A

Report

on

AI Shop Sync

Submitted in partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering
By

Priyanshi Tripathi
Roll No. 2202220100137

Anjali Kushwaha
Roll No. 2202221530004

Divya Arya
Roll No. 2202221530019

Sahil Gupta
Roll No. 2202220100148

Semester – VII
Session - 2024-25

Under the Supervision of

Ms. Akansha Sharma

Assistant Professor - CSE

THE EDUCATION GROUP

I.T.S Engineering College, Greater Noida

Affiliated to

Dr. APJ Abdul Kalam Technical University, Lucknow

CERTIFICATE

This is to certify that the Mini Project Report on "**AI Shop Sync**" by **Priyanshi Tripathi**, **Anjali Kushwaha**, **Divya Arya** and **Sahil Gupta** have been submitted for the partial fulfilment of the requirements of B.Tech. in Computer Science and Engineering (CSE). The work is carried out under my supervision and free from plagiarism.

Ms. Akansha Sharma Asst. Professor – CSE (Supervisor)	Ms. Priyanshi Tripathi 2202220100137
Dr. Hariom Tyagi Professor – CSE (Project Coordinator)	Ms. Anjali Kushwaha 2202221530004
Dr. Jaya Sinha Head of Department – CSE	Ms. Divya Arya 2202221530019
Dr. Vishnu Sharma Dean – CSE	Mr. Sahil Gupta 2202220100148

INDEX

1. INTRODUCTION

1.1 Introduction:

In the modern retail landscape, customers are increasingly pursuing smarter and more efficient ways to make informed purchasing decisions, particularly in grocery shopping. With the explosive growth of online grocery delivery platforms like Instamart, Zepto, Blinkit, and others, consumers face a plethora of options that promise convenience but also lead to confusion due to fluctuating prices, varying delivery charges, and numerous promotional offers. These dynamics make it challenging for shoppers to consistently identify the most cost-effective options for their everyday essentials.

The rapid expansion of the digital shopping ecosystem has raised users' expectations for real-time, automated assistance in simplifying their purchasing decisions. Traditional manual searches—where users must open multiple apps, compare items one-by-one, and constantly check for price updates—are inefficient, time-consuming, and frustrating. This inefficiency often results in missed savings and less optimal purchasing choices, creating a gap in the user experience.

Given this context, there is a pressing need for an intelligent, mobile-first solution that streamlines this decision-making process, offering users instant access to accurate product information and real-time price comparisons across platforms. AI Shop Sync addresses this need by harnessing cutting-edge technologies including Optical Character Recognition (OCR) and Natural Language Processing (NLP) to transform the grocery shopping journey into a seamless, cost-effective, and user-friendly experience.

1.2 What is AI Shop Sync?

AI Shop Sync is an innovative OCR-based price comparison mobile application designed to empower consumers by facilitating instant recognition and comparison of grocery product prices across multiple platforms. The system utilizes advanced text recognition technology to extract detailed product information—such as brand name, product name, quantity, and Maximum Retail Price (MRP)—from images of product packaging captured via a user's smartphone camera.

Once extracted, these textual details undergo normalization and standardization to resolve variations in product naming conventions, aligning similar products across different platforms. For example, entries like "Amul Taza Milk 1L" and "Amul Milk 1000ml" are recognized as the same product to eliminate inconsistencies and improve the reliability of comparisons. AI Shop Sync cross-references this normalized data with both an internal product catalog and real-time pricing information fetched from multiple online grocery platforms using APIs or web scraping techniques.

Beyond simple price checking, AI Shop Sync is designed to be a smart assistant for everyday grocery shopping, providing a modular and scalable system adaptable to diverse retail ecosystems—both online and offline. Its mobile-first design ensures accessibility for users on the go, while its intelligent processing transforms raw data into meaningful insights that empower consumers to easily identify the best deals without exhaustive manual effort.

1.3 How AI Shop Sync Work?

1.3.1 The user initiates a scan of a product's packaging using their smartphone camera via the AI Shop Sync mobile app, which captures high-resolution images in real time.

1.3.2 The built-in OCR engine processes the captured image to accurately extract key textual details including the product name, brand, weight or quantity, and MRP.

1.3.3 Recognizing that raw OCR output often includes noise such as expiry dates, batch numbers, or slogans, the system applies sophisticated Natural Language Processing (NLP) techniques to cleanse and normalize the text. This step ensures matching against a standardized product database is both precise and consistent.

