DEVELOPMENT OF A PAYMENT GATEWAY SYSTEM ON AN IOT-ENABLED DEVICE FOR PRODUCT DISPENSING OR ACTION AFTER PAYMENT

A Project Report Submitted By

Sourabh Shenoy	**4NM21EC147**
T Gautham Poojary	**4NM21EC166**
T Tushar Shenoy	**4NM21EC167**
Tanvi N Shetty	**4NM21EC168**

Under the Guidance of

Dr Sukesh Rao M

Associate Professor

in partial fulfillment of the requirements for the award of the Degree of

Bachelor of Engineering

in

Electronics and Communication Engineering

from

Visvesvaraya Technological University, Belagavi

Nitte-574110, Karnataka, India

November-2024

NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India
ISO 9001:2015 Certified, Accredited with 'A' Grade by NAAC
☎:08258 - 281039 – 281263, Fax: 08258 – 281265

Department of Electronics and Communication Engineering

Certificate

Certified that the project work entitled " ***Development of a payment gateway system on an IoT-enabled device for product dispensing or action after payment****" is a bonafide work carried out by* ***Sourabh Shenoy (4NM21EC147), T Gautham Poojary (4NM21EC166), T Tushar Shenoy (4NM21EC167) & Tanvi N Shetty (4NM21EC168)*** *in partial fulfillment of the requirements for the award of* ***Bachelor of Engineering Degree in Electronics and Communication Engineering*** *prescribed by* ***Visvesvaraya Technological University, Belagavi*** *during the year 2024-2025. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree.*

Signature of the Guide **Signature of the HOD**

Signature of the Principal

Semester End Viva Voce Examination

	Name of the Examiners	**Signature With Date**
1.	_____	_____
2.	_____	_____

Abstract

The rapid advancement of the Internet of Things (IoT) technology has paved the way for innovative applications across various industries. This project aims to develop a payment gateway system integrated with an IoT-enabled device for automated product dispensing or action initiation post-payment. The system combines secure payment processing with IoT capabilities to enhance user convenience and operational efficiency. Integrating QR code technology with IoT-enabled devices presents a novel approach to streamline payment processes and automate product dispensing or action initiation post-payment. The proposed system architecture comprises three primary components: a QR code generation and scanning module, a secure payment processing gateway and an IoT-enabled microcontroller to execute the dispensing or action mechanism. Users initiate a transaction by scanning a QR code with their mobile device, redirecting them to a secure payment gateway supporting the UPI payment method. Upon successful payment verification, the payment gateway communicates with the IoT microcontroller, triggering the intended action such as dispensing a product, unlocking a service or activating a device. This process is designed to ensure seamless and contactless transactions, enhancing user convenience.

Acknowledgement

Our project would not have succeeded without various personalities' encouragement, guidance and support. First and foremost, we would like to express our sincere gratitude towards our project guide **Dr. Sukesh Rao M**, Associate Professor, Department of Electronics and Communication Engineering, N. M. A. M. Institute of Technology, Nitte, for his guidance, encouragement and inspiration throughout the project work.

We extend our gratitude to **Dr. K. V. S. S. S. S. Sairam**, Professor and Head, Department of Electronics and Communication Engineering, N. M. A. M. Institute of Technology, Nitte, for his encouragement and for providing the necessary facilities in the department.

We wish to acknowledge the support of **Dr. Niranjan N. Chiplunkar**, Principal, N. M. A. M. Institute of Technology, Nitte, for providing a motivational academic environment.

We would like to sincerely thank all the teaching and non-teaching staff of the Department of Electronics and Communication Engineering, N. M. A. M. Institute of Technology, Nitte, for their patience and help during our project work.

Finally, we would like to thank all our friends and well-wishers who have helped us whenever needed and supported us.

Sourabh Shenoy
T Gautham Poojary
T Tushar Shenoy
Tanvi N Shetty

Table of Contents

List of Figures

Chapter 1
Introduction

1.1 General Introduction

The concept of the Internet of Things [1] (IoT) was first proposed by Kevin Ashton in 1999. The main objective of IoT was to simplify life by connecting things and not worrying about geographical restrictions. The Internet of Things (IoT) refers to a network of interconnected sensors, processors and electronic devices spread across the globe, capable of communicating with each other via the Internet. These devices utilise Unique Identifiers (UIDs) assigned to each, enabling them to share and transmit information seamlessly.

The Internet of Things (IoT) originated in 1982 when a Coca-Cola vending machine at Carnegie Mellon University became the first device connected to the Internet, reporting its inventory and temperature. In 1990, John Romkey created an internet-controlled toaster, demonstrating early smart device capabilities. The term **Internet of Things** was later coined by Kevin Ashton in 1999, emphasizing the connection of physical objects to the Internet. By 2005, the International Telecommunications Union (ITU) recognized IoT's growing potential. In the 2010s, IoT adoption surged, with applications in smart homes, wearables and industries. By 2020, billions of IoT devices were used globally, transforming sectors like healthcare, agriculture and transportation.

A payment gateway [2] is a technology used to facilitate the processing of online payments for e-commerce and other businesses. It is an intermediary between the customer and the merchant, ensuring that payment transactions are securely authorized and processed. When a customer purchases using credit/debit cards, digital wallets or other online payment methods [3], the payment gateway encrypts sensitive information (like card details) and securely transfers [4] it to the payment processor or bank for verification. Once approved, the funds are transferred from the customer's account to the merchant's account, completing the transaction. Common examples include PayPal, Stripe and Razorpay.

Vending machines [5] are automated devices that dispense products or services to consumers in exchange for payment. They have evolved significantly since their invention, transforming the way goods and services are distributed in commercial markets. Its only goal is to make operations easier,[6] one tap and the user gets the desired product. In today's busy world, people forget to carry their own essential needs with them, they opt to get the things from where they are and so vending

machines can solve their urge. With advancements in technology, people also prefer going cashless and hence any payment in association with cashless methods is preferred.

There are many types of vending machines available in the market. They include RFID-based [7], [8] vending machines built on embedded systems, AI powered vending machines [9], [10], cashless payment vending machines [11] biometric-based vending machines[12], solar-powered vending machines [13], [14], app controlled vending machines [15], [16], blockchain-based vending machines[17],[18], sensor-based vending machines [19],[20].

A payment gateway is essential for IoT-based vending machines as it enables secure and seamless cashless transactions between customers and the machine, enhancing both convenience and efficiency. The process begins when a customer selects a product and initiates payment through methods like mobile apps, or QR codes[21]. The payment information is then sent to the gateway using IoT connectivity, where it is verified with the bank or card provider. Based on the verification, the payment is either approved or denied, allowing the machine to proceed accordingly. Once approved, the funds are transferred to the vendor's account, and a digital receipt is sent to the customer via email or SMS.

A payment gateway is essential for IoT-based vending machines as it enables secure and seamless cashless transactions between customers and the machine, enhancing both convenience and efficiency. The process begins when a customer selects a product and initiates payment through methods like mobile apps or QR codes [21]. The payment information is then sent to the gateway using IoT connectivity, where it is verified with the bank or card provider. Based on the verification, the payment is either approved or denied, allowing the machine to proceed accordingly. Once approved, the funds are transferred to the vendor's account and a digital receipt is sent to the customer via email or SMS.

The need for an IoT-based payment gateway system arises from the limitations of traditional vending machines, which primarily rely on cash or card payments, reducing convenience for customers who prefer digital methods like QR codes. These machines also lack real-time monitoring [3], making inventory management inefficient and costly, as they require physical visits for maintenance and sales tracking, increasing operational costs. Security vulnerabilities in cash-based systems and outdated interfaces lead to a poor customer experience [6]. Despite advances in IoT payment systems, gaps remain. Limited UPI integration is one of these gaps, as most systems still depend on NFC, card, or coin/currency payments. Focusing on UPI, this system provides seamless transactions without needing alternatives. Additionally, many IoT-based systems require OS-based devices and app downloads, reducing engagement. This project uses a low-cost IoT-enabled

microcontroller, eliminating the need for apps or OS devices that users simply scan, pay, and access the service. Beyond vending, this adaptable system supports a wide range of cashless applications.

1.2 Aim

This project aims to design and develop an IoT-enabled payment gateway system focused on UPI-based transactions, utilising a low-cost microcontroller to provide seamless, cashless payment and automation capabilities. This system aims to enhance convenience, security and operational efficiency by eliminating the need for physical currency or OS-based devices and offering versatile applications in product vending, service activation and remote device control.

1.3 Objectives

The Objectives of the project are as follows:

1. Develop a secure, UPI-based payment gateway on an IoT-enabled ESP32 microcontroller system to automate product selection, payment processing and dispensing.

2. Integrate a cashless payment system through Razorpay, enabling UPI transactions for a seamless and secure user experience.

3. Utilise Google Firebase for real-time transaction tracking and data management, ensuring reliable, efficient communication between system components.

4. Eliminate the need for physical currency or OS-based devices, allowing for a versatile, low-cost solution suitable for various applications beyond conventional vending systems

1.4 Problem Formulation

Traditional payment systems for product dispensing and service activation often rely on cash-based transactions or manual processes causing inefficiencies, security risks and limited payment flexibility. These systems typically lack real-time transaction monitoring, making it difficult to manage and verify payments efficiently. Additionally, many existing systems depend on OS-based devices, which can be costly and require frequent maintenance. The challenge is to develop a dedicated IoT-based payment gateway that supports secure, cashless UPI transactions, provides real-time transaction monitoring, and enhances user experience through automation. This system aims to offer a low-cost, versatile solution for various

applications, improving efficiency and user convenience while minimising operational overhead for providers.

