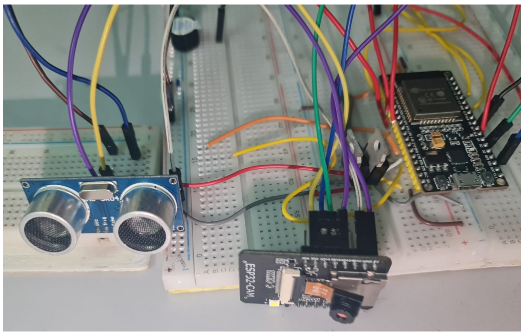

Figure 4.5: Ultrasonic Sensor and Alert System Connections with ESP32-CAM

Pan Detection System

Four limit switches were mounted to detect pan placement. If the pan is removed or misplaced, the system notifies the user via the mobile app and turns off the stove for safety. As shown in Figure 4.6.

Figure 4.6: Limit Switches Installation

4.2.4 Human-Machine Interface (HMI) Design (OLED)

An OLED display is integrated with the ESP32 to serve as a user interface, providing real-time system updates. It displays temperature variations, pan detection status, and other system changes. This setup is detailed in the schematic diagram in the design chapter and illustrated in Figure 4.7.

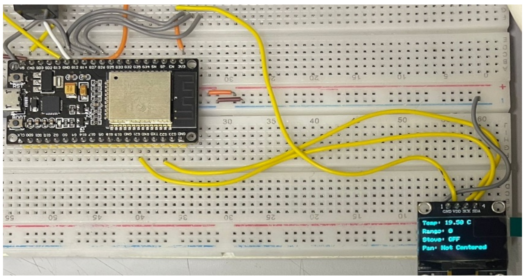

Figure 4.7: OLED Display Connected to ESP32

4.3 Software Implementation

The software components ensure seamless interaction between the hardware and the user. Firebase is used as the backend for real-time data exchange, while the mobile application, developed using Flutter, provides an intuitive interface.

4.3.1 Mobile Application

The Flutter-based mobile app allows users to:

- The Flutter-based Android app provides an intuitive interface for managing stove operations and real-time monitoring.
- The "Smart Set" and "Operations Screen" allow users to track stove status, adjust temperature dynamically, and receive real-time feedback.
- Users can add or remove pre-programmed cooking modes, simplifying the process while allowing flexible adjustments during operation.
- In critical events, such as a safety buzzer alert, users can stop the buzzer directly via the application, ensuring control and convenience.

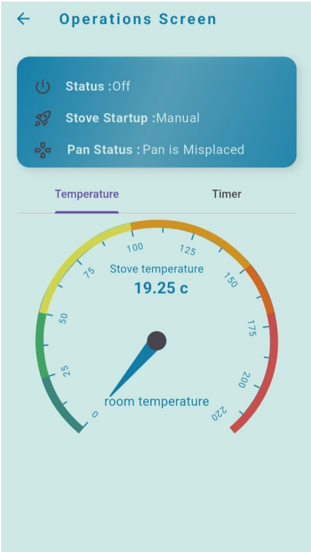

Figure 4.8: Operations Screen

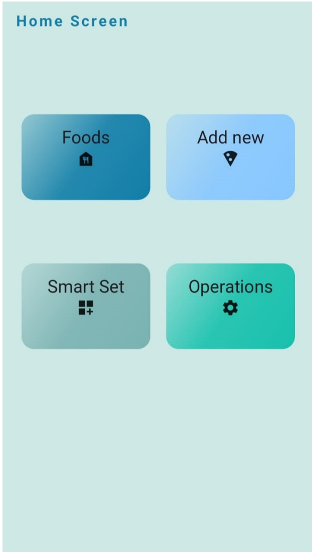

Figure 4.9: Home Screen

Figure 4.10: Smart Set Screen

- The "Operations Screen" in **Figure 4.8** displays real-time stove status, provides live feedback, and enables dynamic temperature adjustments in remote mode.
- The "Settings Screen" in **Figure 4.9** offers a simple and user-friendly interface to configure system settings.
- The "Smart Set Screen" in **Figure 4.10** allows users to preset temperature and cooking time for automation and convenience.