1.3.4 After normalization, the system utilizes a canonical mapping layer to unify different textual representations of the same product, enabling cross-platform consistency. This is central to making accurate price comparisons across varied retailer terminologies.

1.3.5 The backend server queries multiple grocery platforms (e.g., Instamart, Zepto, Blinkit) via APIs or scraping modules to retrieve up-to-date pricing and availability information for the identified product.

1.3.6 The application aggregates and analyzes the retrieved pricing data and displays it in a comparative, user-friendly interface, allowing users to immediately see which platform offers the best deal at that moment.

1.3.7 Additionally, users can add products to an in-app shopping cart. AI Shop Sync further optimizes the shopping experience by analyzing the cart contents and recommending whether purchasing all items from a single platform or splitting across multiple platforms will yield the lowest total cost, factoring in price variations and delivery fees.

More than just a basic scanning tool, AI Shop Sync incorporates advanced filtering, matching algorithms, and machine learning techniques to fine-tune product identification and pricing insights. By eliminating irrelevant information and focusing on accuracy, it converts raw data into actionable smart shopping recommendations tailored to users' preferences and needs. This transforms grocery shopping from a routine task into an intelligent, savings-maximizing activity, making AI Shop Sync a vital assistant for today's cost-conscious consumers in a competitive digital retail environment.

2. LITERATURE REVIEW

2.1. Multimodal Fine-Grained Grocery Product Recognition (2024) [6]

Objective: This study proposes a multimodal recognition framework that combines image-based embeddings with OCR-extracted textual data to achieve fine-grained classification of grocery products. It addresses challenges in differentiating visually similar items—such as different flavors, package sizes, or variants—by effectively leveraging textual information printed on product packaging. The framework enhances recognition accuracy by integrating both visual and textual signals and improves robustness to environmental factors like varied lighting and packaging orientations that typically degrade image-only systems.

Relevance to the Project: AI Shop Sync heavily relies on accurate extraction of textual data from product packaging using OCR. The approach of combining OCR text with visual features offers valuable insights for improving recognition accuracy, especially for closely resembling products. By adopting a similar multimodal strategy, AI Shop Sync can reduce mapping errors and increase the reliability of price comparisons, ultimately improving user trust and satisfaction. This approach also helps the system better manage noisy OCR outputs and packaging variability in practical scenarios.

2.2. PP-OCR: A Practical Ultra Lightweight OCR System (2020, PaddleOCR) [2]

Objective: PP-OCR introduces a lightweight OCR pipeline optimized for deployment on resource-constrained devices such as smartphones and embedded systems. It balances text detection and recognition accuracy with inference speed, incorporating multilingual and font-style support to cater to diverse real-world applications. The system's modular design enables efficient adaptation to various use cases, including those requiring on-device, real-time OCR processing.

Relevance to the Project: AI AI Shop Sync stands to gain significantly by adopting or fine-tuning lightweight OCR models like PP-OCR, enabling fast and highly accurate text extraction directly on mobile devices. This local, on-device processing dramatically reduces reliance on server-side computation, minimizing latency and improving response times critical for a seamless user experience. Such efficiency enhances the app's scalability by lowering backend resource consumption and enabling it to support many concurrent users without degradation in performance. Paddle OCR's strong support for multilingual text recognition and stylized fonts aligns well with the diverse and complex nature of grocery packaging. Supermarket products often feature creative typography, regional scripts, and multiple languages, and Paddle OCR's ability to handle these variations ensures robust and inclusive product recognition. Moreover, the modular architecture of Paddle OCR facilitates continuous improvements such as recognizing curved, rotated, or distorted text, which are common occurrences in real-world packaging like bottles, cans, and flexible pouches.

2.3. Retail-786k: Large-Scale Dataset for Visual Entity Matching (2023/2024) [7]

Objective: Retail-786k presents an extensive and diverse collection of high-resolution product images specifically curated for visual entity matching in the retail sector. The dataset captures a wide spectrum of real-world retail complexities by including substantial variations in packaging design, lighting conditions, image resolutions, and viewing angles. These challenges simulate authentic retail environments, making the dataset a critical resource for training and benchmarking product identification systems. Moreover, the dataset annotates products into semantic groups known as entities, representing clusters of visually and semantically equivalent products, which helps address the challenge of accurately associating diverse product images with their canonical identities at scale.