1.5 Proposed Method

The proposed method involves developing an IoT-enabled payment gateway system for product dispensing or service activation. This system integrates cashless payment options and automates the dispensing process, providing a seamless, user-friendly experience. The system will utilise the following components and steps:

1. **ESP32 Microcontroller:** The core of the system is responsible for processing user inputs, managing transactions, and communicating with external components like the Firebase database. It connects to the internet via an ISP to monitor Firebase for payment information. As soon as Firebase receives the payment details, the system verifies the payment and initiates the dispensing action accordingly.

2. **Google Firebase:** A real-time cloud-based database used to store transaction data and provide real-time updates on the payment status. Firebase will handle the verification of payments and send confirmation to the ESP32 for product dispensing.

3. **Razorpay:** A popular Indian payment gateway platform that enables businesses to accept digital payments through UPI, credit/debit cards, net banking and mobile wallets.Webhooks feature of Razorpay, which allows real-time notifications of payment events. With webhooks, Razorpay can send payment details to external servers like Firebase immediately after a transaction, enabling systems such as ESP32-based devices to monitor and act on payment confirmations in real-time.

4. **User Interaction and Product Dispensing:** The system includes a TFT display and push buttons for user interaction. The display shows product details and payment information, while the push buttons enable product selection, payment confirmation and system interaction. Once the ESP32 verifies a successful payment, it triggers the dispensing mechanism to automatically release the selected product.

1.6 Methodology

The development of the IoT-enabled payment gateway system for product dispensing or service activation follows a structured approach, ensuring a seamless integration of hardware and software components. Below is an outline of the methodology used:

Component Selection and System Design

The process begins with identifying and procuring essential components, including the ESP32 microcontroller for core processing, and mechanisms for product dispensing or service activation. In the User Interaction section, push buttons and a TFT display provide an interface for selecting products and viewing payment information. The system architecture integrates the ESP32, Razorpay as the secure payment gateway, and Google Firebase as a real-time database, ensuring seamless and secure communication between these elements.

Microcontroller Programming

The ESP32 microcontroller is programmed to perform the core functions of the system. This includes monitoring the payment status stored in Firebase, handling user inputs, and activating the dispensing mechanism upon successful payment verification. Wi-Fi connectivity is established to enable real-time communication between ESP32, Firebase, and Razorpay, ensuring smooth and timely transactions.

Google Firebase Integration

Firebase is configured as a real-time, cloud-based database to securely store transaction information sent from Razorpay's webhooks. Firebase monitors payment statuses, enabling the ESP32 to verify each transaction and trigger dispensing only upon confirmation of a successful payment. This setup allows real-time tracking and secure storage of transaction data, essential for the automated system's reliability.

Razorpay Payment Gateway Setup

Razorpay's webhooks feature is integrated to allow secure UPI-based payments. A pre-generated QR code is displayed on the interface for users to scan and make the payment via UPI. Once the transaction is processed, Razorpay sends the payment details, including success or failure status, directly to Firebase via HTTP using webhooks. This seamless integration ensures that only validated payments proceed to the dispensing or service unlocking stage.

User Interface Development

A user-friendly interface is created using the TFT LCD, displaying essential information such as product options, payment instructions and transaction confirmation. Push buttons are included to enable easy product or service selection and confirmation, providing a smooth interaction between users and the system.

Product Dispensing/Service Activation Mechanism

The dispensing or activation mechanism is connected to the ESP32, which is programmed to initiate the action only after receiving confirmation of a successful payment from Firebase. This design ensures that transactions are secure and automated, with the ESP32 acting as a final checkpoint before initiating any physical action.

Final Deployment and Optimization

The completed system is deployed in a real-world environment, such as a vending machine prototype and tested under various conditions to handle multiple users and transactions simultaneously. The system is fine-tuned for optimal performance, reliability and security, offering an efficient, low-cost, IoT-based solution for automated payment and dispensing. This deployment phase confirms that the system provides a seamless user experience and dependable backend operations, meeting the project's objectives effectively.

1.7 Literature Survey

Humans have evolved over time and so has the currency. The history of money is fascinating and goes back thousands of years. From the early days of bartering to the first metal coins and eventually the first paper currency, money has always had an important impact on how people function as a society. Along with paper money and coins, modern-day money has also branched out to include credit and debit cards, online payments and cryptocurrency. However, with the internet boom and the growth of e-commerce, online payments became more popular. The most popular payment methods in India include Banking cards, USSD (Unstructured Supplementary Service Device), AEPS (Aadhaar Enabled Payment System), UPI (Unified Payment Interface), Mobile wallets, Banks Pre-Paid Cards, Internet Banking, Mobile Banking and Micro ATMs.

One such application of online payment is the IoT-based payment gateway. Technology-based payment systems have now replaced cash transaction systems. Transactions using payment gateway are considered profitable because they can reduce business transaction costs and improve service quality to customers. With the rise of various payment methods, India's digital payment ecosystem is booming. By redefining service by introducing tactics like UPI, IMPS, e-KYC, and Aadhar as an authentication mechanism, the government has helped fintechs' growth. However, the purchase of products and services via various electronic means results in digital payment. Significantly, the emergence of coronavirus (COVID-19) disrupted our traditional cash handling means and triggered an inflection point for switching towards contactless digital payments from physical cash payments. Furthermore, Internet of Things (IoT) technology escalates digital payments to the next level by enabling devices to render goods and services without requiring any human interaction [22]. IoT is about connecting devices that help to optimize operations, boost productivity, lower costs and improve lives. In current days, many applications such as healthcare, mobile banking, mobile payments, vehicle monitoring, home monitoring, etc., are implemented with the help of the Internet of

Things (IoT)[23].

The proliferation of the IoT environment is being implemented in various forms of service in various industries such as personalized customer service where [24] technology and personal life can interact with each other, from purchasing to entertainment. IoT devices collect sensor data, exchange data with other devices using industrial protocols and interact with control systems to optimize operations, enhance machinery efficiency, and enable predictive maintenance to be carried out[25]. The payment gateway can be implemented using many microcontrollers, sensors [26], Bluetooth[27], Wi-Fi modules and so on. Using ESP-32 seemed more interesting and fulfilled our requirements.[25] ESP32 is a powerful and cost-effective platform for developing IoT applications. ESP32 has the following features: a dual-core processor, integrated Wi-Fi and Bluetooth connectivity, many general-purpose input/output (GPIO) pins, and low power consumption. ESP32 has built-in Wi-Fi and Bluetooth interfaces that simplify connection and communication with other devices or networks. The market offers various options in microcontrollers, modules and Bluetooth, among others. Choosing the right set of components depends on the specific requirements. Managing finances is crucial and science continually introduces new methods to manage and spend money. With technological advancements, many product dispensing machines have adopted UPI-based payments.

1.8 Organization of the Report

The report is organized as follows:

Chapter 1 discusses the introduction, aim, objectives, problem formulation, proposed method, methodology and literature survey.

Chapter 2 illustrates the system block diagram and specification of major equipments used.

Chapter 3 discusses the schematic circuit connection, Arduino libraries and various APIs used.

Chapter 4 deals with results and discussions.

Chapter 5 discusses the conclusion and future work.

Chapter 2
System Block Diagram

2.1 Block Diagram

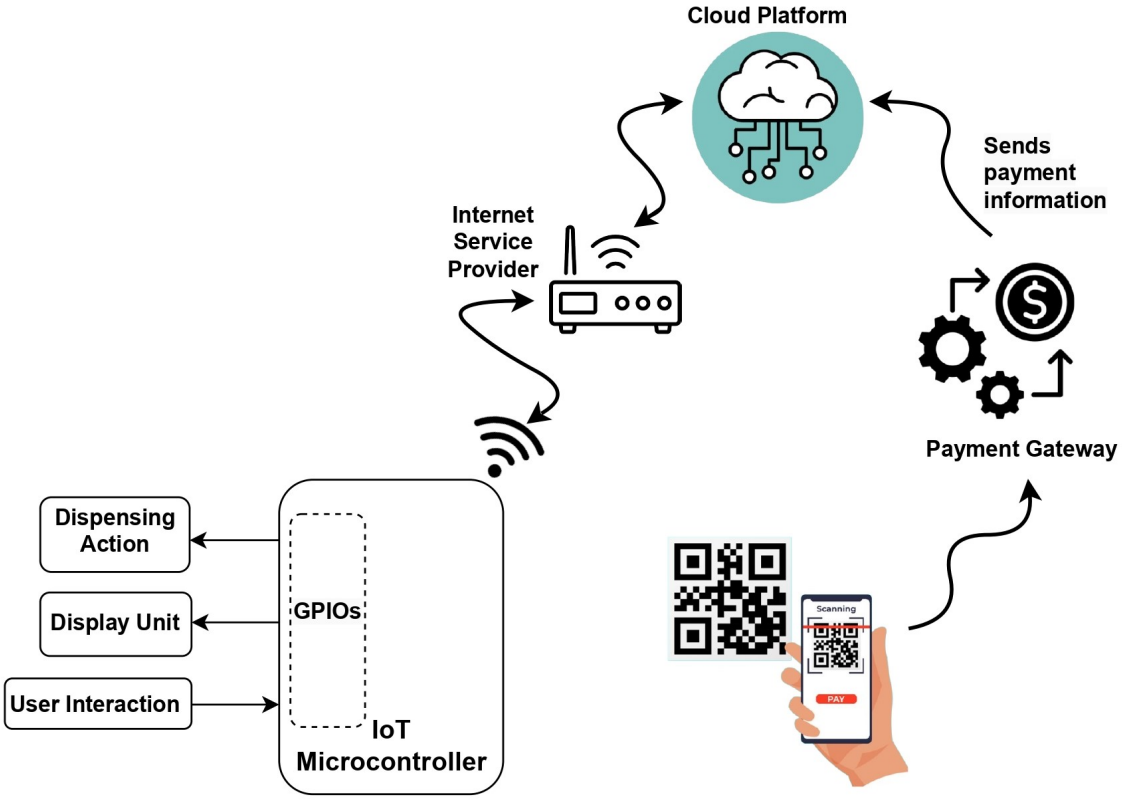

Figure 2.1: Block diagram of the IoT-Enabled Payment Gateway System.