4.3.2 Firebase Real-Time Database

Firebase manages real-time data exchange between the ESP32 microcontroller and the mobile application, ensuring smooth system operation. It plays a crucial role in handling various system functions, as detailed below:

- Real-time data management: Firebase handles stove status, temperature readings, pan placement, and user-set timers.
- Predefined cooking modes: The database stores a list of dishes with preset cooking times and temperatures, simplifying automation.
- Instant alerts: Users receive notifications for critical events, such as a misplaced pan or when predefined safety thresholds are exceeded, ensuring a safe cooking experience.

4.3.3 Firebase Integration and Connection with ESP32

The ESP32 connects to Firebase via the Firebase ESP Client library, authenticated using an API key and database URL. Figures 4.11 and 4.12 show a snippet of the initialization code.

```
#define API_KEY "AIzaSyBZwe9FBouXoEKrimW3F0g4v1Jy37UOvYA"
#define FIREBASE_HOST "iot-project-17fe2-default-rtdb.firebaseio.com"
#define FIREBASE_PROJECT_ID "iot-project-17fe2"
#define FIREBASE_CLIENT_EMAIL "firebase-adminsdk-aaf1a@iot-project-17fe2.iam.gserviceaccount.com"
```

Figure 4.11: Firebase Variable Definition

```
firebaseConfig.api_key = API_KEY;
  firebaseConfig.database_url = FIREBASE_HOST;
  firebaseConfig.service_account.data.client_email = FIREBASE_CLIENT_EMAIL;
  firebaseConfig.service_account.data.project_id = FIREBASE_PROJECT_ID;
  firebaseConfig.service_account.data.private_key = PRIVATE_KEY;
  firebaseConfig.token_status_callback = tokenStatusCallback;
```

Figure 4.12: Firebase Initialization Code

4.3.4 ESP32 Firmware

The ESP32 executes core functionalities, including:

- Sensor data handling for temperature, motion, and pan detection.
- Temperature regulation based on real-time user inputs from the app.
- Timer management, running independently even if WiFi disconnects.
- ESP32-CAM image capture, with images stored in Firebase as Base64 format for easy access.

4.3.5 Temperature Control in ESP32

This code snippet, shown in Figure 4.13, demonstrates how the system regulates the stove's temperature within predefined target ranges while accounting for the metal's thermal properties. For targetRange = 2, the stove remains on until 100°C when heating, even if it was turned off at 90°C. During cooling, the temperature drops to 75°C before rising again to 90°C, turning on at 85°C, and stabilizing at 100°C. Similarly, for targetRange = 3, the stove turns on until 150°C when heating, even if it was turned off at 140°C. When cooling, it lowers to 130°C, then climbs back to 140°C, turning on at 135°C and stabilizing at 150°C.

```
} else if (targetRange == 2) {
  if(panToStoveRem){
  if (temperature <= 50) turnOnStove();
  if (temperature >= 90) turnOffStove();//up to 100
  if (temperature > 50 && temperature < 85) turnOnStove();
  } else {
    turnOffStove();
  }
} else if (targetRange == 3) {
  if(panToStoveRem){
  if (temperature <= 100) turnOnStove();
  if (temperature >= 140) turnOffStove();
  if (temperature > 100 && temperature < 135) turnOnStove();
  } else {
    turnOffStove();
  }
```

Figure 4.13: ESP32 Temperature Control Code for Stove Regulation

4.4 Challenges

Several challenges were encountered and addressed during implementation:

- Remote stove temperature control was challenging, but testing led to an optimized series-parallel relay setup.
- Finding suitable components, especially relays, required extensive research and took longer than expected.
- Uploading code to the ESP32 while connected to relays caused issues, requiring us to disconnect them for each update, adding unexpected delays.
- The limit switch produced unstable analog signals on ESP32 pins 34 and 35; switching to different input pins resolved the issue.
- The PIR sensor produced inconsistent motion detection, occasionally misreading movement. Replacing it with an ultrasonic sensor provided more stable and accurate results.
- We initially considered a TFT screen but switched to an OLED with SPI support for a more stable connection and improved buffering.