Relevance to the Project: For AI Shop Sync, Retail-786k provides foundational data and benchmarks essential for developing reliable product-mapping algorithms that link OCR-extracted textual data with standardized product entries. This capability is vital to maintaining consistency and accuracy across different online retail platforms during price comparison. By leveraging insights from the dataset, AI Shop Sync can better manage noisy OCR output and visually similar packaging, reducing false positives and improving classification precision. Furthermore, Retail-786k's comprehensive coverage of intra- and inter-entity visual variance informs the design of robust matching models capable of adapting to frequent changes in product appearances. This enhances AI Shop Sync's scalability and accuracy, ensuring it delivers dependable price comparison results throughout the dynamic and evolving retail landscape.

2.4. Deep Learning-Based Product Recognition for Retail (2023) [10]

Objective: This research proposes a deep learning framework that seamlessly integrates convolutional neural networks (CNNs) with textual analysis derived from optical character recognition (OCR) outputs to substantially elevate product recognition accuracy within retail environments. The approach is designed to overcome prevalent challenges such as image occlusions, reflective surfaces, and packaging variations by training on extensive, annotated retail datasets. It utilizes CNNs to extract rich visual features that complement OCR-extracted text, enabling a more comprehensive understanding and classification of products. The framework also addresses real-time constraints critical for practical use by optimizing both computational efficiency and recognition speed, making it suitable for deployment in busy retail settings with dense product arrangements and dynamically changing packaging designs.

Relevance to the Project: The integration of CNN-based image recognition with OCR text analysis aligns directly with AI Shop Sync's goals of achieving multimodal product identification to improve recognition precision. This combined approach allows AI Shop Sync to robustly detect and classify items even in visually complex scenarios typical of grocery stores, such as shelf clutter, shadows, and partial occlusions. Moreover, its adaptability to evolving packaging aesthetics plays a vital role in sustaining the accuracy and relevancy of the product database over time, which is crucial for maintaining user trust in price comparisons. Furthermore, the deep learning methods underpinning this research provide a scalable solution that can be fine-tuned as more product data becomes available, thereby continuously enhancing AI Shop Sync's performance and broadening its applicability across diverse retail contexts.

2.5. Real-Time Mobile Price Comparison Using OCR and Web APIs (2022) [11]

Objective: This paper details the design of a mobile system that performs on-device optical character recognition (OCR) to accurately identify product labels, followed by real-time retrieval of pricing information from multiple e-commerce APIs. The focus is on establishing a minimalist and efficient processing pipeline that maintains high responsiveness and low latency on mobile platforms. By emphasizing lightweight data processing and careful API integration, the system ensures synchronized, accurate price data across multiple sources without compromising user experience. The architecture supports scalable performance while minimizing battery consumption and data usage, addressing critical constraints of mobile devices.

Relevance to the Project: AI Shop Sync's architectural blueprint draws significant inspiration from this approach, particularly in its emphasis on lightweight OCR processing directly on users' smartphones combined with real-time price aggregation from diverse online grocery platforms. The use of minimalist yet robust pipelines enables AI Shop Sync to offer instant price comparisons with accurate, up-to-date data while keeping resource utilization in check—a necessity for mobile app performance and user retention. Furthermore, the design principles of seamless API integration and result aggregation enhance backend efficiency and data consistency, enabling AI Shop Sync to reliably provide users with trustworthy and comprehensive shopping decision support even in bandwidth-limited or intermittent connectivity scenarios. Such scalable, mobile-friendly design choices are pivotal in ensuring AI Shop Sync's wide adoption and sustained usage in highly competitive retail environments.

3. METHODOLOGY

3.1 Requirement Gathering and Analysis

3.1.1. Identify the scope of supported products, including FMCG, groceries, and household items, ensuring relevant market coverage

3.1.2. Collect a diverse dataset encompassing product packaging samples from varied brands, fonts, and multiple languages to address real-world variability.

3.1.3. Define key user requirements focusing on fast scanning speeds, high OCR accuracy, cross-platform support, and real-time price comparison capability to cater to user expectations.

3.1.4. Enumerate functional requirements including seamless image capture, precise text extraction, dynamic price retrieval from multiple online sources, and efficient cart management for budget planning.

