RAPIDBILL FUSION

Capstone Project Report

End-Semester Evaluation

Submitted by:

(102103421) Aayushi Bareja

(102117056) Harshvir Singh

(102103799) Manvendra Raj Singh

(102103809) Tanmay Relan

(102115189) Twiny

BE Fourth Year

CPG No. 11

Under the Mentorship of

Dr. Ashima Singh

Associate Professor, CSED

Computer Science and Engineering Department

Thapar Institute of Engineering and Technology, Patiala

December 2024

ABSTRACT

The RapidBill Fusion project is poised to revolutionize the retail shopping experience by introducing an AI-powered autonomous shopping and checkout system. By utilizing advanced computer vision the system automates the checkout process, drastically reducing the need for human interaction and cutting down on waiting times. Items are automatically identified and billed in real-time with high accuracy, drawing from a pre-uploaded price list. This seamless integration of technology ensures that customers enjoy a faster, more efficient shopping experience without the hassle of standing in queues.

At the heart of RapidBill Fusion is a user-friendly interface that allows customers to complete payments effortlessly through a web based platform. The system not only enhances operational efficiency for retailers but also redefines convenience in the retail environment. By streamlining transactions and providing precise billing, RapidBill Fusion sets a new standard for autonomous shopping systems, paving the way for a future where checkout lines are obsolete and customer satisfaction is at its peak.

DECLARATION

We hereby declare that the design principles and working prototype model of the project entitled **RapidBill Fusion** is an authentic record of our own work carried out in the Computer Science and Engineering Department, TIET, Patiala, under the guidance of Dr. Ashima Singh during 6th and 7th semester (2024).

Date: 18 December 2024

Roll No	**Name**	**Signatures**
102103421	Aayushi Bareja	
102117056	Harshvir Singh	
102103799	Manvendra Raj Singh	
102103809	Tanmay Relan	
102115189	Twiny	

Counter Signed By:

Faculty Mentor:

Dr. Ashima Singh

Associate Professor

CSED

TIET, Patiala

ACKNOWLEDGEMENT

We would like to express our thanks to our mentor Dr. Ashima Singh. She has been of great help in our venture and an indispensable resource of technical knowledge. She is truly an amazing mentor to have.

We are also thankful to Dr. Shalini Batra, Head, Computer Science and Engineering Department, the entire faculty and staff of the Computer Science and Engineering Department, and our friends who devoted their valuable time and helped us in all possible ways towards successful completion of this project. We thank all those who have contributed either directly or indirectly towards this project.

Lastly, we would also like to thank our families for their unyielding love and encouragement.

They always wanted the best for us and we admire their determination and sacrifice.

Date: 18 December 2024

Roll No	Name	Signatures
102103421	Aayushi Bareja	
102117056	Harshvir Singh	
102103799	Manvendra Raj Singh	
102103809	Tanmay Relan	
102115189	Twiny	

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

1.1 Project Overview

In The RapidBill Fusion project aims to revolutionize the retail shopping experience with its AI-powered autonomous shopping and checkout system. By harnessing the capabilities of computer vision, the system promises to streamline the checkout process, minimizing human interactions and reducing waiting times for customers. Through the seamless integration of technology, users can expect a faster and more efficient shopping experience, where items are automatically identified and billed with precision without any hassle of standing in queues.

At the core of the solution lies the sophisticated use of computer vision to accurately recognize items in real-time. This enables the system to generate detailed bills swiftly by referencing a pre-uploaded price list, ensuring accuracy in transactions. The user interface is designed to be intuitive and user-friendly, allowing customers to complete the payment process seamlessly through a web-based platform. With RapidBill Fusion, customers can expect a hassle-free shopping experience, where checkout becomes a breeze, and interactions with staff are minimized, enhancing convenience and efficiency.

The ultimate outcome of the RapidBill Fusion project is a comprehensive billing and checkout system that redefines the retail landscape. By using cutting-edge technology, the system not only improves operational efficiency for retailers but also enhances the overall shopping experience for customers. With its ability to seamlessly process transactions and provide accurate billing, RapidBill Fusion sets a new standard for autonomous shopping systems, paving the way for a future where checkout lines are a thing of the past, and convenience reigns supreme in retail environments.

1.2 Need Analysis

India In today's fast-paced retail landscape, there is an increasing demand for innovative solutions that enhance efficiency and convenience for both retailers and customers. The proposed project, RapidBill Fusion, addresses this need by introducing an AI-powered autonomous shopping and checkout system that revolutionizes the traditional retail experience. With the advent of technology and shifting consumer preferences, there is a growing desire for seamless and frictionless transactions, which RapidBill Fusion aims to fulfil. Existing systems and research papers highlight the importance of incorporating computer vision technologies in retail environments to improve operational efficiency and customer satisfaction. For example, studies have shown the effectiveness of computer vision in automating tasks such as product recognition, leading to reduced labour costs and increased accuracy.

RapidBill Fusion's relevance in the real world extends beyond just enhancing the shopping experience; it also addresses current challenges faced by retailers, such as long checkout lines and inefficient transaction processes. By automating the checkout process and leveraging computer vision to accurately identify items, RapidBill Fusion not only streamlines operations but also minimizes human interactions, particularly crucial in today's context of health and safety concerns.

Moreover, the project aligns with broader industry trends towards digital transformation and automation, where retailers are increasingly embracing technology to stay competitive and meet evolving consumer expectations. RapidBill Fusion offers a practical solution that not only improves operational efficiency for retailers but also enhances the overall shopping experience for customers, thereby positioning itself as a relevant and impactful innovation in the retail industry.

1.3 Research Gaps

1. **Real-Time AI Integration:** Limited research on seamless real-time AI-based object detection with low latency on edge devices.
2. **Detection Accuracy:** Challenges remain in handling low lighting, object occlusion, and visually similar products in retail environments.
3. **Scalability:** Insufficient studies on scaling systems for large retail stores with high product volumes and customer throughput.
4. **Edge Computing Limitations:** Research is limited on balancing edge and cloud computing to optimize AI performance on hardware like Raspberry Pi.
5. **User Experience:** Lack of focus on intuitive user interfaces, seamless payment flows, and accessibility for all users.
6. **Dynamic Updates:** Limited solutions for real-time admin updates to product details, pricing, and inventory.
7. **Data Security:** Insufficient studies on securing transaction data and ensuring privacy in AI and IoT systems.
8. **Energy Efficiency:** Limited research on optimizing energy consumption for AI-based edge processing devices.
9. **Customer Feedback:** Gaps in utilizing real-time feedback to improve detection accuracy and system adaptability.

1.4 Problem Definition and Scope

Problem Definition: Current automated billing systems and smart shopping carts face issues such as inefficiencies in processing, inaccuracies in item recognition, high implementation costs, and challenges in user experience, particularly for diverse demographics. Additionally, data security concerns and limited scalability hinder widespread adoption and effectiveness.

Scope of the Project: The RapidBill Fusion project aims to develop an AI-powered autonomous shopping and checkout system that addresses these challenges. It focuses on integrating advanced computer vision and AI technologies to streamline checkout processes, reduce human interaction, and minimize waiting times. The system will feature real-time item recognition, accurate billing, and a user-friendly web-based interface. The project also aims to reduce costs, and ensure scalability and adaptability across various retail environments. Evaluation will include performance metrics and user feedback to refine and improve the system further.

1.5 Assumptions and Constraints

S. No	Assumption and Constraints
1.	The system will rely on advanced computer vision technologies for accurate item recognition.
2.	The infrastructure (e.g., cameras, sensors) will remain functional and calibrated to support real-time processing.
3.	Retailers will provide a pre-uploaded, accurate price list for all products in the system database.
4.	A stable internet connection is assumed for seamless data synchronization between the local system and cloud servers.

Table 1.5: Assumptions and Constraints

1.6 Standards

Phase or Activity Group	Number	Standard Title
Code and Control System Standards	PEP8	Standard for Python Code
	ISO 13489	Guide for Safety of Machinery
Requirement Specifications	IEEE 830	Recommended practice for software specification
	IEEE 1233	Guide for developing system requirement specification
Design Standards	IEEE 1016	Software Design Descriptions
	IEEE 1062	Software acquisition
Implementation, acquisition, and tools Standards	IEEE 1462	Guidelines for evaluation and selection of CASE tools
Testing Standards	IEEE 829	Software test documentation

	IEEE 1012	Software verification and validation

Table 1.6: Standards Implemented

1.7 Approved Objectives

1. To develop an AI-powered autonomous shopping and checkout system using computer vision .
2. To create a user-friendly web application for seamless payment and item recognition.
3. To streamline the checkout process by eliminating traditional checkout lines and reducing waiting times.
4. To implement secure transaction processing and data protection in compliance with privacy regulations.

1.8 Methodology

The RapidBill Fusion will integrate the capabilities of computer vision to create a flexible autonomous checkout system. This checkout will be able to identify the variety of products that will have a corresponding price in a custom-made catalogue. The checkout system will employ object detection algorithms to classify these products. The products will have their prices stored in the catalogue database.

The camera and sensors will help the algorithm identify the specific product that is placed in the cart and will automatically add that item to the user interface of the web/app interface, which will be integrated along with our product. The system will analyse the weight of the product placed in the cart at the moment.

We will also create a web interactive user interface that will be connected to the catalogue database in order to get the price for the corresponding item, and this web interface will also have a payment gateway attached to it, which will enable the user to have a hassle-free experience. This web interface will keep adding items to the checkout list until the user proceeds to the payment option. Once the checkout is done, the user will be redirected to a payment gateway. After the payment is successful, the user will get an email receipt with the provided email address.

1.9 Project Outcomes and Deliverables

- **AI-Powered Checkout System:** A system that uses AI and computer vision for real-time item identification and billing.
- **User Interface:** A user-friendly web interface for seamless payment and system interaction.
- **Accurate Billing:** A reliable billing system that ensures precise transactions with a pre-uploaded price list..
- **Scalable Architecture:** A system designed to handle varying transaction volumes and integrate with existing retail setups.
- **Training and Support:** Comprehensive training materials and technical support for effective system use and implementation.

1.10 Novelty of Work

This project's uniqueness lies in its modern approach to enhancing the retail shopping experience. Unlike existing systems, RapidBill Fusion seamlessly integrates advanced technology with operational efficiency to deliver a fully autonomous shopping and checkout solution.

The system employs cutting-edge computer vision techniques to automate product identification and billing, eliminating the need for traditional barcode scanning. This ensures high accuracy and efficiency in real-time item recognition, while also accommodating unique or non-standardized product packaging, making it versatile across various retail environments.

A web-based platform offers customers a user-friendly interface for effortless interaction, enabling seamless checkout and payment processes. The integration of real-time billing with pre-uploaded price lists ensures precise transactions, drastically reducing human errors and enhancing trust in the system’s reliability.

By blending advanced AI, operational scalability, and customer-centric design, RapidBill Fusion sets a new benchmark for autonomous retail systems, paving the way for a contactless and efficient shopping future while enabling retailers to stay competitive in a rapidly evolving industry.

REQUIREMENT ANALYSIS

2.1 Literature Survey

2.1.1 Theory Associated with Problem Area

The inefficiencies in traditional retail systems, such as long checkout times, manual errors, can be addressed through advancements in AI, computer vision, and automation. AI enables real-time product recognition and billing, reducing human dependency, while computer vision eliminates the limitations of barcode systems.

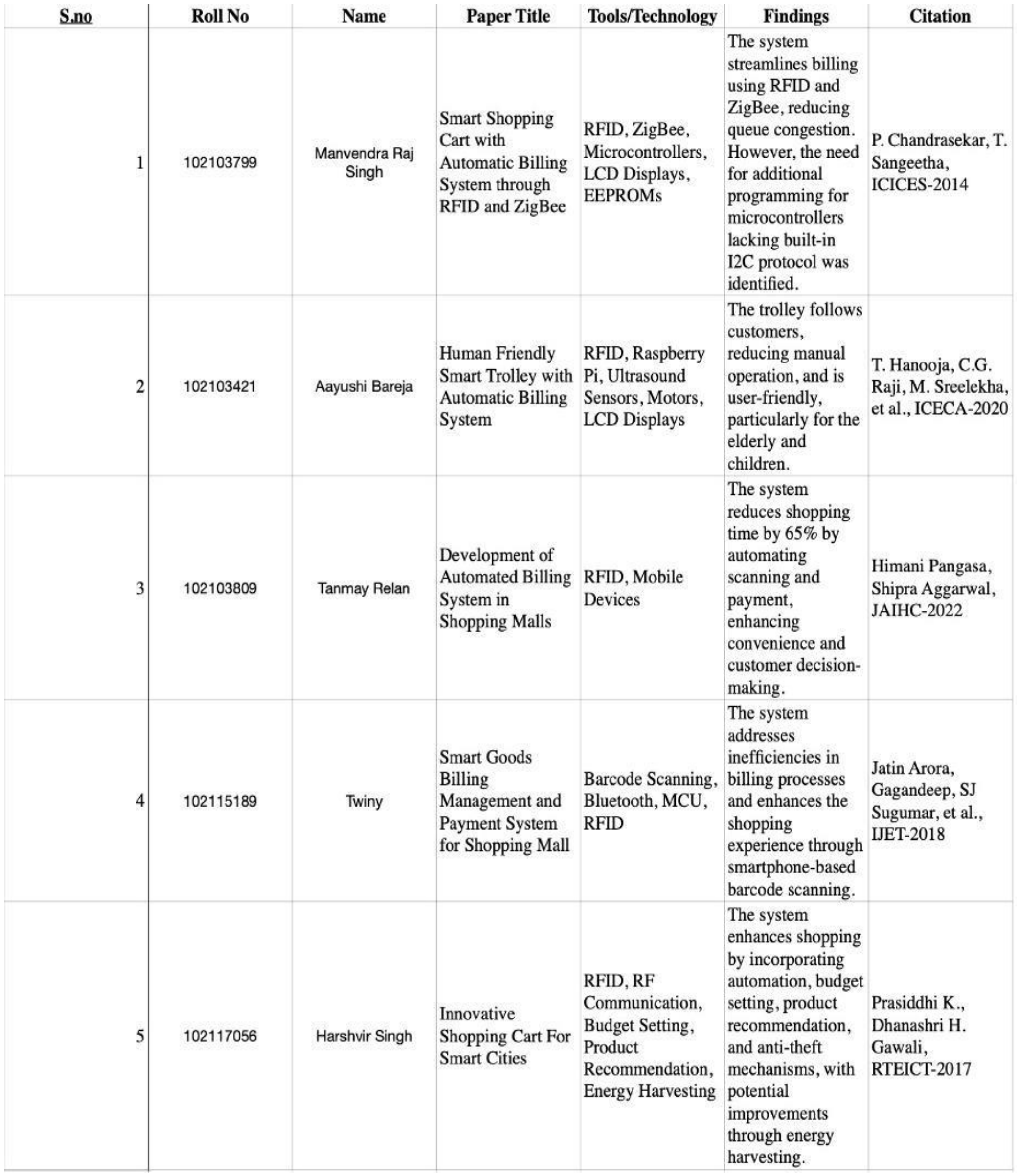
Queuing theory highlights how delays at checkouts reduce customer satisfaction, which autonomous systems mitigate by streamlining processes. Consumer behaviour theories emphasize the growing demand for seamless and contactless shopping experiences, which RapidBill Fusion addresses.