Figure 2.1 illustrates an IoT-enabled payment gateway system designed for a product vending or action-triggering device, such as a vending machine. At the core of this system lies the IoT microcontroller, which manages various components through its GPIO pins. The microcontroller interfaces with the user, a display unit and a dispensing mechanism. When a user interacts with the system, typically through buttons or other input devices this triggers a series of events beginning with the display of a QR code on the display unit. The QR code serves as a prompt for the user to scan it using a mobile device, initiating the payment process.

The payment gateway, a crucial part of this setup, manages the transaction, ensuring that the payment is secure and verified. Once the user completes the payment, the gateway communicates the transaction details to a cloud platform. This platform acts as a bridge, storing the payment information and sending updates to

the IoT microcontroller. The communication between the cloud platform and the microcontroller occurs over the internet, facilitated by an Internet Service Provider (ISP) that ensures real-time connectivity.

Upon receiving confirmation of successful payment from the cloud platform, the IoT microcontroller activates the dispensing mechanism, releasing the selected product or triggering a specified action. If the payment fails, the system will remain inactive for a certain period, giving the user time to attempt the payment again. During this waiting period, no product will be dispensed, and the display will not show any feedback. If the payment is not successfully processed within the specified time frame, the system will reset, returning to its initial state and awaiting further input from the user. This ensures that the process remains secure, automated and the system is ready for a new transaction attempt.

2.2 Specification of Major Equipments

In this project the major equipments are, the ESP32 Wroom 32 microcontroller which serves as the core of the system, managing secure communication and transaction processing. The ESP32's capabilities enable it to continuously monitor Firebase for real-time payment updates and to interface seamlessly with the dispensing mechanism upon transaction confirmation. Firebase acts as the project's real-time database, storing payment statuses and transaction details. Its cloud infrastructure allows the ESP32 to monitor updates continuously, enabling quick, automated responses. Razorpay functions as the secure payment gateway, handling financial transactions by generating a QR code for user payments and updating Firebase with transaction statuses. This integrated setup ensures that only authorized payments are acknowledged, effectively preventing unauthorized access to the system.

The **ESP32 Wroom 32** is an advanced microcontroller with a range of features that make it ideal for managing IoT and payment functions in this project. Its powerful Wi-Fi capabilities enable seamless communication with Firebase, keeping the system responsive and connected. Supporting multiple wireless modes (AP, STA, and AP+STA), the ESP32 allows for flexible network configurations to ensure reliable connectivity. Equipped with dual CPU cores and adjustable clock frequencies, it efficiently handles real-time data from Firebase, monitoring payment statuses and promptly activating the dispensing mechanism upon confirmation. The ESP32 also includes versatile peripherals —such as UART, SPI, I2C and PWM — making it compatible with various hardware components like sensors, displays and actuators, providing expansion potential. With 448 KB of ROM, 520 KB of SRAM, and 4 MB of SPI flash storage, the ESP32 is well-suited for managing code storage, temporary data, and concurrent processing tasks essential to this project's functions of

data monitoring, communication, and hardware control. Operating at 3.3V with an average current of 80 mA, it is efficient for low-power applications and operates reliably within a wide temperature range of -40 to 85°C. Together, the ESP32's capabilities in communication, memory, and processing make it ideal for a secure, automated payment system that coordinates user interactions with Firebase and Razorpay, ensuring real-time, reliable transaction management. Figure 2.2 shows an image of the microcontroller used in this project.

Figure 2.2: ESP 32 Wroom 32

Firebase acts as a cloud-based, real-time database that securely stores transaction data and updates the payment status after processing through Razorpay. The integration with the ESP32 allows for instant transaction verification, enabling timely product dispensing or service activation upon payment confirmation. Firebase's real-time updates ensure seamless, continuous communication between the IoT device and the payment gateway, supporting efficient transaction processing. The ESP32 actively monitors these updates, triggering the dispensing mechanism as soon as payment is confirmed in Firebase.

Figure 2.3 represents the URL used to connect the ESP32 with Firebase, essential for real-time monitoring of transaction data. This link allows the ESP32 to retrieve the current payment status, verify the transaction, and initiate the required action.

Figure 2.4 illustrates Firebase's structure, showcasing how transaction data is stored and updated in real time.

Figure 2.5 highlights the necessary API keys and settings for secure communication between Firebase and the ESP32, ensuring smooth and secure system operation.

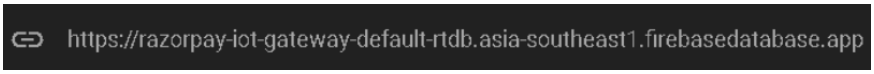

Figure 2.3: Database URL

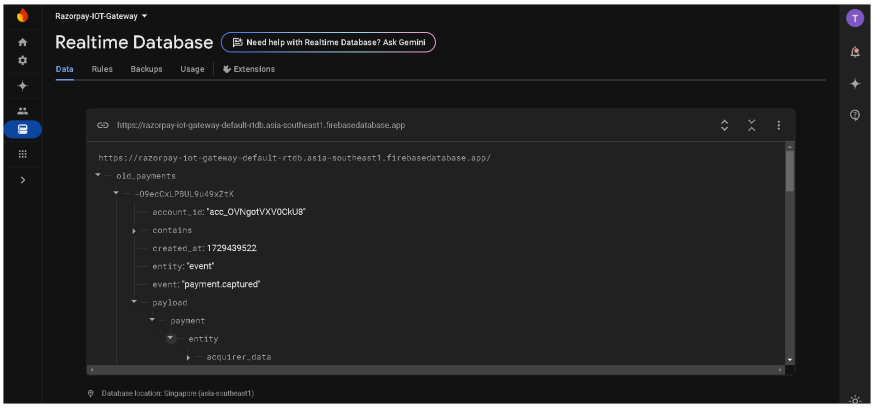

Figure 2.4: Realtime Database

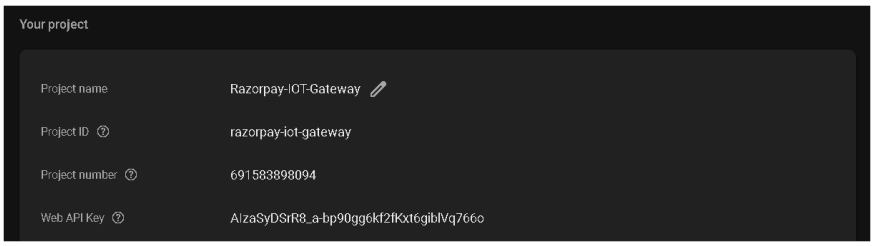

Figure 2.5: Web API keys and others

The payment gateway in this project is **Razorpay**, a secure fintech platform providing streamlined payment solutions. Supporting UPI, credit/debit cards, and net banking, Razorpay enables versatile transaction options. In this project, it facilitates secure, cashless payments by generating QR codes for user transactions. The platform verifies transactions and communicates the payment status to Firebase, allowing the ESP32 to confirm the payment status instantly. This integration supports a smooth, automated dispensing process, ensuring an efficient user experience.

A key Razorpay feature is **webhooks**, automated HTTP requests triggered by specific events. When a user completes a payment, Razorpay sends a webhook with transaction details to Firebase, which then updates the payment status in real time. By using webhooks, the system can monitor transactions without continuous polling, ensuring prompt and reliable payment verification. Figure 2.6 illustrates the configuration of transaction update webhooks while figure 2.7 shows the communication of transaction details via webhook.

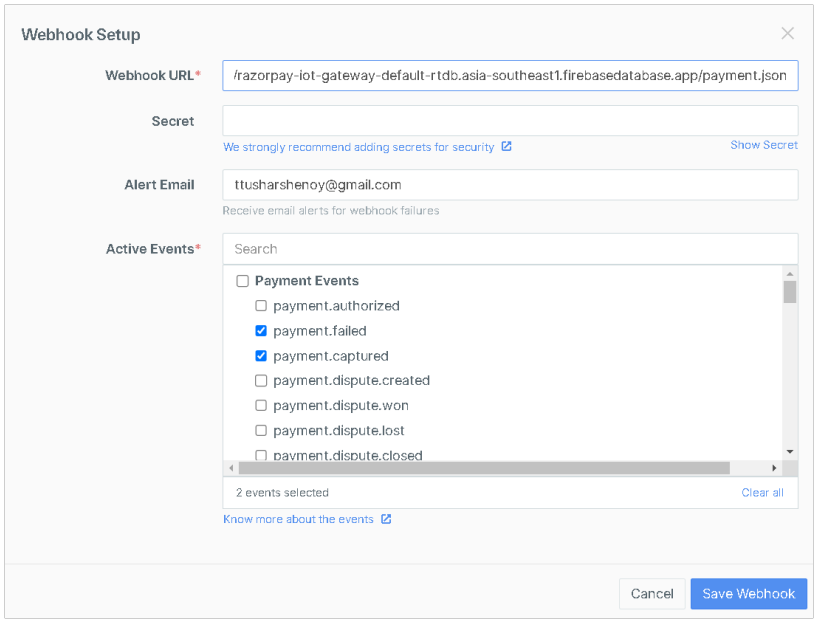

Figure 2.6: Webhook Setup

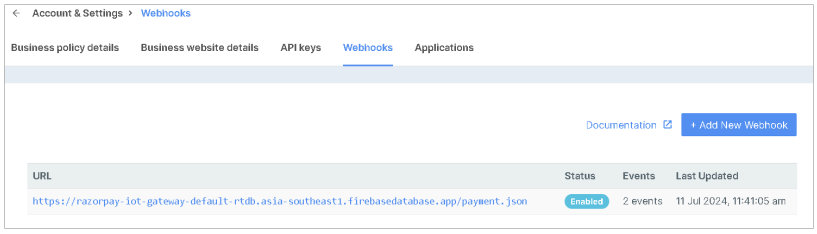

Figure 2.7: Webhooks

Razorpay also generates dynamic QR Codes for various payment amounts, pre-loaded in the system for transaction use. A total of 50 QR codes, ranging from Rs. 10 to Rs. 500 in increments, are stored for quick access. When a product is selected, the system displays the corresponding QR code on the TFT screen for that item, allowing the user to scan and complete the payment.