3.2 Technology Stack Selection

3.2.1 The frontend is developed using React Native to achieve cross-platform compatibility, employing tools such as expo-camera or react-native-vision-camera for robust camera access, with additional native Android support to optimize performance on Android devices.

3.2.2 For OCR, multiple engines are evaluated: Tesseract OCR provides offline text extraction capability, while PaddleOCR and Google ML Kit offer online processing with enhanced accuracy and broader language support. Testing with these OCR options at the backend using Python scripts has been conducted, with optimization efforts ongoing for balancing speed and accuracy.

3.2.3 The backend is constructed with Node.js leveraging the Express.js framework to establish scalable RESTful APIs. Data scraping from grocery platforms is facilitated through Playwright and HTTP libraries, ensuring comprehensive access to pricing data.

3.2.4 MongoDB is utilized for managing product and price data due to its scalability, flexibility, and seamless integration with Node.js environments.

3.2.5 AI and NLP techniques, including string matching algorithms and text normalization, are implemented to convert OCR raw outputs into normalized, standardized product names required for consistent price comparison across platforms.

3.3 System Architecture Design

3.3.1 The system follows a client-server architecture where the mobile application acts as the client, handling image capture and initial OCR, while the backend processes data retrieval, normalization, and price comparison.

3.3.2 Users capture product packaging images in real time using their mobile device. The OCR engine then extracts textual information such as brand, product name, weight, and MRP from these images.

3.3.3 An NLP module cleanses and normalizes this extracted text to match canonical product entries, accommodating variations like spelling differences and unit representations (e.g., “500g” vs “0.5kg”).

3.3.4 The backend communicates with multiple grocery platforms using API calls or web scraping to fetch real-time price data based on the normalized product identifications. This step is repeated efficiently for each product both individually and in grouped cart scenarios.

3.3.5 Users can build and manage their shopping cart within the application, where intelligent analysis suggests optimal purchasing options by comparing prices across platforms and factoring in delivery charges.

3.4 Module Development

3.4.1 **OCR Module:** Responsible for extracting high-quality text from images; currently utilizes Google ML Kit in the React Native app for on-device OCR, and Python-based testing of Tesseract, PaddleOCR, and EasyOCR at the backend level, pending further optimization.

3.4.2 **Normalization Module:** Implements sophisticated text cleaning, matching, and standardization procedures to ensure consistent product identification despite packaging discrepancies.

3.4.3 **Price Fetching Module:** Connects to grocery platform APIs or scrapes websites, retrieving and updating pricing data continuously to provide users with accurate, real-time comparisons.

3.4.4 **Cart Module:** Allows users to save products and view aggregate pricing, enabling budget tracking and buy-split suggestions to minimize overall cost.

3.4.5 **UI/UX Module:** Presents scanned product details and comparative pricing in an intuitive, easy-to-navigate interface designed from completed Figma prototypes.

3.5 **Testing and Validation**

3.5.1 Unit tests evaluate OCR accuracy in diverse lighting and orientation conditions, ensuring text extraction robustness.

3.5.2 Integration tests verify the end-to-end process from OCR extraction, text normalization, product mapping, to price retrieval, ensuring system reliability and correctness.

3.5.3 Field testing in real supermarket environments assesses practical usability, system responsiveness, and overall user satisfaction, guiding iterative improvements**.**