2.1.2 Existing Systems and Solutions

Current Existing retail systems primarily rely on barcode scanning, manual checkout processes, and basic inventory tools, leading to inefficiencies like delays, human errors, and inadequate stock management. Several advancements have been introduced, but they still have limitations. Barcode and RFID-based systems are effective for product identification and billing but struggle with damaged tags or non-standard packaging. Smart shopping carts that automate billing through sensors and RFID are costly and require significant infrastructure changes, making them impractical for widespread use. Mobile self-checkout apps offer convenience, but they rely heavily on user accuracy, which can lead to errors. Camera-based checkout systems using AI for product recognition are emerging, but they face challenges such as high costs and integration difficulties with existing setups. Despite these innovations, most systems address only specific issues and do not offer a comprehensive solution. RapidBill Fusion addresses these challenges by integrating AI and real-time data into a scalable, cost-effective platform, providing a seamless and efficient retail experience.

2.1.3 Research Findings for Existing Literature

To get the necessary knowledge about our project, various existing literary documents and products were studied. Some important observations made are listed below:



S.no	Roll No	Name	Paper Title	Tools/Technology	Findings	Citation
1	102103799	Manvendra Raj Singh	Smart Shopping Cart with Automatic Billing System through RFID and ZigBee	RFID, ZigBee, Microcontrollers, LCD Displays, EEPROMs	The system streamlines billing using RFID and ZigBee, reducing queue congestion. However, the need for additional programming for microcontrollers lacking built-in I2C protocol was identified.	P. Chandrasekar, T. Sangeetha, ICICES-2014
2	102103421	Aayushi Bareja	Human Friendly Smart Trolley with Automatic Billing System	RFID, Raspberry Pi, Ultrasound Sensors, Motors, LCD Displays	The trolley follows customers, reducing manual operation, and is user-friendly, particularly for the elderly and children.	T. Hanooja, C.G. Raji, M. Sreelekha, et al., ICECA-2020
3	102103809	Tanmay Relan	Development of Automated Billing System in Shopping Malls	RFID, Mobile Devices	The system reduces shopping time by 65% by automating scanning and payment, enhancing convenience and customer decision-making.	Himani Pangasa, Shipra Aggarwal, JAIHC-2022
4	102115189	Twiny	Smart Goods Billing Management and Payment System for Shopping Mall	Barcode Scanning, Bluetooth, MCU, RFID	The system addresses inefficiencies in billing processes and enhances the shopping experience through smartphone-based barcode scanning.	Jatin Arora, Gagandeep, SJ Sugumar, et al., IJET-2018
5	102117056	Harshvir Singh	Innovative Shopping Cart For Smart Cities	RFID, RF Communication, Budget Setting, Product Recommendation, Energy Harvesting	The system enhances shopping by incorporating automation, budget setting, product recommendation, and anti-theft mechanisms, with potential improvements through energy harvesting.	Prasiddhi K., Dhanashri H. Gawali, RTEICT-2017

Table 2.1.3: Related Literature Work

2.1.4 Problem Identified

In the realm of retail, several key issues hinder the efficiency and effectiveness of shopping and checkout processes. Traditional systems often lead to long waiting times, errors in billing, and poor inventory management. The main challenges include:

1. **Manual Checkout Delays**: Long queues at checkout counters are a significant problem, especially during peak hours. Customers have to wait for the cashier to scan each item manually, which creates a bottleneck and delays the overall shopping experience.
2. **Human Errors in Billing**: Traditional systems rely heavily on manual processes, which increases the likelihood of errors, such as incorrect item scanning, pricing mistakes, and missed discounts. These errors not only affect customer satisfaction but also lead to potential revenue loss for retailers.
3. **Inconsistent Customer Experience**: With varying levels of customer service and product availability, the shopping experience can be inconsistent, frustrating shoppers and leading to a negative impact on customer retention.
4. **Lack of Seamless Integration**: Many current systems don't integrate well with other retail tools, such as payment gateways, or customer loyalty programs, creating fragmented and inefficient workflows.

2.1.5 Survey of Tools and Technologies used

Several tools and technologies play a crucial role in improving the efficiency and functionality of modern retail systems, addressing issues like checkout automation, and real-time data analysis.

1. **IoT (Internet of Things)**: IoT technologies are widely used for tracking inventory, monitoring product availability, and enabling real-time updates on stock levels. Sensors and RFID tags are often employed in smart shelves and carts to automatically update inventory as products are added or removed.

2. **Computer Vision**: For automated product recognition, computer vision technology is employed to identify items through image recognition. This eliminates the need for barcode scanning and ensures quicker, more accurate item identification, even for untagged or irregularly packaged products.
3. **RFID Technology**: Radio Frequency Identification (RFID) is commonly used for product tracking and automated billing. RFID tags can be placed on products to enable real-time inventory tracking and prevent stock discrepancies.
4. **Payment Gateway Integration**: Secure payment gateways are essential for handling transactions in modern retail environments. These systems enable smooth, contactless payments via credit/debit cards, mobile wallets, and digital currencies, ensuring fast and secure checkouts.
5. **Web Applications:** Web applications are increasingly used for self-checkout, enabling customers to scan items using a web-based platform and complete payments directly through their browsers. These web apps are also integrated with customer loyalty programs and promotional features, providing a seamless and enhanced shopping experience across devices.

2.1.6 Summary

The RapidBill Fusion project builds on existing advancements in retail automation by integrating cutting-edge technologies like computer vision,and IoT to address the shortcomings of traditional checkout systems. It leverages the FOMO algorithm for real-time object detection, surpassing barcode and RFID systems by enabling accurate, tag-free product recognition. Unlike existing solutions, RapidBill Fusion incorporates edge devices such as the ESP32 Cam and Raspberry Pi, optimizing for cost-effectiveness and energy efficiency while ensuring scalability for both small retailers and large chain stores. A unique addition is the integration of weight sensors for real-time validation of product attributes, reducing detection errors and enhancing billing accuracy. The project differentiates itself further by offering an intuitive, web-based user interface coupled with WhatsApp integration for seamless digital bill delivery, promoting a queue-free and paperless shopping experience. Emphasizing security, scalability, and environmental sustainability, RapidBill Fusion redefines retail checkout by addressing the limitations of existing systems while offering a modern, efficient, and customer-centric solution**.**

2.2 Software Requirement Specifications

2.2.1 Introduction

2.2.1.1 Purpose

The purpose of the RapidBill Fusion system is to revolutionize the retail shopping and checkout process by leveraging advanced technologies such as artificial intelligence & computer vision. The system aims to automate product identification & billing thereby enhancing efficiency, accuracy, and convenience for both customers and retailers. By eliminating the need for manual scanning and reducing human errors, the system seeks to streamline the shopping experience, reduce wait times, and provide a seamless, contactless checkout process. Additionally, it aims to optimize inventory levels through real-time updates and predictive analytics, enabling better resource management and improving overall operational efficiency for retailers. Ultimately, RapidBill Fusion is designed to set a new standard for modern, autonomous retail systems, making shopping faster, more convenient, and more efficient.

2.2.1.2 Intended Audience

This document is intended for:

- **Development Team:** To understand the specific features, technical specifications, and design requirements for RapidBill Fusion.
- **Project Managers:** To track project progress and ensure alignment with objectives.
- **Retail Staff and Management:** To provide insights into the system's capabilities and ensure it addresses operational needs.
- **Stakeholders:** To review and confirm that the project meets organizational goals and expectations.

- **Quality Assurance Team:** To use the specified requirements as a basis for system testing and validation.

2.2.1.3 Project Scope

RapidBill Fusion is designed to revolutionize the retail checkout process through AI-powered automation. The project includes:

- **Automated Checkout System:** Streamlines the checkout process using computer vision to recognize items and calculate totals.
- **Real-Time Item Recognition:** Utilizes cameras and AI to accurately identify products and update the bill in real-time.
- **Seamless Payment Integration:** Supports various payment methods including credit/debit cards, mobile payments, and digital wallets.
- **User-Friendly Interface:** Provides a clear and intuitive interface for both retail staff and customers.

2.2.2 Overall Description

2.2.2.1 Product Perspective

RapidBill Fusion is envisioned as a cutting-edge retail checkout solution that integrates seamlessly with existing systems to enhance operational efficiency. The platform addresses common challenges in traditional checkout processes, such as long wait times and manual errors, by offering a unified, automated solution.

- **Current Retail Environment:** Traditional checkout systems are often manual and error-prone, leading to inefficiencies and customer dissatisfaction. RapidBill Fusion aims to replace these outdated systems with a modern, AI-driven solution that improves speed and accuracy.

- **Integration with Existing Systems:** The platform is designed to work with existing retail infrastructure, including point-of-sale systems and inventory management tools. It ensures a smooth transition and minimal disruption to current workflows.

- **Role in Retail Operations:** RapidBill Fusion modernizes the checkout experience, providing faster processing times, reduced errors, and enhanced customer satisfaction. The system's real-time item recognition and seamless payment integration streamline operations and improve overall efficiency.

- **User-Centric Approach:** The platform features a user-friendly interface tailored to the needs of both retail staff and customers. It ensures ease of use and promotes efficient adoption of the new system.

2.2.2.2 Product Features

The RapidBill Fusion offers a robust set of features tailored to revolutionize the retail shopping experience. These features focus on enhancing customer convenience, streamlining checkout processes, and ensuring secure and efficient operations, all while being cost-effective and particularly effective for handling raw materials like vegetables.

Key Features:

1. **Autonomous Shopping and Checkout:**
    - **AI-Powered Item Recognition:** Utilizes advanced computer vision algorithms to identify products in real-time as customers pick them from shelves.
    - **Seamless Checkout Experience:** Allows customers to complete purchases without waiting in line, using automated billing systems that charge them directly as they exit the store.
2. **Web Application Interface:**
    - **Payment Platform:** A web-based application that enables customers to make payments seamlessly for their purchases. Once the payment is completed, the system generates an e-bill and sends it directly to the customer via WhatsApp.
3. **Cost-Effective Design:**

    - The system is designed to reduce operational costs while maximizing efficiency, making it an affordable solution for businesses of all sizes.
    - Specialized for managing **raw materials like vegetables**, ensuring freshness and accuracy in tracking and handling.
4. **Scalability and Flexibility:**
    - The system is easily scalable to suit small retailers or large chain stores, ensuring adaptability to various business needs.
    - Modular design allows integration with existing systems, minimizing disruption during adoption.

2.2.3 External Interface Requirements

2.2.3.1 User Interface:

Web Application:

- The web application should have a user-friendly and intuitive interface, enabling users of all ages and technical backgrounds to interact with it effortlessly.

- It should be compatible with common web browsers (e.g., Chrome, Firefox, Edge) to ensure accessibility across a broad user base.

- The application must feature a real-time display of detected objects, including details like object name, price, weight, and accuracy, with a clearly visible button for bill generation.

- The billing process should be straightforward, guiding users seamlessly from object detection to bill generation and allowing easy download or WhatsApp sharing of the bill.

2.2.3.2 Hardware Interface:

Server Requirements:

- **Robust Architecture:** Needs strong servers for handling large transaction volumes, multiple concurrent users, and ensuring high availability with secure data processing and storage.
- **Scalability:** Must support future growth in user base and transaction data.

End-user Devices:

- **Device Compatibility:** Accessible on desktop/laptop computers, tablets with minimal hardware requirements.
- **POS Terminals:** Essential for processing transactions and updating inventory in real-time.

Peripheral Devices:

- **Item Billing Machine:** For quick checkout and billing.
- **Screen/Tablet:** For display purposes

2.2.3.3 Software Interfaces:

Operating Systems:

- **Compatibility:** Works with Windows, macOS and Linux for broad accessibility.

APIs and Integrations:

- **Payment Gateways:** Supports various payment methods using their APIs.

Third-Party Software:

- **Communication Tools:** Integrates with Whatsapp for sending shopping summary.

- **Customer Website:** An intuitive interface that allows customers to manage their shopping experience, from viewing to purchasing. The website prioritizes ease of use, with straightforward instructions and minimal steps required for common tasks.

2.2.4 Other Non-Functional Requirements

2.2.4.1 Performance Requirements:

- **Scalability:** RapidBill Fusion must handle an increasing number of transactions and users without performance degradation. The system should support both vertical scaling (upgrading existing hardware) and horizontal scaling (adding more servers) to manage growing demands.
- **Response Time:** The system should process item recognition and billing with minimal delay, aiming for a response time of under two seconds for each transaction to ensure a smooth customer experience.

- To ensure continuous availability, the system must maintain an uptime of 99.9%. This requires robust disaster recovery procedures and failover mechanisms to minimize downtime.

2.2.4.2 Safety Requirements:

- **Data Integrity:** The system must accurately record and update all transaction data, ensuring that billing information is error-free and consistent.
- **Emergency Management:** RapidBill Fusion should be capable of handling emergencies, such as system failures or overloads, with reliable procedures to manage crises and maintain operations.
- **User Training:** Comprehensive training must be provided to staff to ensure they can effectively use the system and respond to emergencies or technical issues.

2.2.4.3 Security Requirements:

- **Hardware Security:** To secure the ESP32 Cam and Raspberry Pi, implement secure boot to authenticate firmware during startup, and use tamper detection to identify unauthorized physical access. Data stored locally on the Raspberry Pi should be encrypted to prevent unauthorized access in case of device theft or compromise.
- **Network Security:** Secure all communication between devices using TLS to protect data in transit. A firewall should be configured on the Raspberry Pi to restrict unauthorized traffic, and the network should be segmented, isolating IoT devices from critical systems through VLANs or a dedicated IoT network.
- **API Security:** To secure APIs, implement rate limiting to prevent abuse and store API keys (e.g., Firebase and Twilio) securely using environment variables or hardware security modules. Input validation is critical to protect against injection attacks, ensuring the integrity of the system.
- **Secure Bill Transmission:** Use a trusted API provider like Twilio for WhatsApp integration, ensuring end-to-end encryption for sensitive bill details. Validate the user's WhatsApp number before sending any information to maintain privacy and prevent sending data to unintended recipients.

2.3 Cost Analysis

Item	**Quantity**	**Price (INR)**
Raspberry Pi 3 Model B	1	2000
ESP32 Camera Module	1	514
LED Strip	2	150
Load cell amplifier	1	100
Weight Sensor(0-5kg)	1	150
Total		**2914**

Table 2.3: Cost Analysis

2.4 Risk Analysis

The Risk Analysis identifies potential risks in the RapidBill Fusion project and outlines strategies to mitigate them, ensuring minimal disruptions and successful project completion.

2.4.1 Technical Risks

Potential challenges during the development and implementation of the RapidBill Fusion system:

- **Integration Challenges:** Delays or issues may arise if RapidBill Fusion is difficult to integrate with existing retail systems. Mitigation: Early collaboration with IT staff and thorough testing at integration points.
- **Security Vulnerabilities:** Possible exposure of sensitive transaction data. Mitigation: Implement strong encryption, conduct regular security assessments, and follow best practices for data protection.

2.4.2 Project Management Risks

Risks related to project management, planning, and execution:

- **Schedule Delays:** Delays in key milestones can impact the overall project timeline. Mitigation: Employ adaptive project management techniques and conduct regular progress reviews.
- **Budget Overruns:** Unforeseen costs could lead to exceeding the budget. Mitigation: Detailed cost planning, frequent budget reviews, and maintaining a contingency reserve.