Figure 2.8 shows the process of generating dynamic QR codes in Razorpay and Figure 2.9 presents an example QR code generated for a product.

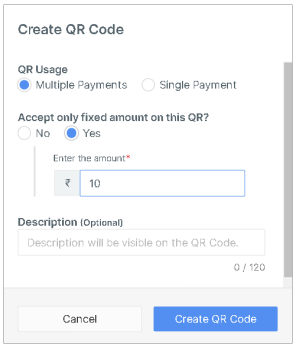

Figure 2.8: Creating QR

Figure 2.9: Dynamic QR Code

The **2.4-inch SPI TFT LCD Display** serves as the primary interface, displaying transaction QR codes and payment statuses, guiding users through the product selection and payment process. Controlled by the ESP32, this display ensures smooth operation. **Push Buttons** allow for product selection and transaction initiation, with each press updating the display as processed by the ESP32. A **buzzer** provides immediate auditory feedback for button presses, confirming actions for improved accessibility and user interaction. Displaying Razorpay-generated QR codes on the TFT screen further enables users to complete secure, cashless payments, supporting a seamless transaction experience.

2.3 Purpose of Interconnections

The purpose of the interconnections in this system is to ensure seamless communication and coordination between all components, enabling the system to function as an integrated unit. The ESP32 microcontroller acts as the central hub, facilitating the transfer of data between the TFT display, payment gateway (Razorpay) via Firebase, user inputs (buttons) and feedback mechanisms (buzzer). Users interact with the system via push buttons to select a product and upon pressing a button, the selected products' details are displayed on the TFT screen for confirmation. The ESP32 displays the dynamic QR code for payment, which users scan via the display using mobile payment options like UPI. The QR code is preloaded onto an SD card and once the user scans it, Razorpay sends transaction data to Firebase using webhooks, a real-time cloud-based database that updates with transaction details. The ESP32 constantly monitors Firebase for payment status updates and when a payment is confirmed, triggers the dispensing mechanism to release the product. These interconnections ensure smooth operation by allowing the system to respond efficiently to user actions and payment statuses, guaranteeing secure product dispensing after payment verification.

Chapter 3
Schematic Circuit Connection

3.1 Connection Diagram Setup

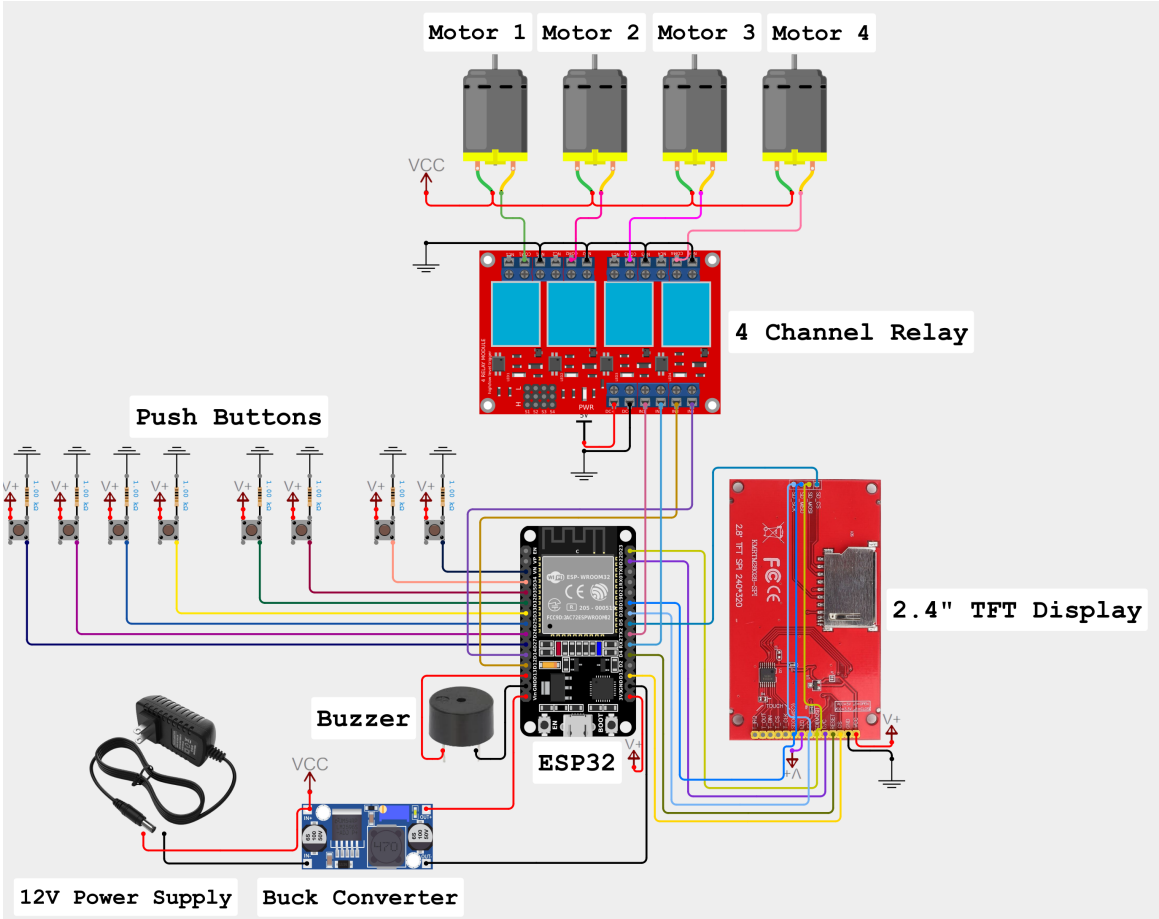

Figure 3.1: Circuit Diagram

The circuit diagram as shown in figure 3.1 depicts a setup featuring an ESP32 microcontroller as the central control unit. The ESP32 is connected to a relay module, a TFT display, push-button switches, a buzzer and multiple motors. Operating as the control hub, the ESP32 receives input from push-buttons connected to its GPIO pins via resistors, which send specific commands, such as activating the motors or triggering other actions.

The relay module, controlled by the ESP32's GPIO pins, powers and operates the motors connected to its output channels. Each motor is linked to a dedicated relay channel, allowing independent operation. The relays receive power from an

external 12V power supply, stepped down to 5V through a buck converter, ensuring proper current requirements for the motors and relays. This 5V output also powers the ESP32 through its VIN pin, with a common ground shared across all components.

For user interaction, the ESP32 connects to an ILI9341 TFT display via SPI communication (using pins D15, D4, D22, D23, D18 and D19 for CS, RESET, D/C, SD (MOSI), SCK and SDO (MISO) respectively, enabling real-time feedback and displaying motor status and relay activity. Additional visual and audio feedback is provided by a buzzer linked to GPIO pin D13. The 4-channel relay module's IN1 through IN4 pins connect to GPIO pins TX2, RX2, D12 and D14 on the ESP32, controlling external devices or motors connected to the relay's output.

Push-buttons are connected to GPIO pins D27, D26, D25, D33, D32, D35, D34 and VN on the ESP32, with each input pin connected to a 3.3V source, allowing the ESP32 to read user commands. The buzzer and display are powered by 3.3V, with ground shared among all components. The motors connect to the relays through the relays' COM and N.O. (Normally Open) pins, with each motor's pin 2 connected to the relay COM pin and grounded through the N.O. pin. This configuration enables the ESP32 to activate relays and complete the motor circuit from a 12V power source connected to motor pin 1.

In this setup, the ESP32 microcontroller manages the whole system, making it easy to control the motors and relays. The TFT display shows information and the buzzer provides sound feedback to the user. The buck converter makes sure the correct voltage is supplied to all parts of the circuit, allowing smooth control over motor functions and with this arrangement, the ESP32 can handle different components, including push-buttons, the buzzer, the display and relays for motor control. The buck converter helps maintain the right voltage across the system, ensuring everything operates correctly.

3.2 Arduino Libraries used:

In this project, various libraries provide essential functionality for interfacing with external components, managing files, communicating over Wi-Fi and integrating with Firebase.

1. **SPI Communication:** The SPI.h library enables Serial Peripheral Interface (SPI) communication, which is crucial for connecting to SPI-based peripherals. In this setup, it establishes a connection between the Arduino board and the SD card or TFT display, allowing smooth data exchange between components over the SPI interface.

2. **File System Management**: Using the FS.h library, the project manages file operations like reading and writing data to storage devices, such as SD cards or internal flash memory. This standardized interface simplifies file handling across different hardware setups, ensuring compatibility and ease of use for interacting with files on storage devices.

3. **SD Card Integration:** The SD.h library provides advanced capabilities to interact with SD cards, which store images or data files for this project. Through this library, files can be created, read and written to the SD card, allowing data logging and retrieval. It also enables navigation through directory structures, making it possible to organize and access multiple files effectively.