- We faced an issue uploading code to the ESP32-CAM due to incorrect wiring. The problem was resolved by properly connecting RX to RX and TX to TX, as shown in **Figure 4.14**, enabling successful uploads.

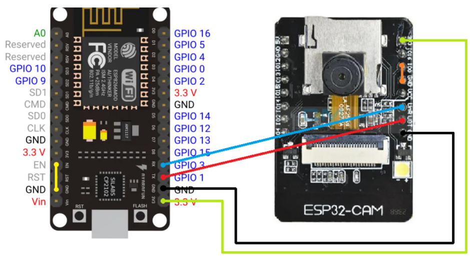

Figure 4.14: Correct wiring configuration for ESP32-CAM to resolve upload issues

4.5 Summary

This chapter covered the hardware and software implementation, the challenges faced, and their solutions. The system successfully integrates real-time monitoring, remote control, and safety mechanisms, ensuring an efficient and user-friendly smart cooking experience.

Chapter 5

Testing and Results

5.1 Preface

The testing of all system components and the results will be covered in this chapter.

5.2 Testing and Detailed Analysis of the Results and Experiments

5.2.1 Limit Switch

We conducted a comprehensive test on the limit switches to verify all the previously explained scenarios by placing a pan on the stove and testing each case. **Figures 5.1, 5.2, and 5.3** illustrate the testing method.

Figure 5.1: Pan in Center

Figure 5.2: Pan Was Moved

Figure 5.3: Pan Displaced

- **Figure 5.1** illustrates the test conducted to ensure that placing the pan on the stove activates it and successfully performs the programmed functions in the ”on” state.

- **Figure 5.2** tested whether moving the pan off the stove triggers the buzzer and turns off the stove.

- **Figure 5.3** tested whether removing the pan from the stove simply turns off the stove (without triggering the buzzer) as the cooking process would have already been completed.

Additionally, the mobile application monitors these scenarios, as illustrated in Figure 5.4, which displays the pan's status on the app. The system sends a notification to the user when the pan is removed from the stove, as shown in Figure 5.2, ensuring that the conditions are effectively tracked, regardless of whether the system is in manual or remote mode.

Figure 5.4: App Screenshot Showing Pan on Stove Status

5.2.2 OLED

We tested the functionality of the OLED to display the system's essential notifications.

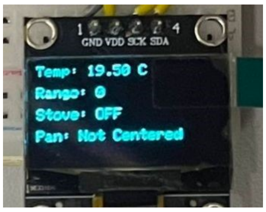

Figure 5.5: OLED notifications

As shown in Figure 5.5, the OLED displays the stove status (on/off), whether the pan is on the stove or not (or if it has been moved, which can also be identified through the buzzer and the application), and the stove's temperature levels continuously. Temporary notifications appear when motion is detected near the stove.

5.2.3 ESP-Cam with UltraSonic

During testing, the ultrasonic sensor was set to detect motion within a range of 150 cm (1.5 meters), with scans occurring every 30 seconds. The ESP32-CAM was responsible for capturing images while the ESP32 handled all managing notifications, controlling the stove (ON/OFF), and activating the buzzer. The following results were observed:

Motion Detection and Image Capture

The ESP32-CAM detects motion, captures images, and notifies the user.