- **Resource Availability:** Key team members may become unavailable during critical project phases. Mitigation: Provide backup resources and cross-train team members to ensure continuity.

2.4.3 Operational Risks

Operational risks associated with the deployment and use of RapidBill Fusion:

- **User Adoption:** Retail staff and customers may be reluctant to adopt the new system. Mitigation: Offer comprehensive training, design a user-friendly interface, and provide ongoing support.

- **System Downtime:** Unexpected outages could disrupt retail operations. Mitigation: Implement robust backup and recovery processes and monitor uptime consistently.

- **Regulatory Compliance:** Non-compliance with data protection laws could lead to legal issues. Mitigation: Regular consultation with legal and compliance experts to ensure adherence to relevant regulations.

METHODOLOGY ADOPTED

3.1 Investigative Techniques

Introduction

To ensure the successful development and implementation of the RapidBill Fusion project, various investigative techniques were employed. These techniques were crucial for understanding the current retail checkout processes, identifying user needs, and designing an effective AI-powered autonomous shopping and checkout system. The chosen methods provided comprehensive insights into the existing retail environment and technological requirements, ensuring the solution aligns with the project's goals.

Primary Research: Interviews and Surveys

Interviews: Conducted with retail managers, staff, and customers to gain detailed qualitative insights into existing checkout challenges, user preferences, and technological needs. Key topics included current checkout procedures, pain points related to waiting times, and the potential benefits of automation.

Surveys: Administered to a broader audience to gather quantitative data on user experiences, attitudes towards technology in retail, and preferences for self-checkout features. The survey results provided valuable feedback on customer expectations and readiness for adopting automated checkout solutions.

Justification: Interviews and surveys were selected for their ability to provide both qualitative and quantitative data, offering a well-rounded understanding of the current retail operations and user requirements. These techniques are effective for identifying gaps and opportunities for improvement, which are critical for developing the RapidBill Fusion system.

Secondary Research: Literature Review and Case Studies

Literature Review: Reviewed academic papers, industry reports, and whitepapers on retail automation, AI in checkout systems, and computer vision technologies. This research provided insights into best practices, emerging trends, and successful implementations of similar systems.

Case Studies: Analysed case studies of retail environments that have adopted advanced checkout technologies to identify challenges, solutions, and strategies for effective implementation.

Justification: The literature review and case studies offered valuable context and validation for the proposed system by highlighting successful approaches and potential pitfalls. These secondary research methods ensured that the RapidBill Fusion project would be informed by industry standards and real-world examples.

Observation and Process Mapping

Observation: Conducted in retail settings to observe current checkout processes, including customer interactions, manual billing practices, and common issues faced at checkout points.

Process Mapping: Created visual maps of existing checkout workflows to identify inefficiencies, bottlenecks, and areas for improvement.

Justification: Observation and process mapping provided real-time insights into the actual retail operations and highlighted specific challenges that might not be apparent through interviews or surveys. This information was essential for designing a system that effectively addresses current checkout inefficiencies.

3.2 Proposed Solution

Overview

The RapidBill Fusion project aims to transform the retail checkout experience through its advanced AI-powered autonomous shopping and checkout system. The proposed solution integrates three key components: Automated Item Recognition, Seamless Checkout, and User Interface. These elements work together to enhance operational efficiency, minimize wait times, and streamline the shopping process.

Automated Item Recognition

The Automated Item Recognition module is the core of the RapidBill Fusion system. It uses computer vision and AI to identify products in real-time as they are placed in the shopping cart. This module ensures accurate billing by automatically tracking items without manual scanning.

Key Features:

- **Real-Time Recognition:** The system identifies and catalogs items instantly, reducing the need for manual input and minimizing errors.
- **Integration with Pricing Database:** Links to a pre-uploaded price list to ensure accurate billing and up-to-date pricing.
- **Enhanced Accuracy**: Uses advanced algorithms to recognize items despite variations in packaging and labelling.

Seamless Checkout

The Seamless Checkout module streamlines the payment process by integrating with the Automated Item Recognition system. It allows customers to complete transactions effortlessly through a web or mobile platform.

Key Features:

User-Friendly Payment Options: Offers multiple payment methods for a smooth transaction experience.

- **Instant Billing:** Generates detailed bills in real-time, allowing customers to view and pay for their purchases quickly.
- **Queue Reduction:** Minimizes the need for traditional checkout lines, enhancing convenience and reducing wait times.

User Interface

The User Interface module is designed to provide a seamless interaction experience for both customers and retail staff. It includes web platforms for managing purchases and payments.

Key Features:

- **Intuitive Design:** Features a simple and easy-to-navigate interface that enhances user experience and reduces learning time.
- **Real-Time Updates**: Provides instant updates on billing and transaction status to keep customers informed throughout the checkout process.
- **Accessibility:** Designed to be accessible from various devices, ensuring that users can interact with the system from anywhere.

Additional Functionalities

The RapidBill Fusion system also includes several additional features to improve overall functionality:

- **Scalability:** Designed to accommodate varying transaction volumes and integrate with existing retail setups.

Benefits of the Proposed Solution

- **Scalability and Flexibility:** Adaptable to different retail environments and capable of handling increased transaction volumes as needed.
- **Operational Optimization:** Provides valuable data insights for retailers to improve operational efficiency and customer service.

3.3 Work Breakdown Structure

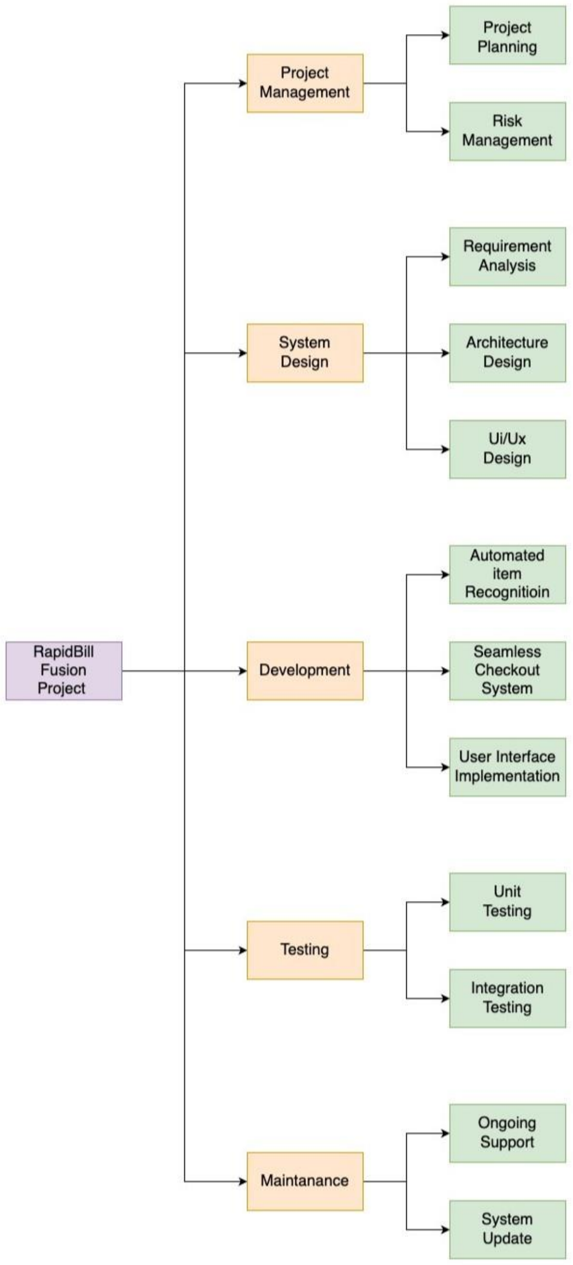


Figure 3.3 Work Breakdown Structure

3.4 Tools and Technology

- Tools used in this project:
    1. ESP32 Cam for capturing real-time images of products.
    2. Raspberry Pi 3B for running machine learning models and processing data.
    3. Weight Sensor for measuring the weight of scanned items.
    4. LED Light for enhancing image quality in low-light conditions.
    5. Breadboard and Jumper Wires for hardware circuitry connections.
    6. Arduino IDE for programming and configuring the ESP32 Cam.
    7. Python for developing and integrating object detection algorithms.

- Technologies used in our project:
    1. HTML, CSS, JavaScript, and React.js for building the user interface.
    2. Node.js and Express.js for backend development.
    3. Firebase for real-time data storage and synchronization.
    4. WhatsApp for sending generated e-bills to users.
    5. HTTP Protocol for communication between hardware devices and the backend.
    6. FOMO Algorithm for lightweight object detection on edge devices.