4. **Image Decoding:** The JPEGDecoder.h library allows for decoding JPEG images from SD card files, SPIFFS or byte arrays. This feature is essential when the project needs to display images on the TFT display, as it enables JPEG files to be converted into pixel data for visualization. The library supports various architectures, ensuring compatibility with the Arduino boards used in this project

5. **TFT Display Control:** The TFTeSPI.h library provides a powerful interface to manage TFT display functionality. It supports displays with SPI or parallel interfaces and includes functions for drawing text, shapes and images on the screen. This library optimizes the visual output of the system, facilitating a user-friendly display for monitoring information, status updates or images in real-time.

6. **Wi-Fi Connectivity:** The WiFi.h library handles the Wi-Fi connection, enabling the board to connect to local networks or the internet. Through this library, the system can operate as a server or client, enabling data transmission over Wi-Fi. It supports secure connections using various encryption methods and provides IP management, ensuring stable network access for the device.

7. **Firebase Integration:** The FirebaseESPClient.h library is pivotal for connecting the project to Firebase services. With this library, the device can interact with Firebase's Realtime Database, Cloud Firestore, Firebase Storage and Cloud Functions. This integration allows real-time data synchronization between the device and the Firebase backend, making it possible to store and retrieve data, send notifications and execute cloud functions. This functionality is essential for projects requiring remote monitoring, data logging or user notifications via Firebase services.

3.3 Functioning of APIs

The IoT Enabled Payment Gateway System code incorporates various functions from the TFT_eSPI, WiFi and Firebase_ESP_Client libraries to enable its core functionalities. TFT functions are utilized to create a visually appealing and user-friendly interface on the display. The user-defined functions and library functions provide a structured approach to managing product dispensing and payment in the system. The runMotorForProduct() function iterates over available products, controlling each relay to dispense the specified quantity based on the selected amount, ensuring precise control over the motors connected to each product. updateStock() adjusts stock levels post-transaction, preventing negative values and maintaining an accurate inventory. The utility function totalSelectedProducts() sums up the quantities selected, allowing verification before advancing to the payment stage. To enhance user interaction, the beep() function briefly activates a buzzer, signaling actions like button presses or errors for immediate feedback. For displaying QR codes on the screen, jpegRender(int xpos, int ypos) uses the JPEGDecoder library to decode and draw images at given coordinates, while drawSdJpeg(const char *filename, int xpos, int ypos) opens JPEG files from the SD card and renders them, displaying error messages if issues arise. The drawHeader() function initializes the screen interface by drawing a header, setting text properties and providing system identification.

The library functions support key operations. SPI.h manages data exchange via Serial Peripheral Interface (SPI), essential for interactions with the SD card and TFT display. FS.h and SD.h handle SD card operations, using SD.begin() to initialize and SD.open() to open files for reading or writing, such as retrieving QR code images. WiFi.h connects the ESP32 to a Wi-Fi network using WiFi.begin() with network credentials and WiFi.status() to check connectivity, enabling communication with Firebase. For Firebase integration, Firebase.h includes functions like Firebase.signUp() to register the device, Firebase.RTDB.beginStream() to start real-time data streaming and Firebase.RTDB.setStreamCallback() to handle payment status updates and timeouts. JPEGDecoder.h decodes JPEG images, with JpegDec.decodeSdFile() for SD card files and JpegDec.read() for reading image data, essential for QR code display. Finally, TFT_eSPI.h controls the TFT display, using functions like tft.init(), tft.fillScreen(), tft.setTextColor(), tft.setCursor() and tft.print() for setting up the interface, displaying product details, quantities, prices and messages, creating a user-friendly interface for the system.

3.4 Flow Chart

The flowchart for the project code as shown in figure 3.2 details a sequence that supports smooth product selection, payment verification and dispensing, ensuring a user-friendly experience. It begins with hardware initialization, setting up components like the ESP32 microcontroller, TFT display, SD card, relays, push buttons, Wi-Fi and Firebase modules, ensuring all elements are configured for communication. Following this, the system establishes a connection with Google Firebase for real-time monitoring, attempting authentication using pre-configured credentials. If Firebase setup fails (for example, due to network problems), an error message appears and the system periodically retries the connection until successful, as Firebase is crucial for payment status updates.

After successful Firebase authentication, the system waits for user input via buttons. Users can select products, adjust quantities or cancel their selection, with each button press updating the display to reflect current choices, enhancing the selection process. Once quantities are finalized, users can choose to 'Proceed' or 'Cancel'. If 'Cancel' is selected, the system resets, clearing selections and returning to the initial state. On selecting 'Proceed', the system calculates the total payment amount based on chosen quantities, displays it and prompts users to scan a QR code for payment. Simultaneously, the system begins monitoring Firebase for real-time payment updates and initiates a five-minute timer to ensure prompt transaction completion.

If Firebase confirms the payment within this period, the system activates relays connected to motors for the selected products, dispensing items based on quantity. The stock count updates accordingly, providing accurate inventory levels and a confirmation message on the display indicates successful completion. The system then returns to an idle state, ready for the next transaction. This flow ensures an efficient interaction process covering product selection, real-time payment monitoring and automatic dispensing after successful payment.

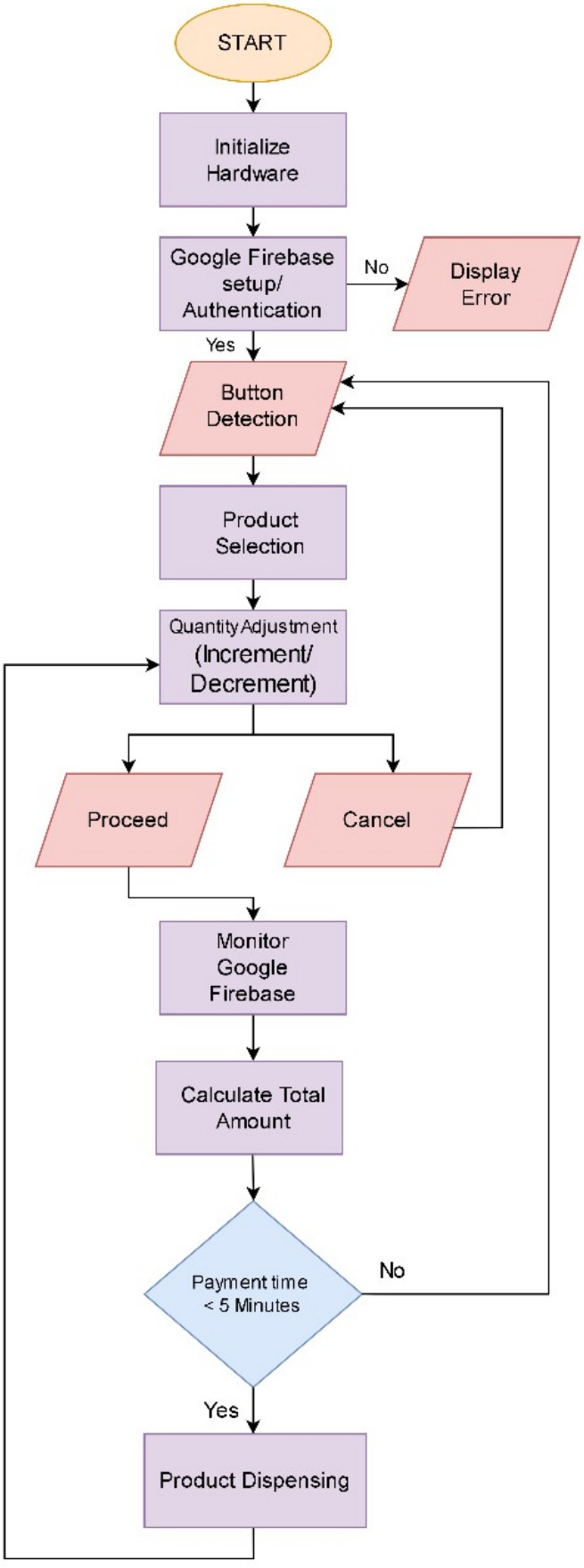

Figure 3.2: Flow Chart

Chapter 4
Results and Discussions

4.1 Experimental Setup:

The experimental setup for the project is as shown in figure 4.1. The breadboard was used to connect and test various components such as the ESP32 microcontroller, push buttons, buzzer, TFT display and product dispensing mechanism before finalizing the design for the vending machine.It was essential for prototyping the system and ensuring the integration of all components as well as testing their functionality before soldering them onto a permanent circuit board. The setup provided valuable insights about wiring, component interaction and debugging.