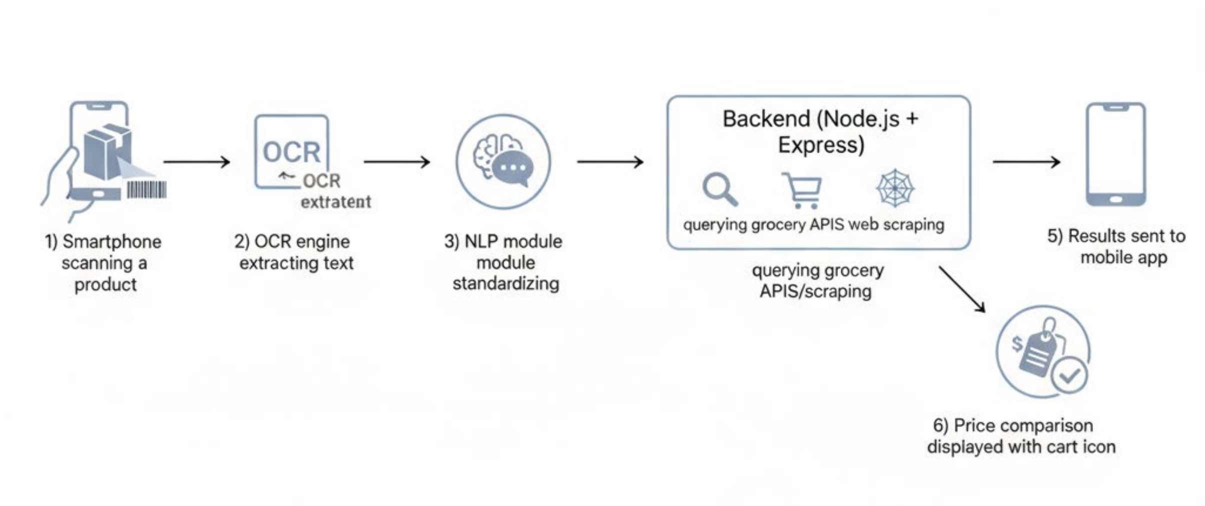

OCR
OCR
extratent
1) Smartphone scanning a product
2) OCR engine extracting text
3) NLP module module standardizing
Backend (Node.js + Express)
querying grocery APIS web scraping
querying grocery APIS/scraping
5) Results sent to mobile app
$
6) Price comparison displayed with cart icon

Fig 3.1 Development Flowchart

4. BACKEND DEVELOPMENT

4.1 Backend Architecture and Technologies

4.1.1. The backend is built using Node.js combined with the Express.js framework to create a scalable, flexible, and modular RESTful API architecture. This setup supports efficient handling of numerous concurrent requests critical for real-time price comparison functionalities.

4.1.2. Web scraping modules are implemented using Playwright along with HTTP libraries to dynamically fetch fresh pricing data and product availability from multiple grocery platforms, ensuring that the app reflects up-to-the-minute market conditions.

4.1.3. MongoDB serves as the primary database, chosen for its schema-less design that easily accommodates the diverse and evolving data formats of product information, user details, and transactional history. Integration with Node.js is seamless, allowing smooth CRUD operations and high performance for complex queries.

4.1.4. Environment variables and configuration files securely store sensitive information like API keys and database credentials, ensuring secure deployment practices.

4.2 Technology User Authentication and Security

4.2.1 JSON Web Token (JWT) authentication has been successfully implemented, providing stateless and scalable user session management. This facilitates secure login/logout features across platforms without server-side session storage overhead.

4.2.2 All API endpoints that access or manipulate user data are protected using JWT validation middleware, guaranteeing that only authenticated users can access sensitive resources or perform restricted actions.

4.2.3 Role-based access controls can be integrated for features requiring different permission levels, enhancing system security and user data privacy.

4.3 Price Fetching and Data Aggregation

4.3.1 The backend efficiently aggregates price information across multiple platforms by orchestrating simultaneous API calls and web scraping operations.

4.3.2 Data normalization techniques are applied to unify product naming conventions and measurement units, allowing accurate cross-platform price comparison despite inconsistencies in source data.

4.3.3 Pricing data is cached with controlled expiry policies to improve response times while balancing freshness of information.

4.3.4 Backend services include error handling and retry mechanisms to manage temporary failures from external grocery platforms, ensuring reliability and seamless user experience.

4.4 OCR Testing and Product Mapping

4.4.1 Extensive backend-level testing of multiple OCR engines—Tesseract, PaddleOCR, and EasyOCR—is done via Python scripts to benchmark accuracy and performance on typical product packaging images. Though not fully optimized yet, these tests guide the selection of the most suitable OCR model for integration.

4.4.2 A dedicated normalization module cleanses OCR outputs, applies string matching algorithms, and links extracted text to standardized product records in the database.

4.4.3 Continuous improvements include noise filtering to exclude irrelevant text (e.g., expiry dates, batch numbers) and corrections for OCR misreadings, crucial for reliable product identification and price matching.