- When motion was detected within the specified range, the ESP32-CAM successfully captured an image and sent it to the mobile application.
- The user received a notification: "There is movement around the stove. Do you want to pause it?", As shown in figure 5.6

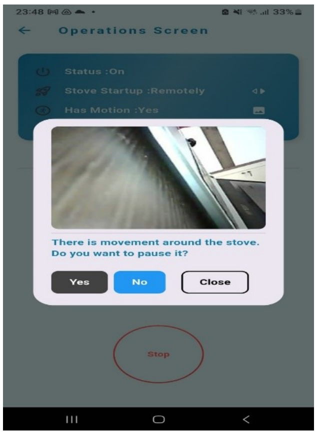

Figure 5.6: App Screenshot when there is Movement

User Response - Turn Off Stove

The following steps outline the sequence the user follows to turn off the stove:

- If the user selected "Yes" to turn off the stove: The stove was successfully deactivated, the buzzer was activated, and the user received a follow-up notification: "The buzzer is on, do you want to stop it?". As shown in figure 5.7

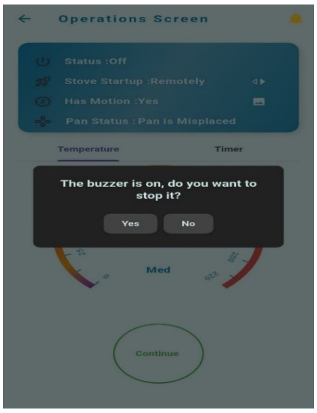

Figure 5.7: App Screenshot when there is Buzzer ON

- If the user selected "Yes" to turn off the buzzer: The buzzer was deactivated, and the stove resumed operation (since the danger was no longer present).
- If the user selected "No": The buzzer remained active, and the stove stayed off.

User Response - Do Not Turn Off Stove

If the user selected "No" to keep the stove on:
The stove continued operating normally, and no further actions were taken unless motion was detected again.

No User Response Within 1 Minute

If the user did not respond within 1 minute:
The system automatically turned off the stove and activated the buzzer, and the user was notified: "The stove has been turned off, and the buzzer is active due to no response.".

Change in Motion Status (No Motion Detected)

If no motion was detected while the stove was off and the buzzer was active:
The buzzer was automatically turned off, and the stove resumed operation.

5.2.4 Stove and Thermocouple Performance Analysis

The following experiments evaluate the stove and thermocouple's accuracy, performance, and response under controlled conditions.

Testing Process for the Temperature

The thermocouple and stove system were tested in manual and remote modes under specific conditions to evaluate accuracy, performance, and safety. The testing setup included:

- **Environmental Conditions:** 17°C ambient temperature.
- **Cooking Parameters:** A stainless steel pan with a diameter of 9 cm^2 containing 200 ml of water.
- **Temperature Ranges:** Predefined ranges (e.g., 50°C to 100°C, 100°C to 150°C) were analyzed.

Results

The results confirm the thermocouple's accuracy and the stove's stable heating performance, as follows:

- **Thermocouple Accuracy:**
  - The thermocouple provided accurate temperature readings within an error margin of ±2°C, ensuring precise feedback for stove operation.
- **Automated Mode Performance:**
  - **Heating Times:** Table 5.1 presents the heating times for different temperature ranges, showing the system's ability to reach target temperatures efficiently.

Test	Current Temp (°C)	Target Temp (°C)	Duration (Minutes)
Test 1	17	50	1:29
Test 2	50	100	1:11
Test 3	100	150	1:20
Test 4	150	200	3:20
Total			**7:20**

Table 5.1: Heating Times

  - **Cooling Times:** Table 5.2 illustrates the cooling durations, demonstrating how long it takes for the stove to return to lower temperatures under controlled conditions.

Test	Current Temp (°C)	Target Temp (°C)	Duration (Minutes)
Test 1	200	170	1:24
Test 2	170	150	0:37
Test 3	150	100	1:50
Test 4	100	50	3:58
Test 5	50	17	10:13
Total			**18:02**

Table 5.2: Cooling Times

- **Safety Observations:**

  The system was evaluated for safety thresholds:

  * At 50°C, the surface is not harmful and remains within a safe range for accidental contact.
  * At 40°C, the stove surface is completely safe, posing no risk to the user.