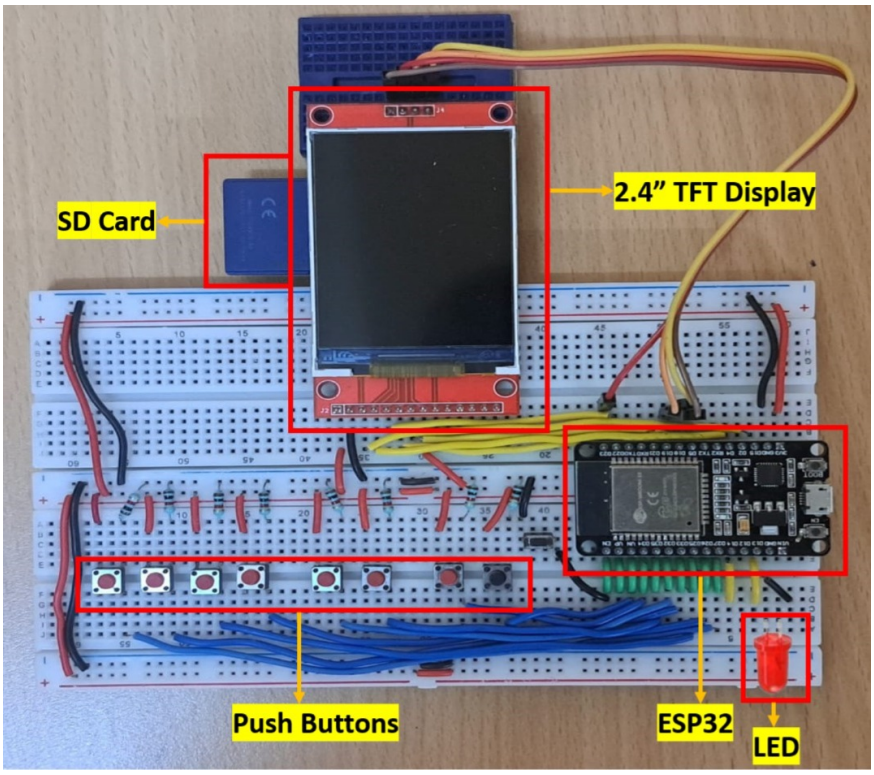

Figure 4.1: Experimental Setup

4.2 Vending Machine Setup

Along with the electronic components, the vending machine structure was built from MDF board, designed using Fusion 360 and cut with a laser cutter for accuracy. This custom-made machine can hold and dispense up to four products

showing the practical use of the developed IoT-based payment gateway system.

The internal setup of the vending machine as shown in figure 4.2 depicts the layout of the dispensing mechanism. It features coiled springs placed in individual compartments, each designed to hold a different product. This arrangement enables organised storage and smooth dispensing, as the coiled springs help push products forward when activated, ensuring each item can be dispensed accurately. Figure 4.3 displays the machine's design as planned in Fusion 360 before adding electronics. It shows an interior view of the vending machine's framework without products or electronic components. The machine features designated compartments with coiled spirals for product dispensing, arranged to support organised storage and smooth product release. The setup ensures each product can be dispensed independently, highlighting the machine's mechanical aspects.

Figure 4.2: Inner View

Figure 4.3: Basic Design

The figure 4.4 shows the fully assembled vending machine with products in place, demonstrating the complete setup. Four distinct product slots are visible, each stocked with different food items. To the right, there's a TFT display and a control panel with labeled push buttons. The display shows the GUI, including options for product selection and controls for operating the vending machine, such as increment and decrement buttons and options to proceed or cancel. This setup illustrates the integration of electronic components with the machine's dispensing mechanism, creating an interactive vending experience.

Figure 4.4: Complete setup

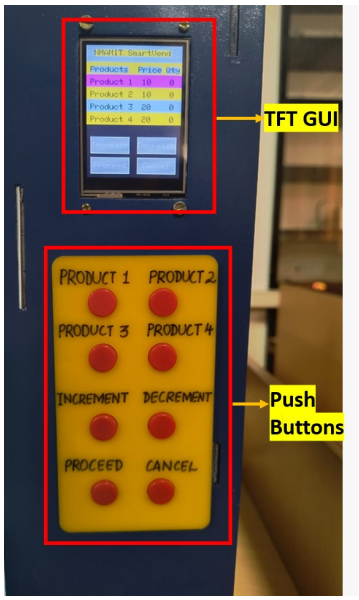

Figure 4.5: TFT GUI and Push Buttons

Figure 4.6: Dynamic QR code

The setup also includes a TFT display interface with easy-to-use buttons and displays like the QR code for payments, product options and navigation buttons for selection as shown in figure 4.5. The images of the TFT display show how users interact with the vending machine, including scanning the QR code to make payments as shown in figure 4.6.

4.3 Successful Integration of the Components

In the development of a payment gateway on an IoT-enabled device for product dispensing or unlocking a service after payment, the ESP32 microcontroller plays a central role in managing and controlling the entire process. The setup of the ESP32 was carefully planned and implemented to ensure smooth operation across multiple components. Here's a breakdown of how the ESP32 was set up:

1. Hardware Configuration:

The ESP32 is equipped with built-in Wi-Fi, making it ideal for IoT-based applications. It was connected to peripheral components like the TFT Display, push buttons, buzzer, and the product dispensing mechanism. The GPIO (General Purpose Input/Output) pins of the ESP32 were mapped to control each of these components. For example, specific pins were designated for reading input from the push buttons, triggering the product dispensing mechanism and driving the buzzer.

2. Wi-Fi Setup:

Since the ESP32 needed to communicate with Google Firebase, establishing a stable Wi-Fi connection was a priority. The Wi-Fi libraries in the ESP32's firmware were configured to connect to a secure network, allowing it to interact with the cloud-based Firebase database in real-time. This was achieved through simple Wi-Fi commands that enabled the ESP32 to send and receive data over the internet.

3. Firmware Development:

The ESP32 was programmed using the Arduino IDE, with custom firmware developed to handle various tasks. This firmware was responsible for:

- Reading user inputs from the push buttons.
- Displaying the selected product and QR code on the TFT screen.
- Establishing communication with Firebase for real-time payment verification using Razorpay as the main payment gateway.
- Utilising Razorpay's webhook feature to send payment information to Firebase, where it is stored and monitored by the ESP32.
- Triggering the product dispensing mechanism once the payment is confirmed as successful.
- Providing audio feedback via the buzzer.

The code was structured to handle these tasks concurrently without interference, ensuring the system could process actions in real-time.

4. Cloud Communication:

The ESP32's firmware was integrated with Firebase using the Firebase ESP32 library, which enabled real-time synchronization of data such as transaction status and user actions. The ESP32 monitors the payment information from Firebase; if the payment is successful, it triggers the product dispensing action. If the payment fails, no action is taken.

5. Testing and Debugging:

During the setup phase, the ESP32 was repeatedly tested to ensure reliable communication with both Firebase and Razorpay. Debugging tools in the Arduino IDE, along with serial print statements were employed to monitor data flow and troubleshoot any issues in communication or logic.

4.4 Cashless Transaction Via Razorpay

The integration of Razorpay as the payment gateway enables the system to process secure, cashless transactions. Users are provided with a dynamic QR code on the TFT screen, linked to their selected product and payment amount. This QR code is scanned via mobile devices using UPI, simplifying the payment process. Once the payment is initiated, Razorpay ensures the transaction is securely processed, eliminating the need for physical currency handling. Additionally, integration with Google Firebase facilitates real-time communication between the ESP32 and Razorpay, ensuring that the system accurately verifies payments and reflects transaction status without delay. This results in a smooth, user-friendly payment process.

4.5 Evidence of Payment Verification

A standout feature of the system is its ability to verify payments in real-time using Google Firebase as a cloud-based database. Firebase updates instantly once the transaction is complete, sending confirmation to the ESP32 microcontroller, which then triggers the product dispensing mechanism if the payment is successful. The entire process from payment to dispensing occurs within seconds, ensuring that users do not face unnecessary delays. This integration ensures that transactions are securely validated before any product is dispensed, preventing accidental or fraudulent dispensations.

1. Successful Transaction Data

For successful transactions, Firebase receives the transaction data and once the payment status is verified as successful, the system triggers the product dispensing

action. Below figure 4.7 is a snapshot of the data retrieved from Firebase for a successful transaction:

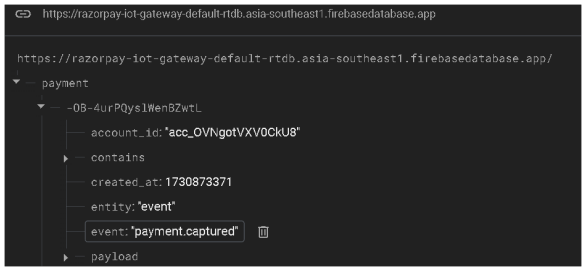

Figure 4.7: Payment Captured Status

This data is retrieved in real-time by the ESP32 and used to trigger the product dispensing mechanism. When the payment status is captured as 'payment.success', the payment is considered valid, and the dispensing action is triggered. This seamless payment verification process ensures that only successful payments lead to the dispensing of products, preventing any errors or fraudulent transactions.

2. Failed Transaction Data

For failed transactions, the data is captured and stored in Firebase which is shown in figure 4.8 below:

Figure 4.8: Payment Failed Status

When the payment status is captured as 'payment.failed', the system recognizes the transaction as unsuccessful. In this case, no dispensing action is triggered, ensuring that no product is dispensed without a valid payment. This process helps maintain the integrity and security of the system by preventing fraudulent or incomplete transactions.

4.6 System Performance

The system demonstrates consistent and reliable performance throughout its operations. The Wi-Fi communication between the ESP32, Firebase and Razorpay is stable, ensuring that all components work in sync without any major connectivity issues. The delay between product selection, payment initiation and product dispensing is minimal, usually taking only a few seconds to complete all actions. This fast processing time enhances user satisfaction by ensuring that there are no long waits for transactions to complete. The system's performance under various conditions, including multiple users and simultaneous transactions, proves its efficiency and scalability for practical use in automated vending machines or similar applications.

4.7 Major Milestones and Issues

Throughout the development of the IoT-enabled payment gateway system, several key milestones were achieved, along with challenges that had to be overcome to ensure the system functioned as expected.

Wi-Fi and Firebase Communication: One of the initial hurdles was establishing stable communication between the ESP32 and Firebase. A reliable Wi-Fi connection was essential for real-time payment verification. The first challenge encountered was Wi-Fi instability, which was resolved by optimizing network configurations, ensuring proper authentication, and ensuring the network was secure. Once stable, the ESP32 was able to reliably send and receive data from Firebase, which was crucial for payment status updates and product dispensing actions.