4.1 System Architecture

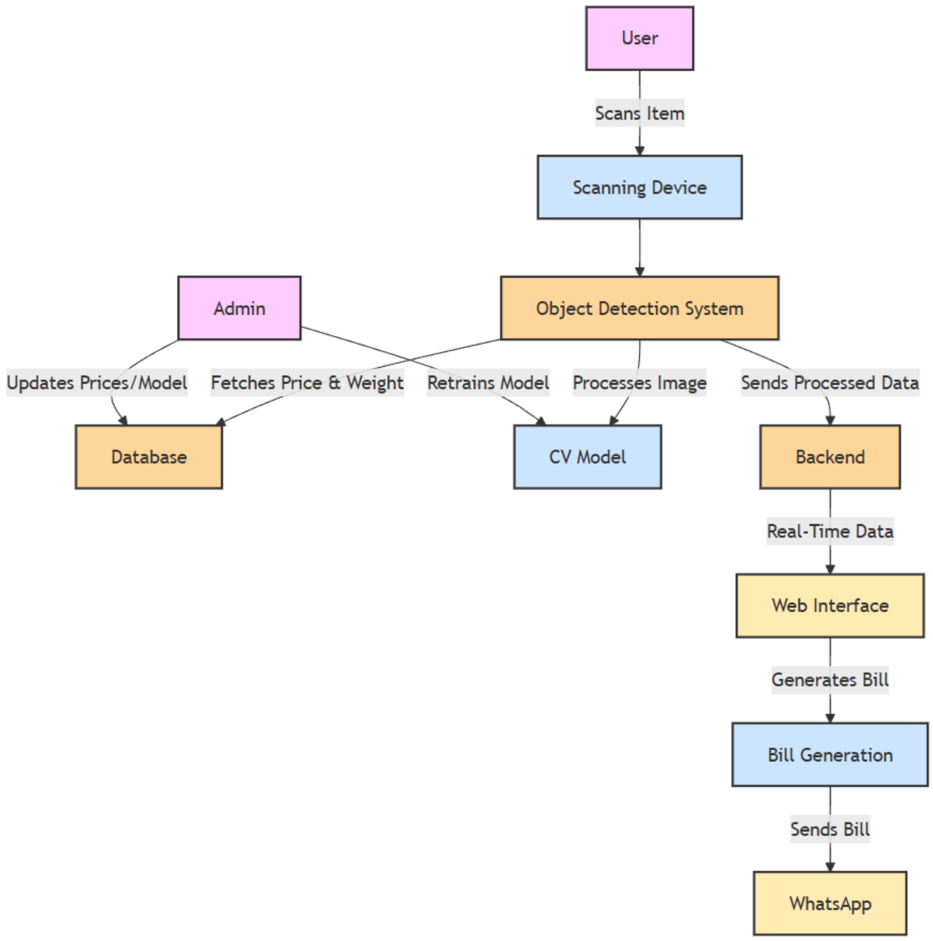


Figure 4.1.1 Architecture Block Diagram

4.2 Design Level Diagrams

4.2.1 Entity Relationship Diagrams

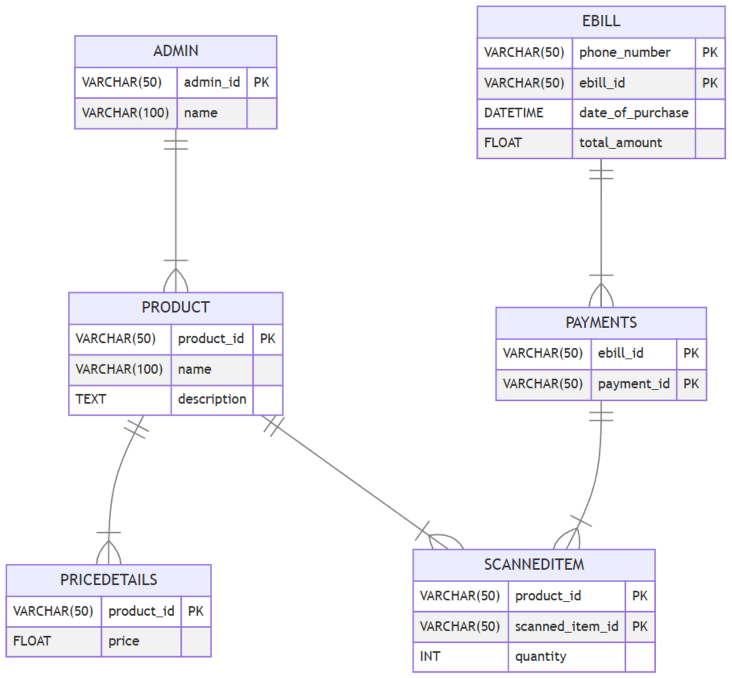


Figure 4.2.1 ER Diagram

4.2.2 Data Flow Diagrams

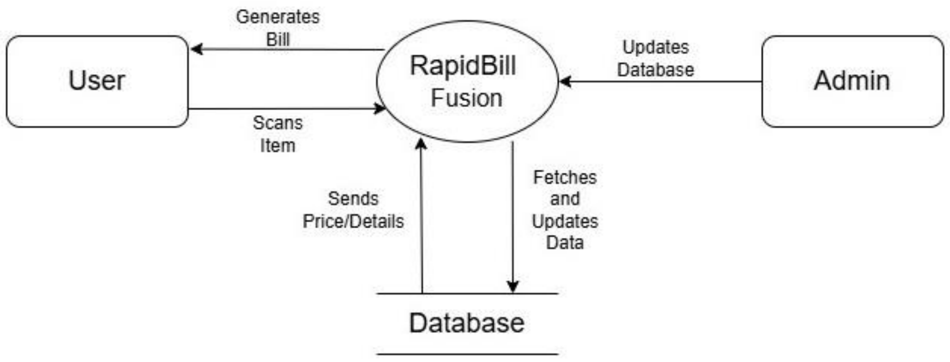


Figure 4.2.2.1 Level 0 Data Flow Diagram

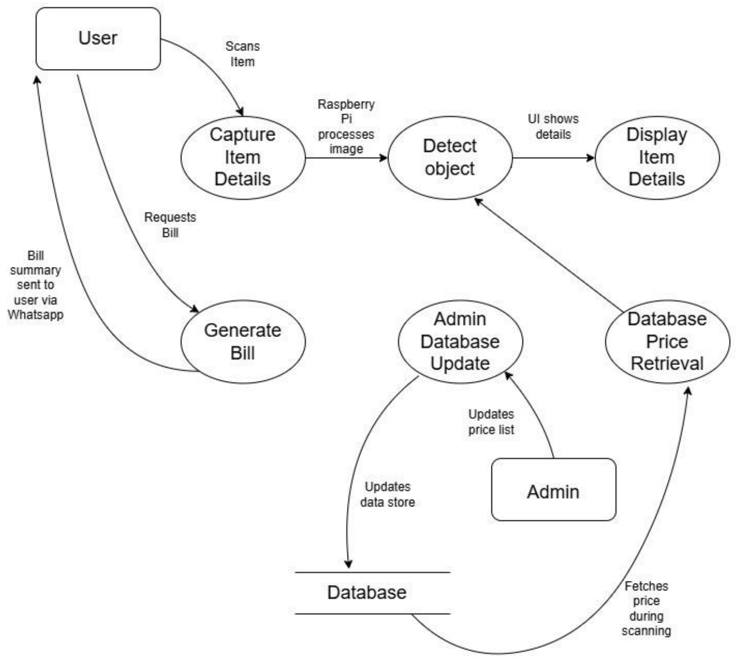


Figure 4.2.2.2 Level 1 Data Flow Diagram

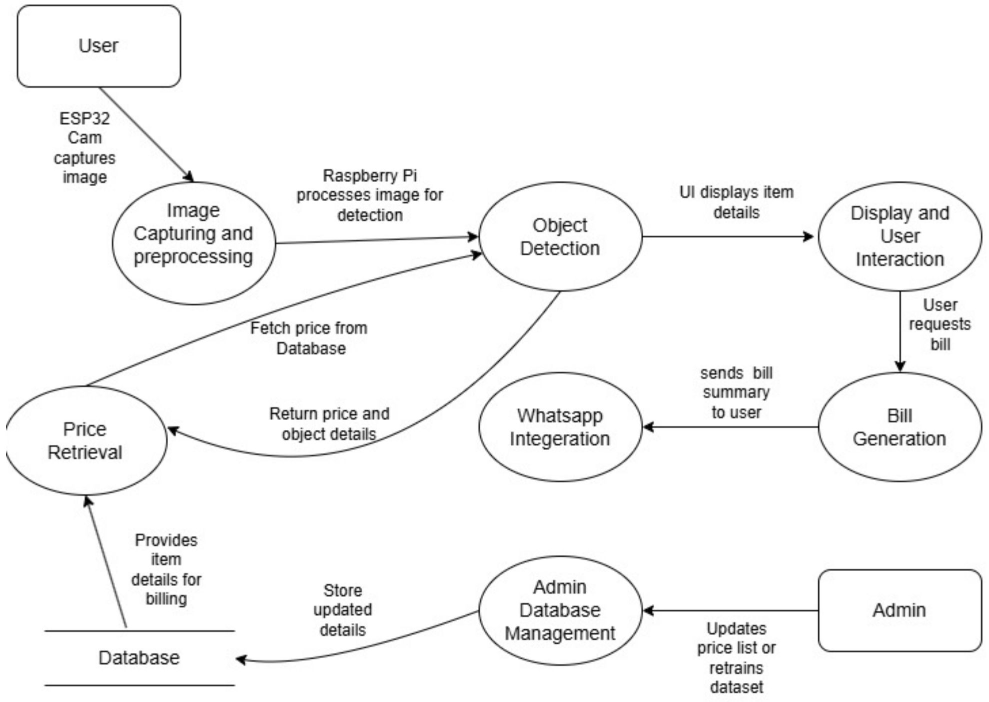

User
ESP32 Cam captures image
Image Capturing and preprocessing
Raspberry Pi processes image for detection
Object Detection
UI displays item details
Display and User Interaction
User requests bill
Fetch price from Database
Return price and object details
Price Retrieval
sends bill summary to user
Whatsapp Integeration
Bill Generation
Provides item details for billing
Store updated details
Admin Database Management
Admin
Updates price list or retrains dataset
Database

Figure 4.2.2.3 Level 2 Data Flow Diagram

4.2.3 Class Diagrams

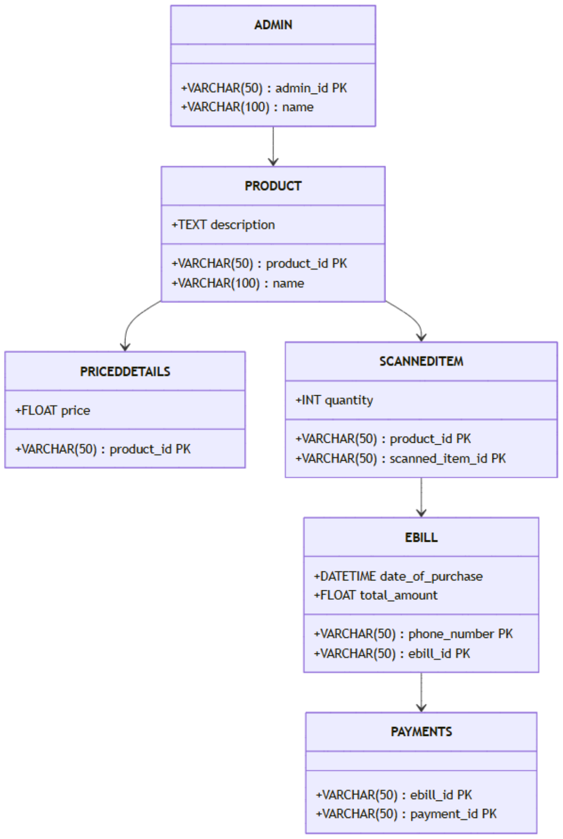

ADMIN
+VARCHAR(50) : admin_id PK
+VARCHAR(100) : name
PRODUCT
+TEXT description
+VARCHAR(50) : product_id PK
+VARCHAR(100) : name
PRICEDDETAILS
+FLOAT price
+VARCHAR(50) : product_id PK
SCANNEDITEM
+INT quantity
+VARCHAR(50) : product_id PK
+VARCHAR(50) : scanned_item_id PK
EBILL
+DATETIME date_of_purchase
+FLOAT total_amount
+VARCHAR(50) : phone_number PK
+VARCHAR(50) : ebill_id PK
PAYMENTS
+VARCHAR(50) : ebill_id PK
+VARCHAR(50) : payment_id PK

Figure 4.2.3 Class Diagram

4.3 User Interface Diagrams

4.3.1 Use Case Diagrams

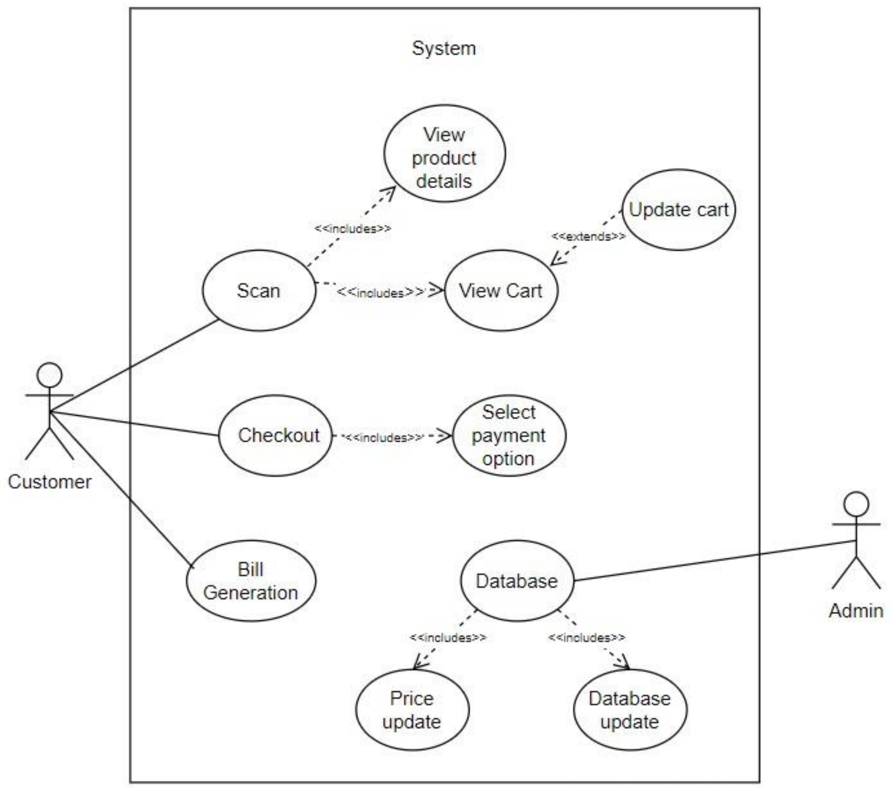


Figure 4.3.1.1 Use Case Diagram – Phase 1

4.3.2 Activity/Swimlane Diagram

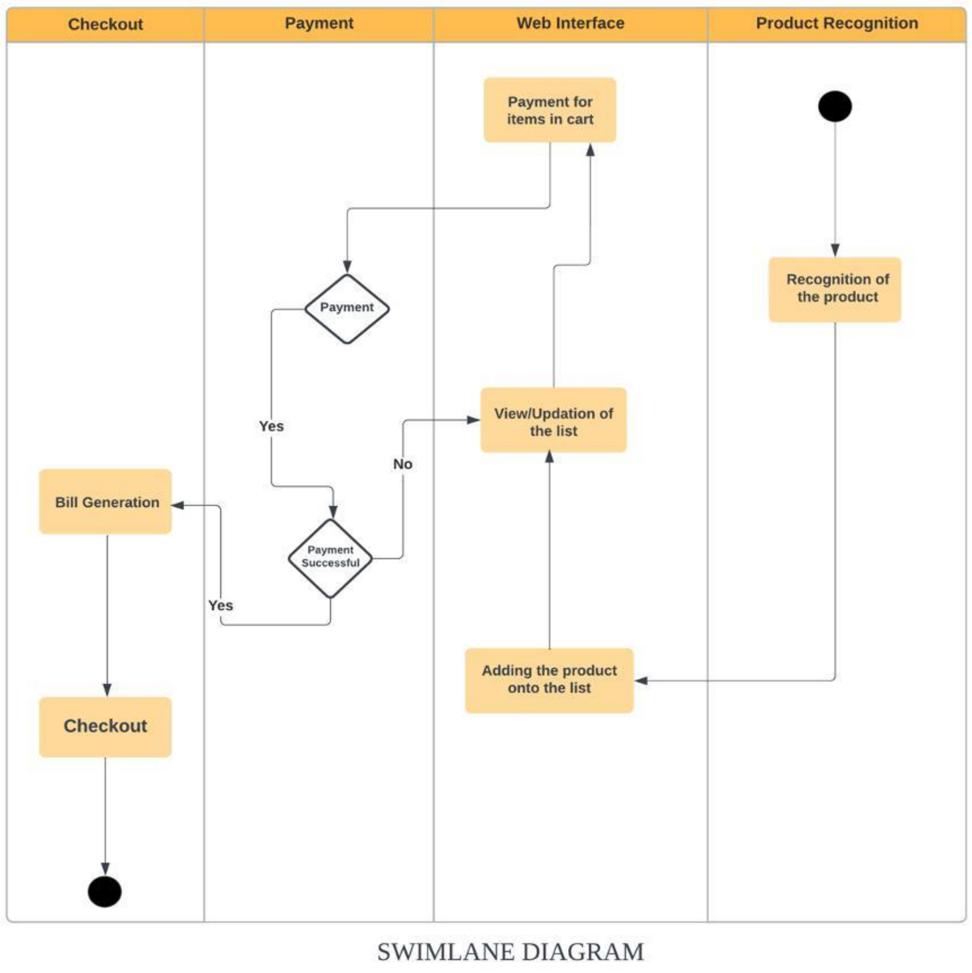
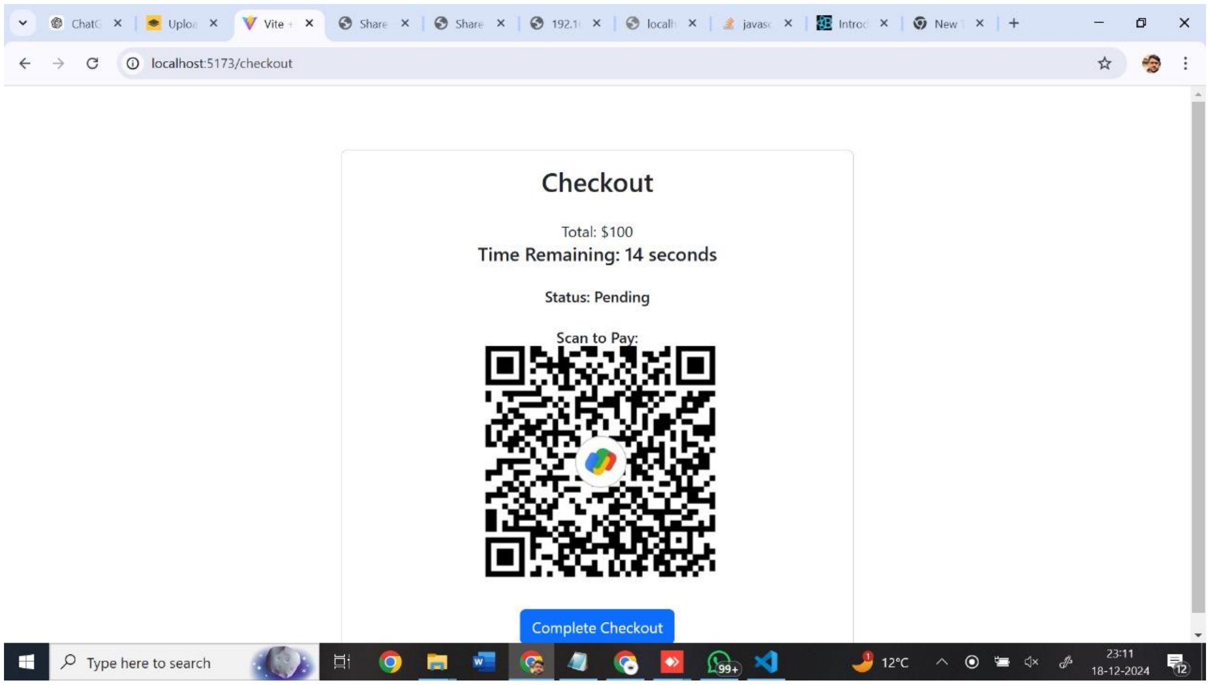


Figure 4.3.2 Swimlane Diagram

4.4 Snapshots of Working Prototype

Website:

Main screen –

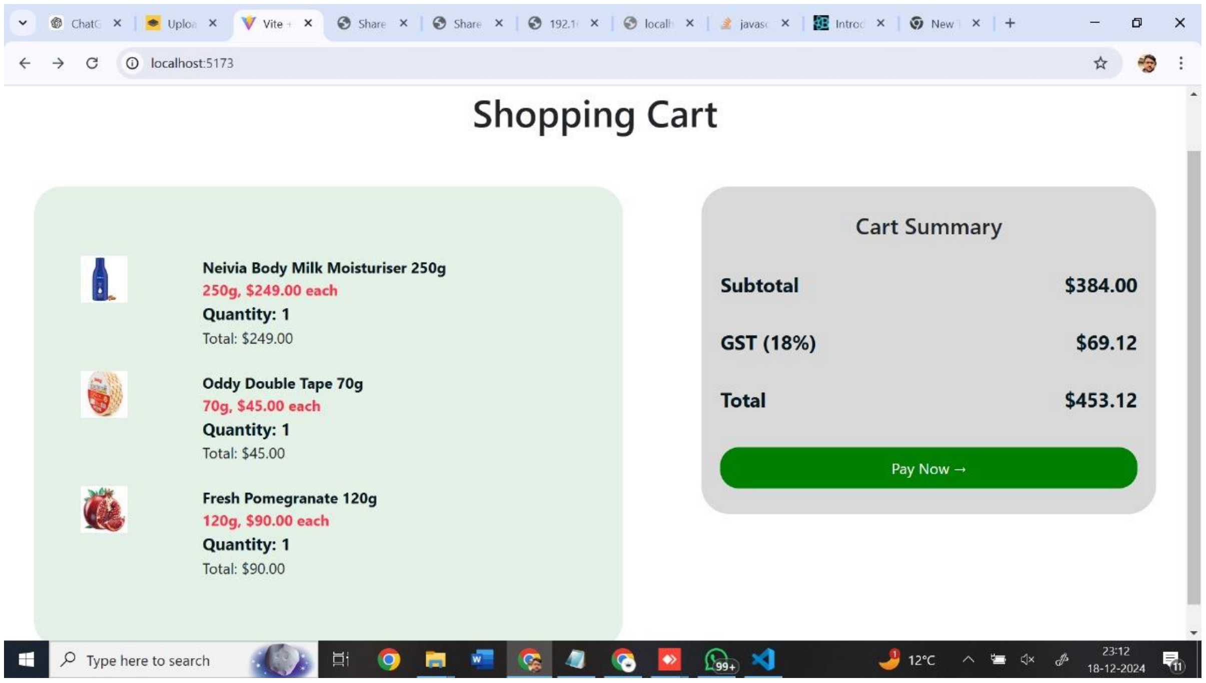


Checkout

Total: $100
Time Remaining: 14 seconds

Status: Pending

Scan to Pay:



Shopping Cart

Neivia Body Milk Moisturiser 250g
250g, $249.00 each
Quantity: 1
Total: $249.00

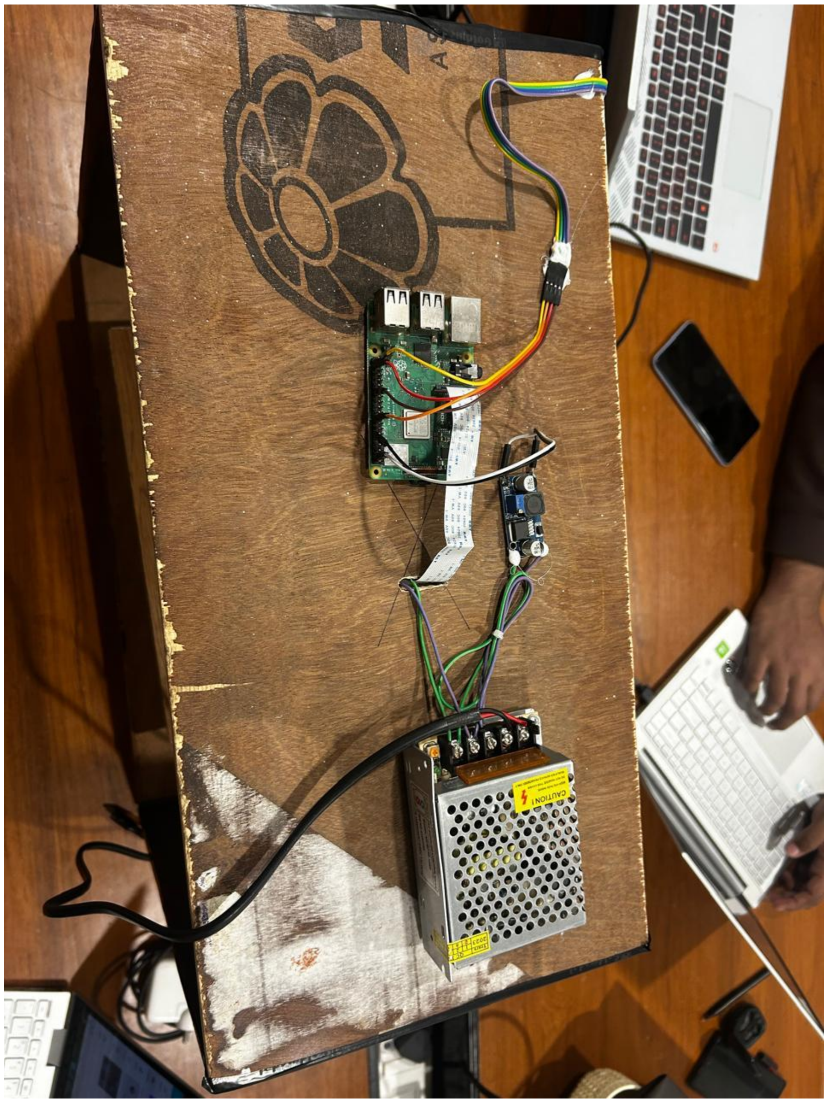
Oddy Double Tape 70g
70g, $45.00 each
Quantity: 1
Total: $45.00

Fresh Pomegranate 120g
120g, $90.00 each
Quantity: 1
Total: $90.00

Cart Summary

Subtotal	**$384.00**
GST (18%)	**$69.12**
Total	**$453.12**

IoT Module:



IMPLEMENTATIONS AND EXPERIMENTAL RESULTS

5.1 Experimental Setup

Phase - I: Object Detection and Billing System Setup

In this phase, the ESP32 Cam is placed on a stationary platform to simulate a retail environment where users scan products. A dedicated Wi-Fi network connects the ESP32 Cam to the Raspberry Pi 3B, which processes the captured images in real-time. The machine learning model, deployed using Edge Impulse, identifies the product and retrieves its details.

To capture the weight of the scanned item, a **Weight Sensor** is integrated into the platform. This enables simultaneous detection of both the product and its weight. A smartphone or tablet running the **React.js-based application** is used for user interaction, displaying product details such as name, price, and weight in real time.

Once the product is scanned and verified, a bill is sent to the user's registered phone number. This phase focuses on real-time product detection, user interaction, and automated billing.

Phase - II: Edge Processing with System Calibration

In the second phase, the **Raspberry Pi 3B** is placed on an elevated platform to simulate an edge-processing environment. The **ESP32 Cam** captures images of products moving on a **conveyor belt**, which represents a dynamic scanning setup. A **Weight Sensor** is placed on the conveyor to record the weight of each product in real time.

Finally, system performance is measured in terms of detection accuracy, billing latency, and real-time synchronization. Products are displayed on the connected application with details such as **name, price, and weight**, enabling performance validation and system calibration.

.

5.2 Experimental Analysis

The experimental analysis evaluates the performance, accuracy, and efficiency of the RapidBill Fusion system. The object detection accuracy of the machine learning model is tested under varying conditions, including low lighting, glare, and overlapping items, with detection precision and error rates recorded.

System latency is analysed for the end-to-end process, including image capture by the **ESP32 Cam**, processing on the **Raspberry Pi 3B**, and bill generation via the **WhatsApp API**. The **weight sensor** is tested for accuracy across products of different sizes, with calibration performed to minimize errors.

The **user experience** is assessed based on the ease of scanning, real-time updates on the user interface, and reliability under continuous use. Feedback is gathered to improve usability and responsiveness. **Scalability** is analysed under multiple product scans and concurrent user loads to observe system performance on edge devices.

Lastly, the **energy efficiency** of the ESP32 Cam and Raspberry Pi is evaluated to optimize power usage during operation. The analysis highlights key improvements in detection accuracy, reduced latency, and scalability for real-world deployment.

5.2.1 Data

The dataset used for RapidBill Fusion consists of 500 images representing 5 unique products. Each product is captured from multiple angles and under varying conditions to ensure the model can generalize well during real-world usage. The images are collected in diverse lighting conditions, including natural and artificial light, to simulate real retail environments.

Each product in the dataset has:

1. **Multiple Angles**: Front, side, and top views to improve object recognition accuracy.

2. **Different Backgrounds**: Images include clean, cluttered, and textured backgrounds to enhance the model's ability to differentiate products.

3. **Varying Distances**: Products are captured at close and medium ranges to account for distance variations during scanning.

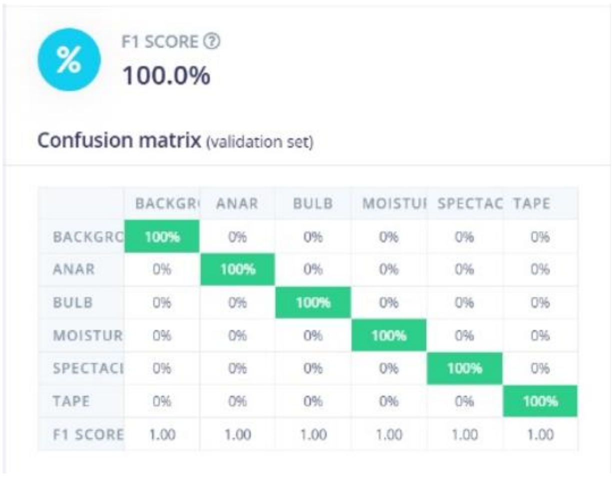
4. **Labeling**: Each image is annotated with the product name (label) to train the object detection model.

5.2.2 Performance Parameters

The ML models were tested for Confidence Score and F1 score parameter and our main focus was on the accuracy of the models.

Labels	Confidence Score
Anar	0.92
Moisturizer	0.03
Tape	0.02
Bulb	0.02
Spectacles	0.01

Table 5.2.2.1 Confidence Score


F1 SCORE ⓘ
100.0%

Confusion matrix (validation set)

	BACKGR	ANAR	BULB	MOISTUI	SPECTAC	TAPE
BACKGRC	100%	0%	0%	0%	0%	0%
ANAR	0%	100%	0%	0%	0%	0%
BULB	0%	0%	100%	0%	0%	0%
MOISTUR	0%	0%	0%	100%	0%	0%
SPECTACI	0%	0%	0%	0%	100%	0%
TAPE	0%	0%	0%	0%	0%	100%
F1 SCORE	1.00	1.00	1.00	1.00	1.00	1.00

5.3 Working of the Project

5.3.1 Procedural Workflow

Step 1: Hardware Setup

1. **ESP32 Cam Configuration:**

    - Install and configure the ESP32 Cam module.
    - Enable real-time image feed capturing.
    - Set up communication to transmit image data to the Raspberry Pi.

2. **Raspberry Pi Setup:**
    - Install necessary dependencies (Python, ML libraries, Edge Impulse).
    - Deploy and test ML models for object detection.
    - Establish connectivity between ESP32 Cam and Raspberry Pi.

Step 2: Object Detection on Raspberry Pi

1. **Image Processing:**
    - Receive real-time image feed from the ESP32 Cam.
    - Preprocess image data for ML model input.
2. **Run Object Detection:**
    - Identify objects in the image feed with:
        - Object name.
        - Detection accuracy score.
    - Prepare detection results for further processing.

Step 3: Data Preparation for Transmission

- Format detection results into structured data packets containing:
    - Object name.
    - Price (from pre-uploaded price list).

   - Weight (if applicable).
   - Detection accuracy.

Step 4: Data Transfer via HTTP Protocol

1. **Setup Communication:**
   - Configure the Raspberry Pi to use HTTP for communication.
   - Establish a real-time connection between Raspberry Pi and the User Interface (UI).
2. **Transmit Data:**
   - Send detection results to the UI in real time.
   - Ensure minimal latency during data transmission.

Step 5: Real-Time Display on User Interface (UI)

1. **UI Design:**
   - Create a user-friendly interface (web or app-based).
   - Display details such as:
     - Object name.
     - Price.
     - Weight.
     - Detection accuracy.
2. **User Interaction:**
   - Include a button for bill generation.

- Enable seamless interaction between users and the system.

Step 6: Bill Generation and WhatsApp Integration

1. **Generate Bill:**
    - On user request, compile detected object details into a summarized bill.
    - Calculate the total cost.
2. **WhatsApp API Integration:**
    - Trigger bill sharing via WhatsApp using APIs (e.g., Twilio).
    - Send the generated bill to the user's registered WhatsApp number.

Step 7: System Refinement and Calibration

1. **Validation:**
    - Conduct weight surveys and validate detection accuracy.
    - Refine and adjust system thresholds.
2. **Data Update:**
    - Synchronize updated prices and detection parameters with Firebase or other storage systems.
    - Periodically calibrate the system for optimal performance.

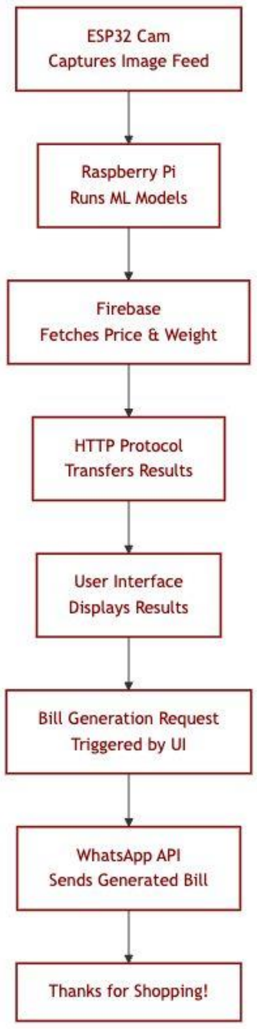


Table 5.3.1 Procedural Workflow

5.3.2 Algorithmic Approaches Used

FOMO (Fast Object Detection Model):

FOMO is a lightweight object detection algorithm optimized for edge devices, making it ideal for IoT applications like the RapidBill Fusion project. Unlike traditional CNN-based models, FOMO focuses on detecting multiple objects in real-time with high efficiency while requiring minimal computational resources. It uses grid-based feature extraction to predict the presence of objects and their locations, ensuring fast and accurate detections even on constrained hardware like Raspberry Pi.

Key Formula:

FOMO uses a simplified version of bounding box regression and confidence scoring

$$L = L_{\text{cls}} + \lambda L_{\text{loc}}$$

Transfer Learning with FOMO:

Transfer learning is used to fine-tune the FOMO model for the specific objects in the retail environment. A pre-trained FOMO model is adapted to the dataset of retail items, allowing the system to detect objects with high accuracy. This approach significantly reduces training time and improves performance, particularly when limited labeled data is available.