- **Manual Mode Performance:** In manual mode, the stove took 3:30 minutes to reach boiling water from (200°C), under similar ambient conditions (17°C). **Manual Mode Heating Times:** Table 5.3 presents the heating times for different temperature ranges in manual mode, demonstrating the system's response time and efficiency when manually controlled.

Test	Current Temp (°C)	Target Temp (°C)	Duration (Minutes)
Test 1	17	50	1:35
Test 2	50	100	1:15
Test 3	100	150	1:25
Test 4	150	200	3:30
Total			**7:45**

Table 5.3: Heating Times in Manual Mode

- **Boiling Water:** Boiling time and max temperature tests assessed heating efficiency.
  - At level 3, boiling water took 10:45 minutes.
  - At level 4, boiling water took 6:35 minutes.
- **Maximum Temperature:** The stove could reach a maximum of 260°C but required more than 10 minutes to rise from 200°C to 260°C.

Comparison with a Normal Gas Stove

A gas stove has three temperature levels (137°C, 170°C, and 220°C) and it was tested for comparison:

- **Heating Times:**
  - Reaching 220°C took just 0:05 minutes, significantly faster than the metal stove due to direct flame transfer.

  - Boiling water at 220°C took 3:30 minutes, comparable to the manual mode of the electric stove set to 200°C (3:30 minutes excluding preheating).

- **Cooling Times:**
  - The gas stove cooled from 150°C to 100°C in 0:50 minutes, significantly faster than the metal stove's 1:50 minutes.

Discussion for Temperature

- The thermocouple demonstrated high accuracy and maintained target ranges effectively.
- The heating and cooling times for the electric stove were longer compared to the gas stove due to the thermal properties of metal and indirect heat transfer.
- Safety features ensured user protection, particularly at lower temperatures (40°C to 50°C), which are safe for accidental contact.

5.2.5 Heating Error

This subsection compares the heating time differences between automatic mode and manual mode across various temperature ranges, evaluating the efficiency of each mode.

Time Comparison

Table 5.4 presents the heating times for both automatic mode and manual mode, along with the percentage error in each range.

Temperature Range (°C)	Automatic Mode (Minutes)	Manual Mode (Minutes)	Error (%)
17 - 50	1:29 (89s)	1:35 (95s)	$\frac{89-95}{95} \times 100\% = -6.32\%$
50 - 100	1:11 (71s)	1:15 (75s)	$\frac{71-75}{75} \times 100\% = -5.33\%$
100 - 150	1:20 (80s)	1:25 (85s)	$\frac{80-85}{85} \times 100\% = -5.88\%$
150 - 200	3:20 (200s)	3:30 (210s)	$\frac{200-210}{210} \times 100\% = -4.76\%$
Total	**7:20 (440s)**	**7:45 (465s)**	$\frac{440-465}{465} \times 100\% = -5.38\%$

Table 5.4: Heating Time Comparison Between Automatic and Manual Mode with Corrected Error Calculation

Heating Error Analysis

From Table 5.4, the average heating error between automatic and manual modes is calculated as:

$$\text{Average Heating Error} = \frac{-6.32 + (-5.33) + (-5.88) + (-4.76) + (-5.38)}{5} = -5.53\%$$

The small variation suggests that the automated mode heats up faster than the manual mode, demonstrating improved efficiency while maintaining accuracy within an acceptable margin.

5.2.6 Cooling Error

Unlike heating, the cooling process in both automatic mode and manual mode follows the same behavior since they occur within the same system. In both cases, once the stove is turned off, heat dissipation relies entirely on the stove itself. The average cooling error between automatic and manual modes is calculated as:

$$\text{Average Cooling Error} = \frac{0.00 + 0.00 + 0.00 + 0.00 + 0.00}{5} = 0.00\%$$

Since no active cooling system is implemented in either mode, there is no significant difference in cooling time. Therefore, the cooling error is effectively zero, as both modes exhibit identical cooling characteristics.