Razorpay Integration: Integrating Razorpay into the system for secure payment processing was a major milestone. However, handling real-time webhook data from Razorpay posed challenges. The payment status from Razorpay needed to be accurately fetched and relayed to Firebase for validation, which required continuous testing and debugging. Ensuring that the system could efficiently handle the webhook data and communicate with Firebase without delays or data inconsistencies was critical for seamless payment verification.

User Interface Design: The development of the user interface was another essential aspect of the project. The TFT display had to show clear instructions, QR codes for payments, and real-time payment updates. One of the challenges here was ensuring the interface was intuitive and responsive, especially in dynamically displaying QR codes based on the selected product amount.

Product Dispensing Mechanism: Ensuring that the product dispensing mechanism was triggered appropriately after a successful payment was one of the most

critical milestones. The synchronization between hardware and software was vital to guarantee no delays between payment confirmation and product dispensing. This process involved real-time communication between the ESP32 and Firebase, ensuring the transaction status was correctly verified before triggering the dispensing action.

Project Progression: The project started with a simple task of lighting up an LED to confirm a successful transaction. This served as a foundational step before moving on to more complex tasks. The next major step was dynamically displaying the QR code on the TFT display based on the payment amount and ensuring that payment verification occurred accurately. Once the transaction details were confirmed, attention shifted to automating the product dispensing mechanism, which was successfully implemented.

Through iterative testing and debugging, the system was refined to ensure it performed consistently, resulting in a fully functional IoT-enabled payment gateway system with seamless payment processing and product dispensing.

Chapter 5
Conclusion and Future Work

This project demonstrates the successful development of a payment gateway on an IoT-enabled device and vending machine, showcasing the system's potential. It integrates modern technologies to enable efficient, seamless cashless transactions. The system is built around the ESP32 microcontroller, managing core functions like product selection, payment processing, and dispensing. Key hardware components, including push buttons, a TFT LCD, a product dispensing mechanism, and a buzzer, create an intuitive user experience. Cloud-based services like Google Firebase handle real-time transaction management, while Razorpay's payment gateway facilitates secure, dynamic QR code generation for UPI or mobile wallet payments.

The process starts with the user selecting a product via a push-button interface, with the selected product displayed on the TFT screen. Pre-generated QR codes stored in the ESP32 allow users to scan and complete payments. Razorpay sends payment information to Firebase, which the ESP32 monitors for updates. Upon payment confirmation, the ESP32 triggers the dispensing mechanism. The system was developed using a structured methodology, ensuring smooth communication between ESP32, Firebase, and Razorpay, with Wi-Fi connectivity and HTTP security. Custom firmware was created for the ESP32 to manage all functions, while Firebase handled transaction data and Razorpay processed payments. The system was rigorously tested to ensure minimal delays in payment processing and dispensing. Each step, from product selection to dispensing, is completed efficiently, highlighting the system's real-world viability. The user interface, with real-time feedback, ensures ease of interaction.

This project integrates IoT and secure payment processing to streamline cashless transactions for automated product dispensing. Combining QR codes, secure payment gateways, and an IoT microcontroller enables seamless, contactless transactions, enhancing user convenience. This approach demonstrates the potential for IoT-enabled payment gateways in various systems.

As for future work, several enhancements can be explored, such as:

- Adding support for additional gateways would provide more payment options, enhancing accessibility and convenience.
- Expanding the system to handle a wider variety of products or applying it in areas like retail kiosks or automated counters would increase its commercial

potential across IoT-driven applications.

- Adding stronger error-handling features would help the system manage problems like lost connections, payment issues, or device malfunctions more effectively, making it more reliable.

Bibliography

[1] G. Shini, J. L. F. Daya, and P. Balamurugan, "Iot-based real-time monitoring of supercapacitors used in electric vehicles," *Journal of Applied Research and Technology*, vol. 22, 2024.

[2] J. Patel, "Secured and efficient payment gateways for ecommerce," *International Journal of Research Publication and Reviews*, vol. 2, no. 7, 2021.

[3] M. Omer, S. J. Ali, S. M. Raza, D. T. Le, and H. Choo, "Real-time object detection with iot using a smart cart," in *Proceedings of the 2024 18th International Conference on Ubiquitous Information Management and Communication, IMCOM 2024*, 2024.

[4] M. N. M. B. et al., "Towards secure iot-based payments by extension of payment card industry data security standard (pci dss)," *Wireless Communications and Mobile Computing*, vol. 2022, 2022.

[5] R. Calabro, E. Kemps, I. Prichard, and M. Tiggemann, "Vending machine backgrounds: nudging healthier beverage choices," *Current Psychology*, vol. 43, no. 2, 2024.

[6] H. Mu'min, D. Dharmayanti, and F. A. Rizky, "The influence of payment gateways and ease of shopping on customer satisfaction at the up2beat marketplace," *Journal Transnational Universal Studies*, vol. 2, no. 2, 2024.

[7] R. Jadhv, M. Jejurkar, and P. Kave, "Smart coffee vending machine using rfid," *Advances in Wireless and Mobile Communications*, vol. 10, no. 4, 2017.

[8] J. Paruvathavardhini, S. Bhuvaneswari, C. Kavitha, and A. Mythily, "Automatic vending machine using rfid," *International Journal of Engineering Research & Technology*, vol. 9, no. 10, pp. 20–24, 2021. [Online]. Available: www.ijert.org

[9] I. Journal, "A study to know impact of ai on crm," *International Journal of Scientific Research in Engineering and Management*, vol. 08, no. 02, 2024.

[10] B. Sharma, R. Singhvi, P. Chauhan, and N. Naik, "A study to know the role of ai and sustainability in agriculture," *International Journal of Scientific Research in Engineering and Management*, vol. 08, no. 01, 2024.

[11] K. Hilyati, W. Gata, E. H. Hermalani, A. Bayhaqy, and F. Frieyadie, "Reserve vending machine food waste sebagai deposito melalui qris bank sampah rumah tangga," *Jurnal Ilmiah Informatika*, vol. 10, no. 01, 2022.

[12] G. Bardwell, A. Ivsins, J. R. Wallace, M. Mansoor, and T. Kerr, ""the machine doesn't judge": Counternarratives on surveillance among people accessing a safer opioid supply via biometric machines," *Social Science & Medicine*, vol. 345, 2024.

[13] I. M. Savaniu, A. P. Chiriță, O. Tonciu, M. Culcea, and A. Neagu, "Neural-network-based time control for microwave oven heating of food products distributed by a solar-powered vending machine with energy management considerations," *Energies (Basel)*, vol. 16, no. 19, 2023.

[14] R. Calotă, M. Savaniu, A. Girip, I. Năstase, M. R. Georgescu, and O. Tonciu, "Study on energy efficiency of an off-grid vending machine with compact heat exchangers and low gwp refrigerant powered by solar energy," *Energies (Basel)*, vol. 15, no. 12, 2022.

[15] C. J. Sampayo-Rodriguez, G. Castillo-Quiroz, A. Hernández-Luna, and I. Cabrera-Hernández, "Design and construction of a token vending machine for wireless internet connection," *Revista de la Invención Técnica*, 2022.

[16] , "Conclusion of transactions using blockchain technology in the digital space," *Eurasian Advocacy (Evraziiskaya Advokatura)*, no. 4(63), 2023.

[17] S. Henriques, S. Lewis, and G. Kotian, "An iot-based vending machine using blockchain for enhanced security," *Journal of ISMAC*, vol. 4, no. 3, 2022.

[18] R. Harish, S. C. Menon, and A. K. Tyagi, "The role of artificial intelligence, blockchain, and internet of things in next generation machine based communication," in *Robotic Process Automation*, 2023.

[19] J. Li, F. Tang, C. Zhu, S. He, S. Zhang, and Y. Su, "Bp-yolo: A real-time product detection and shopping behaviors recognition model for intelligent unmanned vending machine," *IEEE Access*, vol. 12, 2024.

[20] F. Manzano-Agugliaro, M. Chihib, M. Chourak, J. A. Martínez, A. J. Zapata-Sierra, and A. Alcayde, "Monitoring energy consumption of vending machines in university buildings," *Energy Reports*, vol. 10, 2023.

[21] D. Maulana, H. Nasution, and M. A. Nasution, "Snack vending machine based on iot using mqtt protocol and real-time monitoring system for university campus," *Jurnal Teknik Informatika*, vol. 9, 2023.

[22] W. Alam, D. Sarma, R. J. Chakma, M. J. Alam, and S. Hossain, "Internet of things based smart vending machine using digital payment system," *Indonesian Journal of Electrical Engineering and Informatics*, vol. 9, no. 3, 2021.

[23] S. Bojjagani, P. V. V. Rao, D. R. Vemula, B. R. Reddy, and T. J. Lakshmi, "A secure iot-based micro-payment protocol for wearable devices," *Peer Peer Netw Appl*, vol. 15, no. 2, 2022.

[24] H. R. Lee, W. J. Kim, K. H. Park, H. J. Cho, and C. H. Lin, "Development of an easy payment system based on iot gateway," in *International Conference on Electronics, Information and Communication, ICEIC 2018*, 2018.

[25] D. Hercog, T. Lerher, M. Truntič, and O. Težak, "Design and implementation of esp32-based iot devices," *Sensors*, vol. 23, no. 15, 2023.

[26] V. Manda, V. Kumar, B. Sarat, and A. Info, "Iot based sanitary pad vending machine," *International Journal for Modern Trends in Science and Technology*, vol. 7, 2021.

[27] S. Wang, Y. Hou, F. Gao, and X. Ji, "A novel iot access architecture for vehicle monitoring system," in *2016 IEEE 3rd World Forum on Internet of Things, WF-IoT 2016*, 2016.