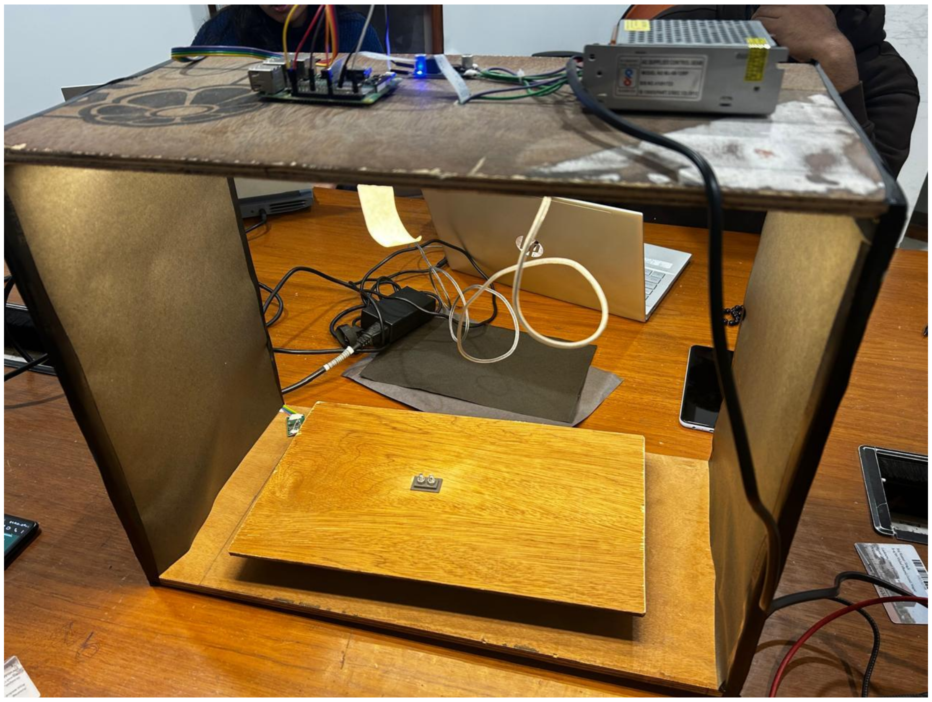
5.3.3 Project Deployment

The deployment of the **RapidBill Fusion** project involves setting up the ESP32 Cam and Raspberry Pi for real-time image capturing and processing. The Raspberry Pi runs the lightweight FOMO object detection model, optimized for edge devices, to identify items and transmit results via HTTP to a user-friendly web interface. The UI displays object details, facilitates bill generation, and integrates with WhatsApp APIs for seamless bill delivery. The system is designed for scalability and can be easily updated to improve accuracy and performance through periodic model retraining and calibration.

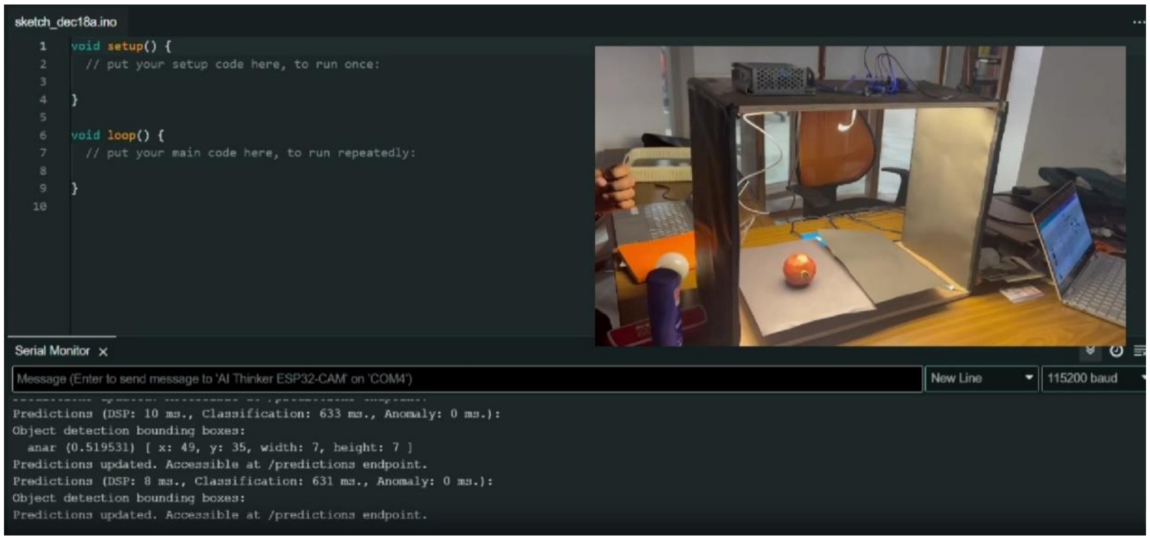
5.3.4 System Screenshots

The below images show the working prototypes of our project:

IoT Enable Checkout System



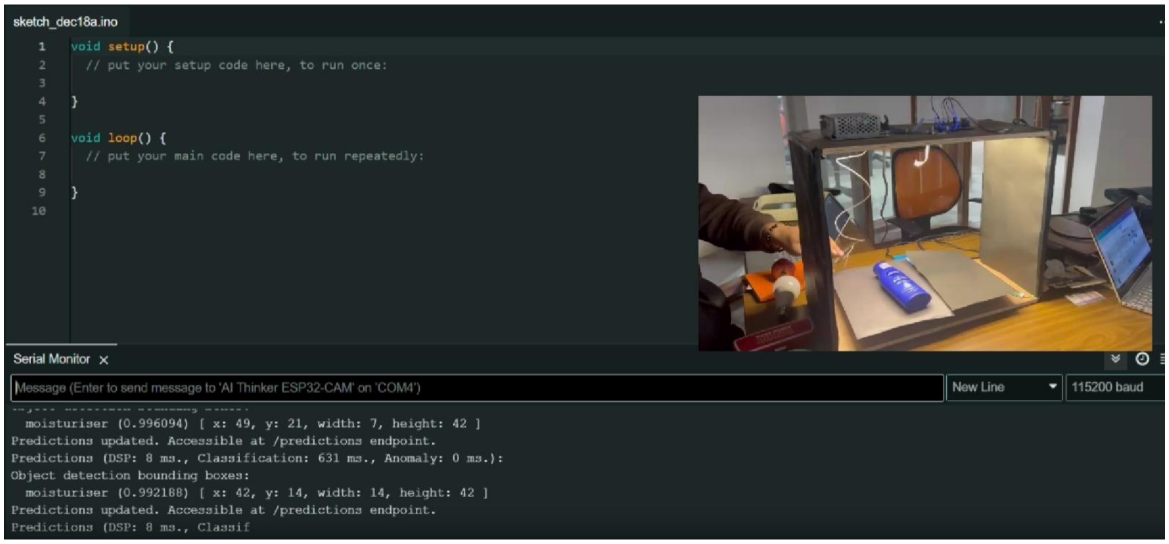
Web Interface



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Message (Enter to send message to 'AI Thinker ESP32-CAM' on 'COM4')

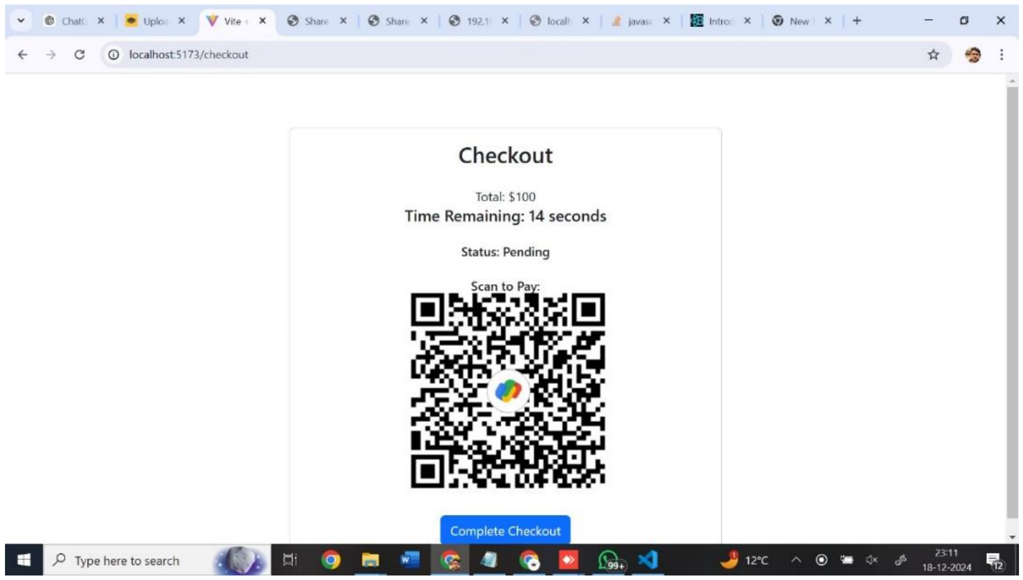
```
Predictions (DSP: 10 ms., Classification: 633 ms., Anomaly: 0 ms.):
Object detection bounding boxes:
  anar (0.519531) [ x: 49, y: 35, width: 7, height: 7 ]
Predictions updated. Accessible at /predictions endpoint.
Predictions (DSP: 8 ms., Classification: 631 ms., Anomaly: 0 ms.):
Object detection bounding boxes:
Predictions updated. Accessible at /predictions endpoint.
```



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Message (Enter to send message to 'AI Thinker ESP32-CAM' on 'COM4')

```
  moisturiser (0.996094) [ x: 49, y: 21, width: 7, height: 42 ]
Predictions updated. Accessible at /predictions endpoint.
Predictions (DSP: 8 ms., Classification: 631 ms., Anomaly: 0 ms.):
Object detection bounding boxes:
  moisturiser (0.992188) [ x: 42, y: 14, width: 14, height: 42 ]
Predictions updated. Accessible at /predictions endpoint.
Predictions (DSP: 8 ms., Classif
```


Checkout

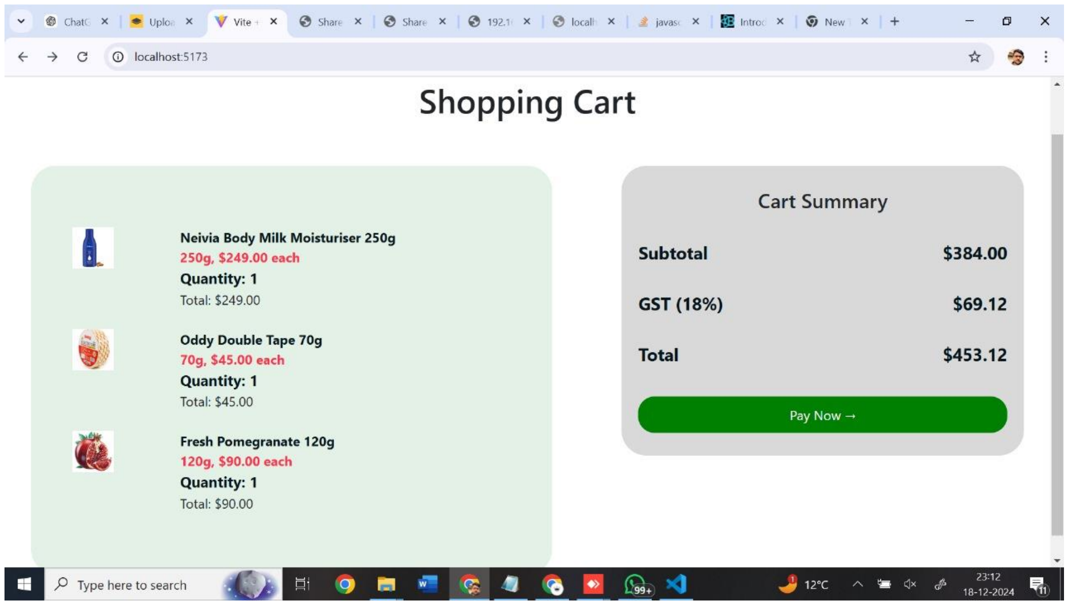
Total: $100

Time Remaining: 14 seconds

Status: Pending

Scan to Pay:

Complete Checkout


Shopping Cart

Neivia Body Milk Moisturiser 250g
250g, $249.00 each
Quantity: 1
Total: $249.00

Oddy Double Tape 70g
70g, $45.00 each
Quantity: 1
Total: $45.00

Fresh Pomegranate 120g
120g, $90.00 each
Quantity: 1
Total: $90.00

Cart Summary

Subtotal	$384.00
GST (18%)	$69.12
Total	$453.12

Pay Now →

5.4 Testing Process

Testing the RapidBill Fusion system involves verifying each component for functionality and accuracy. Initial testing begins with the hardware setup, ensuring the ESP32 Cam captures and transmits clear image feeds to the Raspberry Pi. The object detection model is then tested for precision, ensuring accurate identification and classification of items. HTTP communication is validated to ensure seamless real-time data transfer to the user interface (UI). The UI is tested for responsiveness, proper display of object details, and smooth bill generation functionality. Finally, the integration with WhatsApp API is verified by sending generated bills to test numbers, ensuring reliable and timely delivery. System performance is monitored under various conditions to refine and optimize for accuracy, speed, and user experience.

5.4.1 Test Plan

Component-Level Testing:

- Verify hardware functionality (ESP32 Cam, Raspberry Pi).
- Test the object detection model for accuracy and reliability with various item types.

Integration Testing:

- Ensure seamless data transfer via HTTP protocol between Raspberry Pi and the UI.
- Validate the UI's responsiveness, real-time updates, and accurate bill generation.

End-to-End System Testing:

- Perform comprehensive testing of the entire workflow, including WhatsApp API integration for bill delivery.
- Simulate real-world usage scenarios to assess performance, latency, and user experience.

The overall success of the project depends on the smooth integration and coordination of the different components.

5.4.2 Features to be Tested

The below mentioned are the features to be tested:

1. Object Detection Accuracy: Verify correct object identification and classification.
2. Real-Time Processing: Ensure low-latency data processing and transmission.
3. UI Functionality: Test real-time display and user-friendly navigation.
4. Bill Generation: Validate accurate and detailed bill creation.
5. WhatsApp Integration: Confirm reliable bill delivery via WhatsApp.
6. Dataset Storage: Test optional image saving for model retraining.
7. Scalability: Evaluate system performance under increased load.

5.4.3 Test Strategy

The testing in the project would be the most critical part, since the aim of the project is related to the betterment of the waste management and public sanitation in the country thus it becomes a matter of high importance that the various components work well together. This would ensure a streamlined process of efficient object detection.

5.4.4 Test Techniques

5.4.4.1 Unit Testing

Unit testing entails creating test cases to ensure that the core programme logic is proper and that programme inputs create valid outputs. Verify all decision branches and internal code flow. This is a test of an individual software unit of the application that is run after completing the specific unit prior to the integration. This is an invasive structural check that relies on knowledge of its structure. Unit tests perform basic component-level testing to test specific business processes, applications, and system configurations. Unit tests use well-defined inputs and expected results to ensure that all paths in a business process adhere precisely to a documented specification. Unit testing is usually performed as part of the combined code and unit testing phase of the software lifecycle, but it is not uncommon for coding and unit testing to be performed in two separate phases.

5.4.4.2 Integration Testing

Integration tests are designed to test built-in software components to see if they actually run as programs. Tests are event driven and more concerned with the underlying results of a screen or field. Integration testing shows that the combination of components is correct and consistent while the components are satisfied individually, as indicated by the successful unit tests. Integration testing is specifically aimed at uncovering problems arising from the combination of components.

Software integration testing is incremental integration testing of two or more integrated software components on a single platform and produces errors caused by interface errors. The task of integration testing is to validate a component or software application. The components within a software system or sophisticated enterprise-level software application interact flawlessly.

5.4.4.3 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing relies on process descriptions and flows, with an emphasis on pre-driven process connections and integration points. User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

5.4.4.4 Functional Testing

Functional testing provides systematic evidence that the functionality tested is available according to business and technical requirements, system documentation, and user guides. Functional testing focuses on the following elements:

- **Valid Inputs:** Must accept identified classes of valid inputs.
- **Invalid Input:** The class of invalid input identified should be rejected.
- **Function:** Required to perform the specified function.
- **Output:** Must use application's identified output class.
- **System Procedure:** Must call connection system or procedure.