5.2.7 Timer Accuracy Analysis

The timer was rigorously tested using lap timing on phones to ensure precise operation. This method provided a highly accurate assessment of the timer's performance during different cooking scenarios. **Success Rates:**

- Range Control: 97%
- Timer Accuracy: 99%

5.3 Justifications of the Obtained Results

This section explains the system's heating, cooling, and safety performance, highlighting expected delays due to material properties and ensuring safe temperature thresholds.

- **Heating Performance:**
  - Longer heating times were expected for the electric stove due to the thermal properties of metal and the ambient temperature (17°C).
  - Error analysis showed that actual heating times were slightly longer in manual mode than in automatic mode, with calculated errors ranging from -6.32% to -4.76%, indicating that automation provides a small but measurable improvement in efficiency.

  - These discrepancies highlight the importance of experimental validation to assess the real-world performance of the system, ensuring that automation improves heating efficiency while maintaining reliability.

- **Cooling Performance:** Since both the automatic and manual modes rely on the same passive cooling mechanism, the cooling process is identical in both cases. The results confirm that heat dissipation occurs naturally through conduction, convection, and radiation, leading to zero cooling error between modes.

- **Safety Thresholds:** The system ensures user safety by maintaining temperatures under 50°C as non-harmful and 40°C as completely safe.

5.4 Summary

This chapter evaluated the system's components, confirming their accuracy and reliability. The limit switches, OLED, and ESP32-CAM functioned effectively, ensuring accurate motion detection, feedback, and notifications. The thermocouple maintained precise temperature control within ±2°C, and the timer demonstrated high accuracy using lap timing. While the metal stove's heat transfer was slower than a gas stove, the system successfully met all functionality, safety, and usability requirements.

Chapter 6

Conclusion and Future Work

6.1 Preface

This chapter summarizes the achievements of the project and outlines potential areas for future development.

6.2 Concluding Remarks

This project successfully developed a smart cooking system that enhances safety, usability, and efficiency. By integrating hardware components—limit switches, thermocouples, OLED displays, ESP32-CAM, and ultrasonic sensors—with a mobile application and Firebase, the system ensures reliable performance and user-friendly operation.

The limit switches enabled accurate pan detection, preventing unsafe use, while the OLED display provided clear real-time feedback. The ESP32-CAM and ultrasonic sensor effectively detected motion, enhancing safety through timely notifications. The thermocouple ensured precise temperature regulation, though the stove's heat retention extended heating and cooling times. The timer demonstrated high accuracy, validated through rigorous testing.

The system met all design objectives, achieving high reliability despite minor challenges, such as environmental effects on heat transfer. This project demonstrates the potential of smart cooking systems to improve kitchen safety and efficiency, paving the way for future advancements in intelligent home automation.

6.3 Future Work

- Implementing a user management system with authentication and authorization to:
  - Create individual user accounts for each device.
  - Support multiple users on the same device with secure access and personalized settings.
- Utilizing the camera as a second level of feedback to monitor and provide real-time status updates on the stove's operation.
- Implementing image processing capabilities in the camera to recognize the presence of a child and trigger safety mechanisms automatically.
- Upgrading the manual screen to a touch-enabled interface for enhanced user interaction and ease of operation.
- Expanding the mobile application to support iOS devices, enabling compatibility with Apple mobile devices and broadening accessibility for users.

Bibliography

[1] Avnet Abacus. Pir and ultrasonic sensors: What's the difference and how do they work? https://my.avnet.com/abacus/resources/article/pir-andultrasonic-sensors-whats-the-difference-and-how-do-they-work/, 2024. Last accessed: 12 May 2024.

[2] Android. Developer documentation. https://developer.android.com/, 2021. Last accessed: 12 May 2024.