Appendix A
Hardware Details

A.1 ESP32 and Pinout

The ESP32 Wroom 32 serves as the central microcontroller for this project, responsible for managing communication between user inputs, Google Firebase and the Razorpay payment gateway. This development board combines powerful Wi-Fi and Bluetooth capabilities with a compact 30-pin design as depcited in figure A.1, makes it ideal for IoT projects. It supports AP, STA and AP+STA wireless modes, providing flexible network connectivity options to suit diverse applications. Equipped with dual CPU cores, adjustable clock frequencies and numerous peripherals like UART, I2C and SPI, the ESP32 allows for smooth data transmission and processing. This versatile microcontroller is used here to continuously monitor Firebase for real-time updates on payment status and, once a transaction is confirmed, trigger the dispens ing mechanism, making it a crucial component for the system's secure, automated operations.

Specifications:

- Integrated Crystal 40 MHz
- Module Interfaces: UART, SPI, I2C, PWM, ADC, DAC, GPIO
- Integrated SPI Flash: 4 MB
- ROM: 448 KB (for booting and core functions)
- SRAM: 520 KB
- Integrated Connectivity Protocols: Wi-Fi, Bluetooth, BLE
- Operating Temperature Range: 40-85°C
- Operating Voltage: 3.3V
- Operating Current: 80 mA (average)

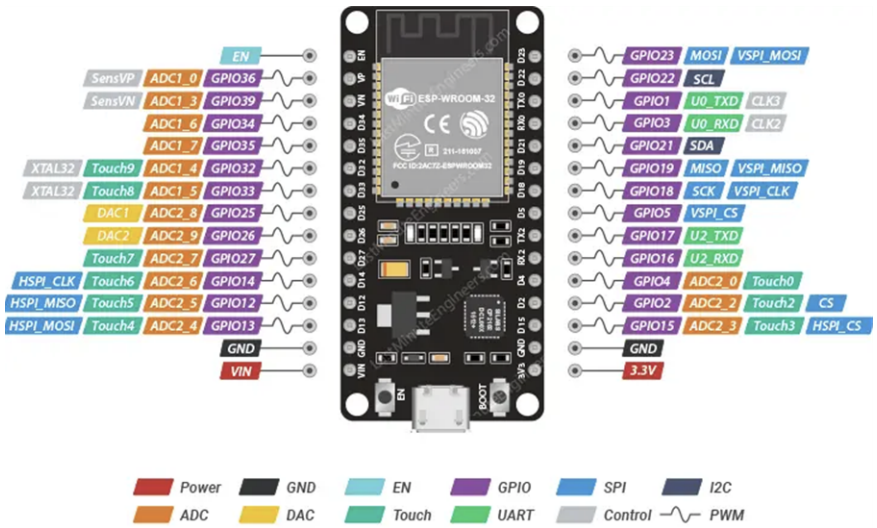

Figure A.1: ESP32 Pinout

A.2 2.4-inch SPI TFT LCD Display Module

The project incorporates a 2.4-inch SPI TFT LCD display module, which serves as the user interface for product selection and payment confirmation. This display as shown in figure A.2 has a 240×320 resolution and a wide viewing angle, provides clear visibility of the QR code and payment status. The TFT display is controlled via the ESP32 and connects using only five pins (CS, RS, SCL, SDA and RST), which optimizes the use of I/O resources. The display shows the QR code generated by Razorpay for each transaction, guiding users through the payment process and confirming successful or failed payments.

Specifications:

- 2.4-inch SPI TFT LCD Module
- 10-pin interface
- Includes an SD card slot
- LCD Driver IC: ILI9341
- Resolution: 240 × 320
- Wide viewing angle with suitable contrast
- Serial display interface requiring 5 wires: CS, RS, SCL, SDA, RST
- SD card uses hardware SPI interface (CS / MOSI / MISO / SCK)

Figure A.2: TFT Display

A.3 4-Channel DC 12V Relay Module

The project utilizes a 4-channel DC 12V relay module as shown in figure A.3 to control the motors that perform the product dispensing. This module is capable of managing high-current loads safely by isolating the low-current signals from the microcontroller (ESP32) through an SMD optocoupler, ensuring robust safety. Each relay operates with TTL logic signals and features LEDs for clear status indication. It provides both Normally Open (NO) and Normally Closed (NC) contacts, making it versatile for various control configurations. With a trigger current of at least 5mA per channel, the module is ideal for reliably activating motors and other high-current components within the system.

Specifications:

- Number of Relays: 4
- Working Voltage: 12V DC
- DC Control Signal: TTL level
- Rated Load: 7A/250VAC; 10A/125VAC; 10A/28VDC
- Contact Action Time: 10ms/5ms
- Current: 15-20mA per channel

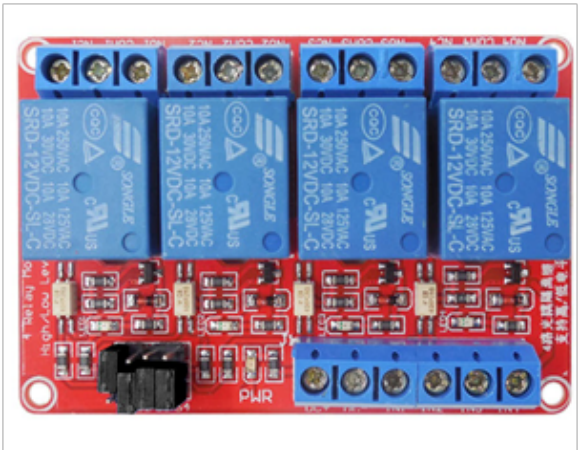

Figure A.3: 4 Channel Relay

A.4 DC Motor

The 12V DC geared motor used in this project as shown in figure A.4 is designed to operate efficiently at 100 RPM, making it suitable for precise, low-speed applications like product dis pensing in vending systems. This motor features a centre shaft with durable metal gearing, providing reliable performance even under demanding conditions. The motor is triggered through a relay controlled by the ESP32, allowing seamless integration into the system. With a voltage range between 5V and 35V, it offers flexibility for a variety of power sources. The motor's shaft includes a 3 mm threaded drill hole, enabling easy mounting to wheels or other mechanical structures, making it ideal for automated product vending or similar applications.

Figure A.4: 12V DC Motor

Appendix B
Software Details

B.1 Firebase Setup

The figure B.1 displays the Firebase project dashboard sidebar, highlighting essential services such as authentication and realtime database. The authentication service secures user access, ensuring that only verified users can make payments or interact with the IoT device. Meanwhile, the realtime database stores real-time transaction data, enabling the IoT device (e.g., ESP32) to monitor payment statuses and trigger specific actions, such as dispensing a product, once payment confirmation is received. Configuring these services in Firebase is crucial for seamless integration and realtime synchronization between the payment gateway and the IoT device.

The figure B.2 shows the Firebase Console's main dashboard, where you can initiate the creation of a new project by selecting Create a project. This step is essential for setting up the backend for the IoT-enabled payment gateway system, as Firebase will act as the central hub for managing user authentication, payment data and real-time transaction records. Once the project is created, necessary services like the realtime Database and Authentication can be enabled to support the project's functionality.

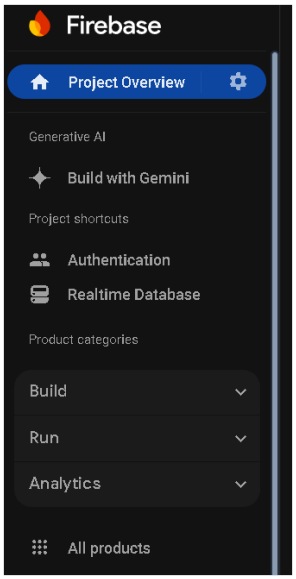

Figure B.1: Dashboard Sidebar

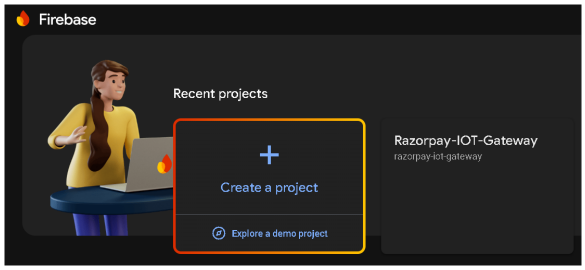

Figure B.2: Firebase Dashboard

To enable seamless interaction between the IoT device and Firebase, anonymous authentication was configured. This setup allows users to interact with the device without the need for a full login, streamlining the user experience while still providing Firebase with unique session identifiers. Additionally, realtime Database rules were set to allow read and write permissions for the ESP32. This configuration enables the device to access and update transaction data in real time, ensuring it can monitor payment statuses and execute automated actions, such as product dispensing, upon payment confirmation. This setup creates a responsive, secure and user-friendly backend that supports the project's objective of automating actions based on transaction outcomes as shown in figure B.3. and B.4.

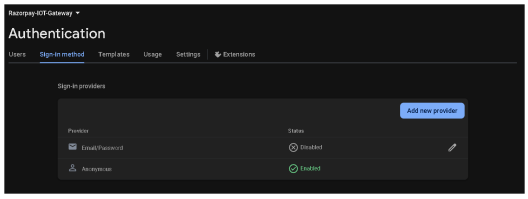

Figure B.3: Authentication Setup

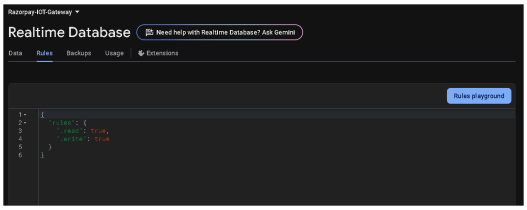

Figure B.4: Rules Setup