Functional test organization and preparation focuses on requirements, major functions, or special test cases. Additionally, systematic coverage should be considered in terms of identifying business process flows, predefined data field processes, and a set of processes for testing. Before completing a functional test, additional tests are identified and valid values for the current test are determined.

It can be done in two approaches:

1. Black box testing
2. White box testing

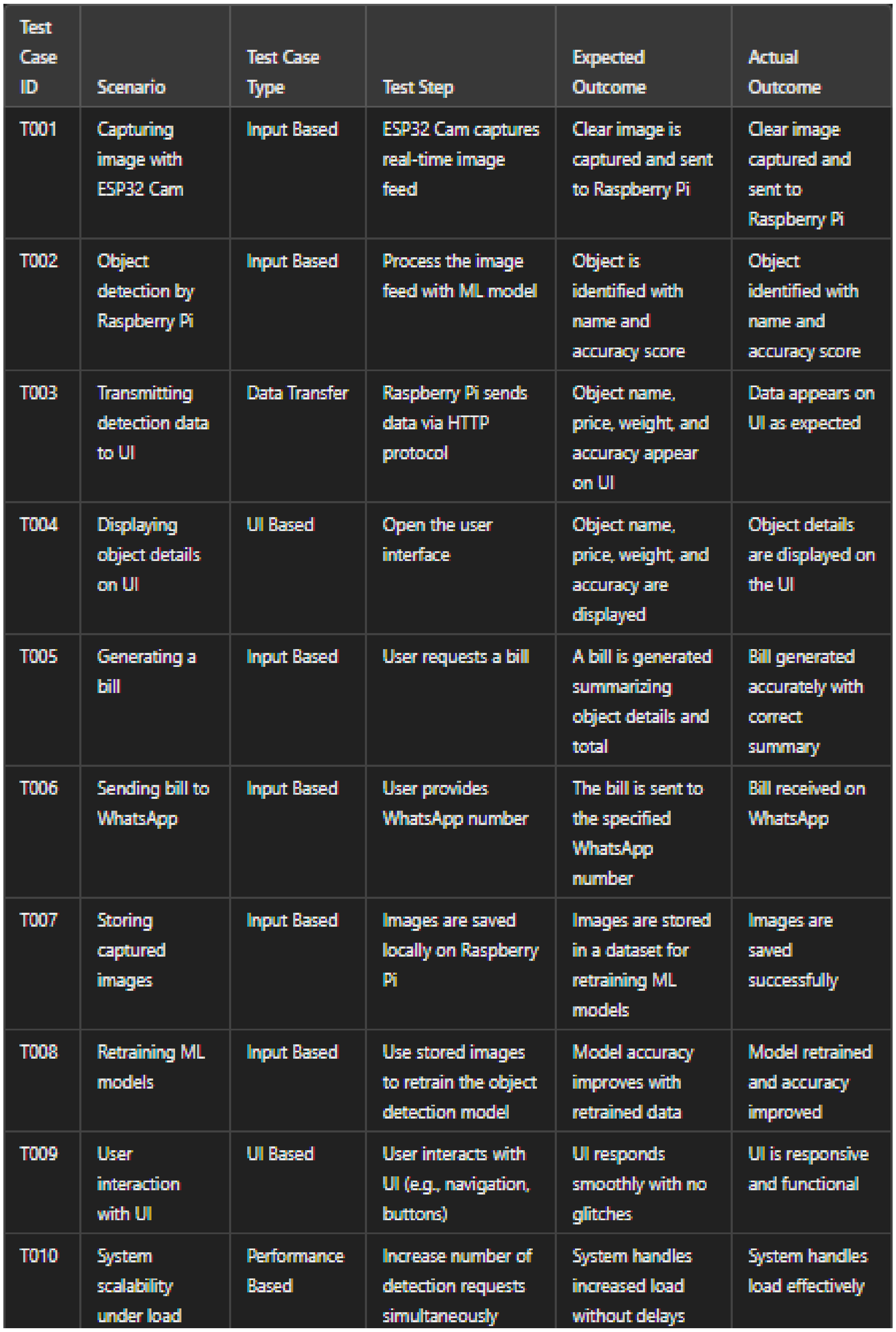
5.4.4.4.1 Black Box Testing

This method is used when knowledge of the specific functionality for which the product is designed is known. Black boxes are used to represent system hoses in the work that are not available for inspection. In the black box, the test object is eaten as "black". The logic is that we don't know what goes in and what goes out, or the inputs and outputs. Black box testing involves trying different inputs and examining the resulting outputs. Black-box testing can also be used for scenario-based testing. This test verifies that valid input is retrieved, and results are returned to the user. A fictional box test that hides the inner workings. For our project, valid image information is the resulting output of a well-structured image that we need to receive.

5.4.4.4.2 White Box Testing

White-box testing is used to test the program's implementation. The purpose of structural tests is not to test all inputs or outputs, but to test various programming and data structures used in the program. Structural testing is therefore aimed at achieving test cases that enforce the desired coverage of the various structures.

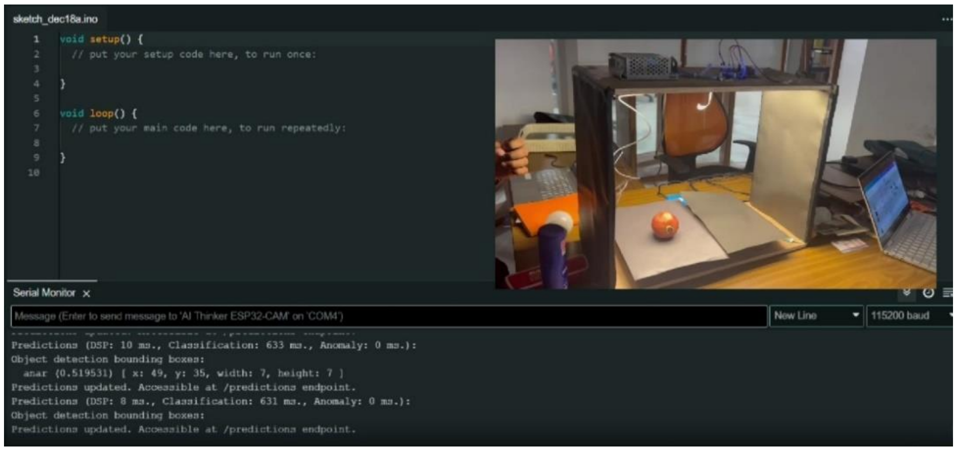
5.4.5 Test Cases



Test Case ID	Scenario	Test Case Type	Test Step	Expected Outcome	Actual Outcome
T001	Capturing image with ESP32 Cam	Input Based	ESP32 Cam captures real-time image feed	Clear image is captured and sent to Raspberry Pi	Clear image captured and sent to Raspberry Pi
T002	Object detection by Raspberry Pi	Input Based	Process the image feed with ML model	Object is identified with name and accuracy score	Object identified with name and accuracy score
T003	Transmitting detection data to UI	Data Transfer	Raspberry Pi sends data via HTTP protocol	Object name, price, weight, and accuracy appear on UI	Data appears on UI as expected
T004	Displaying object details on UI	UI Based	Open the user interface	Object name, price, weight, and accuracy are displayed	Object details are displayed on the UI
T005	Generating a bill	Input Based	User requests a bill	A bill is generated summarizing object details and total	Bill generated accurately with correct summary
T006	Sending bill to WhatsApp	Input Based	User provides WhatsApp number	The bill is sent to the specified WhatsApp number	Bill received on WhatsApp
T007	Storing captured images	Input Based	Images are saved locally on Raspberry Pi	Images are stored in a dataset for retraining ML models	Images are saved successfully
T008	Retraining ML models	Input Based	Use stored images to retrain the object detection model	Model accuracy improves with retrained data	Model retrained and accuracy improved
T009	User interaction with UI	UI Based	User interacts with UI (e.g., navigation, buttons)	UI responds smoothly with no glitches	UI is responsive and functional
T010	System scalability under load	Performance Based	Increase number of detection requests simultaneously	System handles increased load without delays	System handles load effectively

5.4.6 Test Results

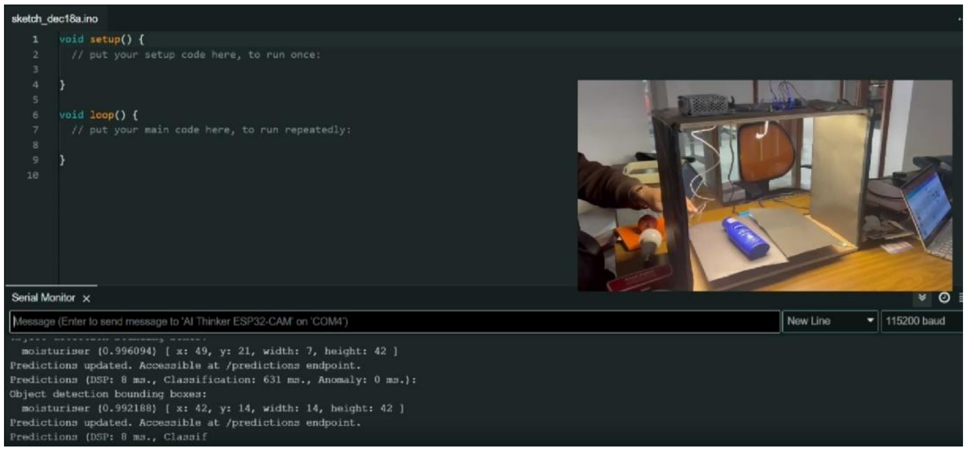
Following are the test results of the Model:



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

```
Message (Enter to send message to 'AI Thinker ESP32-CAM' on 'COM4')
```

```
Predictions (DSP: 10 ms., Classification: 633 ms., Anomaly: 0 ms.):
Object detection bounding boxes:
  anar (0.519531) [ x: 49, y: 35, width: 7, height: 7 ]
Predictions updated. Accessible at /predictions endpoint.
Predictions (DSP: 8 ms., Classification: 631 ms., Anomaly: 0 ms.):
Object detection bounding boxes:
Predictions updated. Accessible at /predictions endpoint.
```



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

```
Message (Enter to send message to 'AI Thinker ESP32-CAM' on 'COM4')
```

```
  moisturiser (0.996094) [ x: 49, y: 21, width: 7, height: 42 ]
Predictions updated. Accessible at /predictions endpoint.
Predictions (DSP: 8 ms., Classification: 631 ms., Anomaly: 0 ms.):
Object detection bounding boxes:
  moisturiser (0.992188) [ x: 42, y: 14, width: 14, height: 42 ]
Predictions updated. Accessible at /predictions endpoint.
Predictions (DSP: 8 ms., Classif
```

5.5 Results and Discussion

The web application in the **RapidBill Fusion** system provides users with a seamless and intuitive shopping experience. The application displays real-time details of detected objects, including the object name, price, weight, and detection accuracy, directly on the interface. Users can review the list of items as they are added and monitor the dynamic update of the total bill. The application also features a simple one-click option for generating a detailed bill, summarizing the detected items. Furthermore, the system integrates with WhatsApp to deliver the generated bill directly to the user's registered number for added convenience. The captured data is also stored for retraining and improving the ML model’s accuracy, ensuring continuous refinement and efficiency of the system.

5.6 Inferences Drawn

From the analysis of the results achieved and discussions, the following inferences can be drawn for the **RapidBill Fusion** project:

1. Real-time image capture and transmission from the ESP32 Cam to the Raspberry Pi were successful.
2. Object detection by the FOMO model on the Raspberry Pi, including identification and classification, was accurate and efficient.
3. The transmission of detected object details (name, price, weight, accuracy) to the user interface via HTTP was successful.
4. The user interface displayed object details in real time and allowed seamless interaction.
5. The bill generation process, summarizing object details and total cost, worked accurately.
6. Integration with WhatsApp for bill delivery was successfully implemented, ensuring timely delivery to the user's registered number.
7. The optional storage of images for retraining the ML model was effective, enabling improved accuracy in detection.
8. The overall system, including the hardware-software interface and user interactions, functioned as intended, providing a streamlined and autonomous shopping experience.

5.7 Validation of Objectives

1. To study the existing autonomous retail systems and identify their limitations was successfully achieved.
2. To design and develop an IoT-integrated system for real-time object detection and billing was successfully achieved.
3. To implement and deploy a machine learning model (FOMO) for accurate object detection and classification was successfully achieved.
4. To develop a user-friendly web application that displays real-time object details, enables seamless bill generation, and integrates with WhatsApp for bill delivery was successfully achieved.

CONCLUSIONS AND FUTURE DIRECTIONS

6.1 Conclusions

The RapidBill Fusion project successfully demonstrates the implementation of an AI-powered autonomous checkout system designed to improve operational efficiency and enhance the retail shopping experience. By leveraging advanced technologies such as computer vision, edge-based machine learning, and cloud-based communication, the system delivers a reliable, scalable, and efficient solution for modern retail environments.

The system is built around real-time object detection, where images captured by the ESP32 Cam are processed using the FOMO algorithm on a Raspberry Pi. The lightweight and resource-efficient FOMO model ensures minimal latency while maintaining high detection accuracy, making it ideal for edge devices. Product details such as price and weight are retrieved dynamically from the Firebase database, ensuring accurate and consistent billing information.

The integration of a WhatsApp API for bill generation and delivery streamlines the payment process, reducing the need for physical receipts and minimizing human interaction. This contributes to a faster, queue-free checkout experience for customers, enhancing convenience and satisfaction.

Key Achievements of the Project:

- Real-time Image Processing: Implementation of the FOMO algorithm on the Raspberry Pi enables real-time object detection at 20+ FPS with high accuracy.
- Accurate Billing: Integration with Firebase allows retrieval of product price and weight in real time, ensuring billing accuracy.
- Seamless Communication: Bill generation and automated delivery via the WhatsApp API provide a smooth and modernized customer experience.
- Efficiency and Convenience: The system significantly reduces checkout times, eliminates waiting queues, and minimizes human intervention.

In conclusion, RapidBill Fusion is a transformative solution for retail businesses, offering a scalable and efficient system to modernize the checkout process. By combining the power of edge AI, cloud computing, and communication technologies, this project sets the foundation for the future of autonomous shopping systems, enhancing both retailer operations and customer satisfaction.

6.2 Environmental, Economic and Societal Benefits

Environmental Benefits:

- Reduced dependency on printed receipts through digital bill generation, thereby saving paper and contributing to sustainability goals.
- Promotes energy-efficient operations with low-power hardware like Raspberry Pi and ESP32 Cam.

Economic Benefits:

- Reduces labor costs by minimizing the need for manual billing personnel.
- Enhances operational efficiency, leading to better resource utilization and higher profitability for retailers.
- Provides cost-effective edge solutions using lightweight AI algorithms.

Societal Benefits:

- Customer Convenience: Faster, queue-free checkout enhances shopping experiences.
- Health and Safety: Reduces physical interaction, particularly relevant in post-pandemic scenarios.
- Encourages the adoption of smart retail technologies, driving technological advancement and skill development.

6.3 Reflections

The project provided hands-on experience in developing a real-world AI-based solution.

Key Challenges:

- Hardware resource limitations on the Raspberry Pi (addressed using FOMO).
- Achieving real-time communication between devices and Firebase.