[3] Arduino. Arduino mega 2560 documentation. https://docs.arduino.cc/hardware/mega-2560, 2024. Last accessed: 12 May 2024.

[4] B. Bouchard, K. Bouchard, and A. Bouzouane. A smart cooking device for assisting cognitively impaired users. *Journal of Reliable Intelligent Environments*, 6(2):107–125, 2020.

[5] Energy Sector Management Assistance Program (ESMAP). *The State of Access to Modern Energy Cooking Services*. World Bank, Washington, DC, 2020. Last accessed: 05 April 2024.

[6] Flutter. Documentation. https://flutter.dev/, 2021. Last accessed: 01 May 2024.

[7] Raspberry Pi Foundation. Raspberry pi 4 model b specifications. https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/, 2024.

[8] Nabto. Guide to iot esp-32. https://www.nabto.com/guideto-iot-esp-32, 2024. Last accessed: 12 May 2024.

[9] React Native. Documentation. https://reactnative.dev/, 2021. Last accessed: 12 May 2024.

[10] Aaron Russell. Firebase vs supabase vs aws amplify vs appwrite, 2025. Accessed: 2025-01-29.

[11] Peak Sensors. What is rtd sensor and how does it work? https://peaksensors.com/blog/resistance-thermometer/what-is-rtd-sensor-and-how-doesit-work/, 2024. Last accessed: 12 May 2024.

[12] H. Su, C. Wu, A. Lin, P. Liu, and P. Leung. Smart stove functional specification. Technical report, 2009.

[13] C. Svosve and L. Gudukeya. Design of a smart electric cooking stove. *Procedia Manufacturing*, 43:135–142, 2020.

[14] World Health Organization. Burns - fact sheet. https://www.who.int/news-room/fact-sheets/detail/burns, 2024. Last accessed: 05 April 2024.

Appendix A

Appendix: ESP32 Firmware Code

```
#define FIREBASE_DEBUG
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include "max6675.h"
#include <HardwareSerial.h>
#include <WiFi.h>
#include <Firebase_ESP_Client.h>
#include "addons/TokenHelper.h"
#include "addons/RTDBHelper.h"

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET -1
#define SCREEN_ADDRESS 0x3C
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT,
&Wire, OLED_RESET);

// Relay and Thermocouple pins
#define RELAY_PIN 12
#define RELAY_PIN_2 13

// Global Variables
int thermoDO = 26;
```

```
int thermoCS = 27;
int thermoCLK = 14;

// Limit Switch Pins
#define LIMIT_SWITCH_PIN1 33
#define LIMIT_SWITCH_PIN2 32
#define LIMIT_SWITCH_PIN3 25
#define LIMIT_SWITCH_PIN4 17
#define NUM_LIMIT_SWITCHES 4

const char* ssid = "Wi-Fi";
const char* password = "***";
const int LIMIT_SWITCH_PINS[NUM_LIMIT_SWITCHES] =
{LIMIT_SWITCH_PIN1, LIMIT_SWITCH_PIN2,
LIMIT_SWITCH_PIN3, LIMIT_SWITCH_PIN4};

// UART Pins for Serial Communication
const int serialTxPin = 4;
const int serialRxPin = 15;

// Buzzer Pin
const int buzzerPin = 5;

bool motionDetected = false;
bool userDecisionReceived = false;
bool userWantsToTurnOff = false;
const unsigned long responseWaitTime = 30000;

int targetRange = 0;
bool stoveOnRem = false;
bool stoveOnMan = true;
bool manualMode = true;
int stoveTemperature = 25;
int powerLevel = 3;
bool panInCenter = false;
bool panDisplaced = false;
bool panDetected = true;
bool systemRunning = true;
bool panToStoveRem = true;
```

```
// Firebase Configuration
#define API_KEY "API_KEY"
#define FIREBASE_HOST "FB-host"
#define FIREBASE_PROJECT_ID "PROJECT_ID"
#define FIREBASE_CLIENT_EMAIL "CLIENT_EMAIL"

const char PRIVATE_KEY[] PROGMEM = "PRIVATE_KEY";
#define DEVICE_REGISTRATION_ID_TOKEN "TOKEN"
FirebaseConfig firebaseConfig;
FirebaseAuth firebaseAuth;
FirebaseData fbdo;
FirebaseData stream;

unsigned long sendDataPrevMillis = 0;
volatile bool dataChanged = false;

String token = "";
bool signupOK = false;
bool hasMotion = false;
int panIsCentered = 0;
int buzzer = 0;
int response = 0;

// UART2
HardwareSerial espCamSerial(2);

bool limitSwitchState1;
bool limitSwitchState2;
bool limitSwitchState3;
bool limitSwitchState4;

// MAX6675 object for thermocouple
MAX6675 thermocouple(thermoCLK, thermoCS, thermoDO);

void setup() {
    Serial.begin(115200);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(1000);
        Serial.print(".");
    }
```

```
    Serial.println("\nConnected to WiFi!");

    if (!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
        Serial.println(F("SSD1306 allocation failed"));
        for (;;);
    }
    display.clearDisplay();
    display.display();

    pinMode(RELAY_PIN, OUTPUT);
    digitalWrite(RELAY_PIN, HIGH);
    pinMode(RELAY_PIN_2, OUTPUT);
    digitalWrite(RELAY_PIN_2, LOW);
    pinMode(buzzerPin, OUTPUT);
    digitalWrite(buzzerPin, LOW);

    firebaseConfig.api_key = API_KEY;
    firebaseConfig.database_url = FIREBASE_HOST;
    firebaseConfig.service_account.data.client_email =
    FIREBASE_CLIENT_EMAIL;
    firebaseConfig.service_account.data.project_id =
    FIREBASE_PROJECT_ID;
    firebaseConfig.service_account.data.private_key =
    PRIVATE_KEY;

    Firebase.begin(&firebaseConfig, &firebaseAuth);
    Firebase.reconnectWiFi(true);
}

void loop() {
    if (stoveOnMan) {
        Manually();
    } else {
        Remotely();
    }
    delay(1000);
}

void Manually() {
    Serial.println("Operating in Manual mode");
    float stoveTemperature = thermocouple.readCelsius();
```

```
    Serial.print("Current Temperature: ");
    Serial.print(stoveTemperature);
    Serial.println(" C ");
}

void Remotely() {
    Serial.println("Operating in Remote mode");
    float stoveTemperature = thermocouple.readCelsius();
    controlStove(stoveTemperature);
    Serial.print("Current Temperature: ");
    Serial.print(stoveTemperature);
    Serial.println(" C ");
}

void controlStove(float stoveTemperature) {
    if (!panDetected) {
        turnOffStove();
        return;
    }
    if (targetRange == 1) {
        if (stoveTemperature <= 25) turnOnStove();
        if (stoveTemperature >= 50) turnOffStove();
    } else if (targetRange == 2) {
        if (stoveTemperature <= 50) turnOnStove();
        if (stoveTemperature >= 90) turnOffStove();
    }
}

void turnOnStove() {
    digitalWrite(RELAY_PIN, LOW);
    stoveOnRem = true;
    Serial.println("Stove is now ON");
}

void turnOffStove() {
    digitalWrite(RELAY_PIN, HIGH);
    stoveOnRem = false;
    Serial.println("Stove is now OFF");
}
```

```
void sendTemperatureToFirebase(float stoveTemperature) {
    Serial.println("Sending temperature to Firebase...");
    if (Firebase.RTDB.setFloat(&fbdo, "/stoveTemperature",
    stoveTemperature)) {
        Serial.println("Temperature sent successfully!");
    } else {
        Serial.println("Failed to send temperature.");
    }
}
```