Key Learnings:

- Optimizing machine learning models for edge devices.
- Integration of hardware, cloud services, and communication APIs for a seamless workflow.
- Effective team collaboration and problem-solving to overcome technical hurdles.

The project reflects the potential of AI-powered edge computing in solving real-world problems and highlights its applications in the evolving retail industry.

6.4 Future Work Plan

1. Enhanced Object Detection:
   Improve accuracy for smaller and overlapping objects using advanced edge-based models or hybrid approaches.
2. Scalability:
   Deploy the system in large-scale retail environments and test performance under heavy loads.
3. Product Recommendation System:
   Integrate machine learning to provide product suggestions based on customer preferences.
4. Integration with Payment Gateways:
   Automate payments through seamless integration with digital payment platforms.
5. Advanced Hardware Integration:
   Explore more powerful edge devices (e.g., Jetson Nano) for improved processing power and performance.

PROJECT METRICS

7.1 Challenges Faced

1. **Hardware Resource Constraints:**
    - Issue: Raspberry Pi had limited CPU and RAM, which created performance bottlenecks.
    - Solution: Used the lightweight FOMO algorithm to ensure real-time object detection.
2. **Latency in Communication**:
    - Issue: Delays occurred while retrieving data from Firebase and updating the UI.
    - Solution: Optimized database queries and HTTP communication for minimal latency.
3. **Hardware Integration Issues:**
    - Issue: Synchronizing ESP32 Cam with Raspberry Pi for continuous image capture.
    - Solution: Debugged hardware connections and ensured proper power supply.
4. **Model Accuracy for Smaller Objects:**
    - Issue: Detection of small or overlapping objects posed challenges.
    - Solution: Retrained the model using an enhanced dataset to improve performance.

7.2 Relevant Subjects

Subject Code	Subject Name	Description
UML501	**Machine Learning**	**Approaches to automated knowledge acquisition that are both cost-effective and focus on the computational implications in burgeoning data-rich areas.**
UCS310	**Database Management System**	**The importance of information systems is emphasised. This course focuses on E-R diagrams, relational databases, and the principles of normalisation and denormalization.**
UCS503	**Software Engineering**	**Specify, abstract, verify, validate, plan, build, and manage big software projects, and learn about current software engineering techniques.**
UCS411	**Artificial Intelligence**	**Application, strengths, and shortcomings of fundamental knowledge representation, problem solving, machine learning, knowledge acquisition, and learning methodologies in the solution of specific engineering challenges.**
UCS761	**Deep Learning**	**Utilized deep learning methods like neural networks and transfer learning to build the classifiers.**
UCS614	**Embedded System Design**	**Outlines the concept of Internet of Things.**

Table 7.2 Relevant Subjects

7.3 Interdisciplinary Knowledge Sharing

In this project, we applied the ideas of the Internet of Things learned in Embedded System Design and Wireless and Mobile Communication topics to record a patient's vitals. We used Data Analytics principles to assist illustrate the data once we finalised the product to be used. Furthermore, Machine Learning and Artificial Intelligence principles aided us in developing the model responsible for establishing the threshold values. Artificial Intelligence had helped us brief up with the concepts of Python which helped in the 70 implementations of the machine learning model. Software Engineering ideas were helpful in drafting the report and creating project diagrams. Database management was also performed at the backend to store and process user-related information utilising the Database Management System course.

		Evaluation of				
		Aayushi	**Harshvir**	**Manvendra**	**Tanmay**	**Twiny**
Evaluation By	**Aayushi**	1	4.5	4	4.5	5
	Harshvir	4.5	1	5	4	4.5
	Manvendra	4	4.5	1	5	4.5
	Tanmay	4.5	5	4.5	1	4
	Twiny	5	4	4.5	4.5	1

Table 7.4: Peer Assessment Matrix

7.5 Role Playing and Work Schedule

The individual roles of each team member were as following:

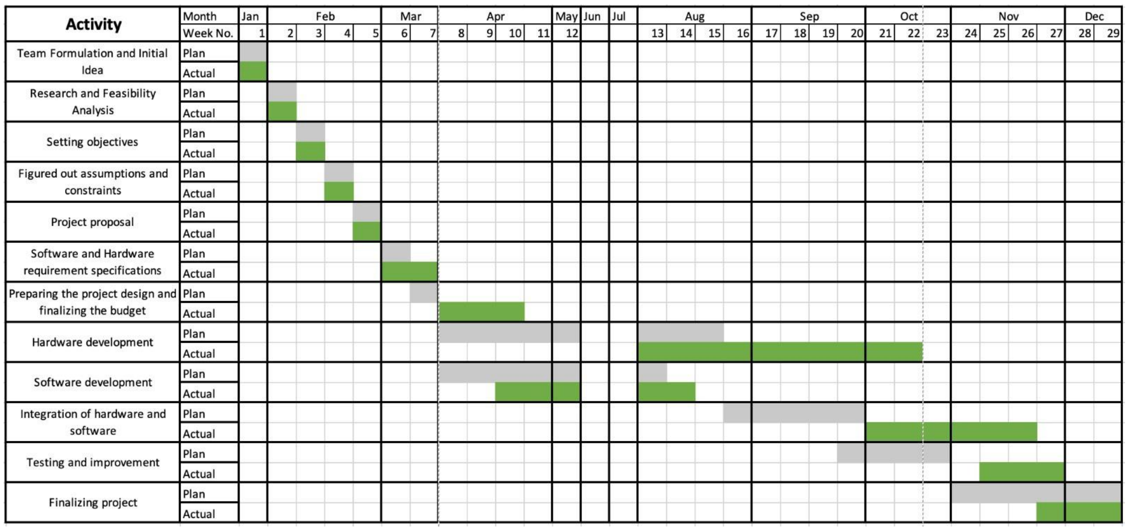
Manvendra Raj Singh: User Interface (UI) Development, Backend Development

Tanmay Relan: Hardware Setup and Integration, Computer Vision Model (FOMO)

Aayushi Bareja: Computer Vision Model (FOMO) development, Trained and tested the model using Edge Impulse for object detection.

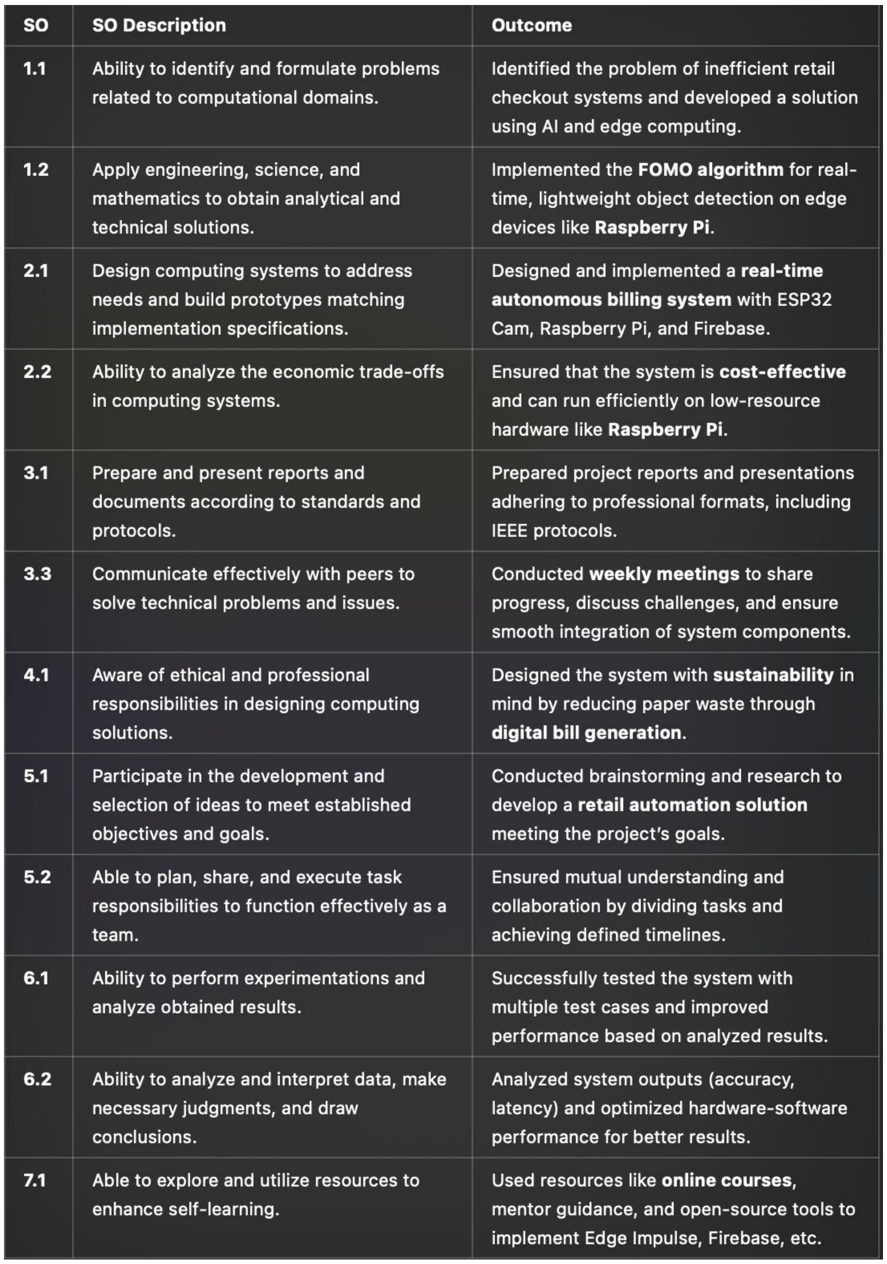
Harshvir Singh: Computer Vision Model (FOMO) development, Dataset generation.

Twiny: Hardware Setup and Integration, Dataset generation.



Activity	Month	Jan	Feb				Mar		Apr				May	Jun	Jul	Aug				Sep				Oct			Nov				Dec	
	Week No.	1	2	3	4	5	6	7	8	9	10	11	12			13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
Team Formulation and Initial Idea	Plan	▨																														
	Actual	🟩																														
Research and Feasibility Analysis	Plan		▨																													
	Actual		🟩																													
Setting objectives	Plan			▨																												
	Actual			🟩																												
Figured out assumptions and constraints	Plan				▨																											
	Actual				🟩																											
Project proposal	Plan					▨																										
	Actual					🟩																										
Software and Hardware requirement specifications	Plan						▨																									
	Actual						🟩	🟩																								
Preparing the project design and finalizing the budget	Plan							▨																								
	Actual								🟩	🟩	🟩																					
Hardware development	Plan								▨	▨	▨	▨	▨			▨	▨	▨														
	Actual															🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩							
Software development	Plan								▨	▨	▨	▨	▨			▨																
	Actual										🟩	🟩	🟩			🟩	🟩															
Integration of hardware and software	Plan																		▨	▨	▨	▨	▨									
	Actual																							🟩	🟩	🟩	🟩	🟩	🟩	🟩		
Testing and improvement	Plan																						▨	▨	▨	▨						
	Actual																											🟩	🟩	🟩		
Finalizing project	Plan																										▨	▨	▨	▨	▨	▨
	Actual																													🟩	🟩	🟩

7.6 Student Outcomes Description and Performance Indicators (A-K Mapping)



SO	SO Description	Outcome
1.1	Ability to identify and formulate problems related to computational domains.	Identified the problem of inefficient retail checkout systems and developed a solution using AI and edge computing.
1.2	Apply engineering, science, and mathematics to obtain analytical and technical solutions.	Implemented the **FOMO algorithm** for real-time, lightweight object detection on edge devices like **Raspberry Pi**.
2.1	Design computing systems to address needs and build prototypes matching implementation specifications.	Designed and implemented a **real-time autonomous billing system** with ESP32 Cam, Raspberry Pi, and Firebase.
2.2	Ability to analyze the economic trade-offs in computing systems.	Ensured that the system is **cost-effective** and can run efficiently on low-resource hardware like **Raspberry Pi**.
3.1	Prepare and present reports and documents according to standards and protocols.	Prepared project reports and presentations adhering to professional formats, including IEEE protocols.
3.3	Communicate effectively with peers to solve technical problems and issues.	Conducted **weekly meetings** to share progress, discuss challenges, and ensure smooth integration of system components.
4.1	Aware of ethical and professional responsibilities in designing computing solutions.	Designed the system with **sustainability** in mind by reducing paper waste through **digital bill generation**.
5.1	Participate in the development and selection of ideas to meet established objectives and goals.	Conducted brainstorming and research to develop a **retail automation solution** meeting the project's goals.
5.2	Able to plan, share, and execute task responsibilities to function effectively as a team.	Ensured mutual understanding and collaboration by dividing tasks and achieving defined timelines.
6.1	Ability to perform experimentations and analyze obtained results.	Successfully tested the system with multiple test cases and improved performance based on analyzed results.
6.2	Ability to analyze and interpret data, make necessary judgments, and draw conclusions.	Analyzed system outputs (accuracy, latency) and optimized hardware-software performance for better results.
7.1	Able to explore and utilize resources to enhance self-learning.	Used resources like **online courses**, mentor guidance, and open-source tools to implement Edge Impulse, Firebase, etc.

Table 7.6 Student Outcome Description

7.7 Brief Analytical Assessment

Q1. What sources of information did your team explore to arrive at the list of possible Project Problems?

We identified challenges by researching existing autonomous checkout systems and analyzing their limitations through IEEE journals, technical papers, and official documentation. Consultations with our mentor further refined the project scope.

Q2. What analytical, computational, and/or experimental methods did your project team use to obtain solutions to the problems in the project?

We used lightweight ML models like FOMO, transfer learning for enhanced accuracy, and experimental methods to validate hardware-software integration and real-time data processing.

Q3. Did the project demand demonstration of knowledge of fundamentals, scientific, and/or engineering principles? If yes, how did you apply?

Yes, we applied principles of ML, IoT, and software engineering for object detection, hardware integration, and UI development. Database management concepts were used for organizing price lists and training data.

Q4. How did your team share responsibility and communicate the information of schedule with others in a team to coordinate design and manufacturing dependencies?

The project was divided into modules and assigned to team members based on expertise. Weekly meetings ensured collaboration, progress tracking, and resolution of dependencies.

Q5. What resources did you use to learn new materials not taught in class for the course of the project?

We referred to official documentation, YouTube tutorials, Coursera courses, Stack Overflow, and research articles. Guidance from our mentor helped address complex challenges.

Q6. Does the project make you appreciate the need to solve problems in real life using engineering, and could the project development make you proficient with software development tools and environments?

Yes, the project provided hands-on experience in solving real-world problems and improved our proficiency with tools like Edge Impulse, web development frameworks, and communication protocols

.

REFERENCES

[1] Ariponnammal, S. and Natarajan, S. (1994) ‘Transport Phonomena of SmSel – X Asx’, Pramana – Journal
of Physics Vol.42, No.1, pp.421-425.

[2] Anderson T. , Peterson L., Shenker S., Turner J.(2005).Overcoming the Internet impasse through
virtualization. IEEE Computer, 38(4):34-41.

[3] W. Zeng, H. Yu, C. Lin. (2013, Dec 19). Multimedia Security Technologies for Digital Rights
Management [Online]. Available: http://goo.gl/xQ6doi. Accessed on [Date]