5. UI/UX AND REACT NATIVE APPLICATION DEVELOPMENT

5.1 UI/UX Design Overview

The UI/UX for AI Shop Sync has been thoughtfully designed in-house to provide a clean, modern, and user-friendly interface tailored specifically for grocery price comparison and shopping assistance. The design emphasizes intuitive navigation, simplicity, and accessibility, ensuring that users of all technical backgrounds can effortlessly scan products, compare prices, and manage their shopping within a seamless experience. Each screen is crafted to balance functionality with aesthetic appeal, creating a cohesive and engaging user journey throughout the app.

5.2 Screen Description

5.2.1 **LoginScreen:** Allows users to securely log into the app using email and password inputs, with options for password recovery and social media sign-in. The screen features clean input forms and a visually appealing, welcoming layout with clear call-to-action buttons.

5.2.2 **RegisterScreen:** Enables new users to create accounts by entering details such as name, email, password, and phone number. Inline form validation immediately informs users about input errors to streamline onboarding. The design maintains consistency with the login screen for a cohesive authentication flow.

5.2.3 **HomeScreen:** The app's dashboard, showing personalized greetings, featured deals, and quick access to key functionalities like scanning, categories, and search. It displays curated product lists and banners highlighting discounts or popular items, ensuring users can easily start browsing or shopping.

5.2.4 **ScannerScreen:** Integrates directly with the device camera using React Native Vision Camera and React ML Kit's OCR capabilities to enable real-time scanning of product packaging. This screen provides users with an interactive camera viewfinder, scanning feedback, and quick results display after text extraction.

5.2.5 **CartScreen:** Displays items added by the user, showing product details, quantities, individual prices, and total cost. It supports editing quantities, removing items, and presents price comparison insights for possible savings, helping users make informed purchase decisions.

5.2.6 **ProductScreen:** Lists products under selected categories or search results with thumbnail images, brief descriptions, pricing, and ratings. It facilitates easy browsing and quick access to detailed product information.

5.2.7 **ProductDetailScreen:** Provides comprehensive details about a single product, including images, full description, nutritional information, user reviews, and price comparison table across grocery platforms. Clear actionable buttons let users add the product to their cart or wishlist.

5.2.8 **SearchScreen:** Offers advanced search capabilities with filters for brands, price range, categories, and ratings. It displays real-time suggestions as users type and shows relevant results quickly, enhancing product discovery.

5.2.9 **ProfileScreen:** Allows users to view and update personal information, manage saved delivery addresses, payment methods, and app settings including notifications and preferences. It also provides order history and support access.

5.2.10 **CategoriesScreen:** Organizes products into intuitive categories and subcategories with clear visuals and quick navigation, enabling users to explore product groups efficiently.

5.3 React Native Application Modules

5.3.1 **JWT Authentication Module:** Handles secure user authentication with JWT tokens for login, registration, and session management. The module includes token storage and refresh logic to maintain persistent user sessions. API calls are protected with token validation to secure sensitive endpoints.

5.3.2 **HomeScreen Module:** Renders dynamic personalized content fetched from backend APIs, including featured product lists, promotional banners, and user-specific recommendations. Utilizes efficient state management to ensure smooth UI responsiveness.

5.3.3 **ScannerScreen Module:** Implements real-time OCR-based product scanning by integrating React Native Vision Camera for high-performance camera access, combined with React ML Kit for efficient text extraction. It provides user feedback during scanning, error handling for no-detection cases, and navigates seamlessly to product details or comparison results post scan.

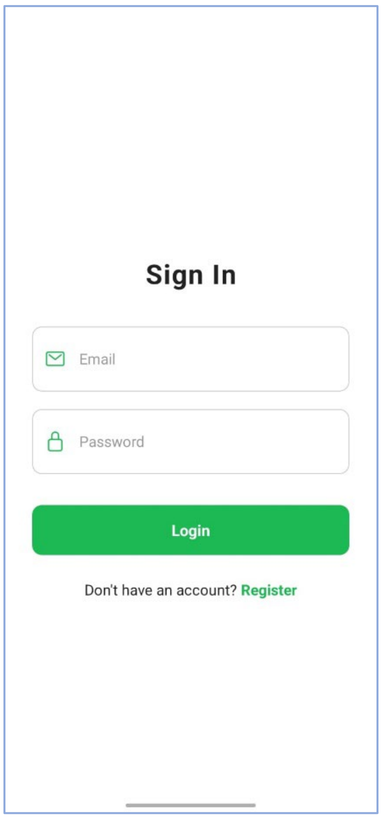


Fig 5.1 Login Screen

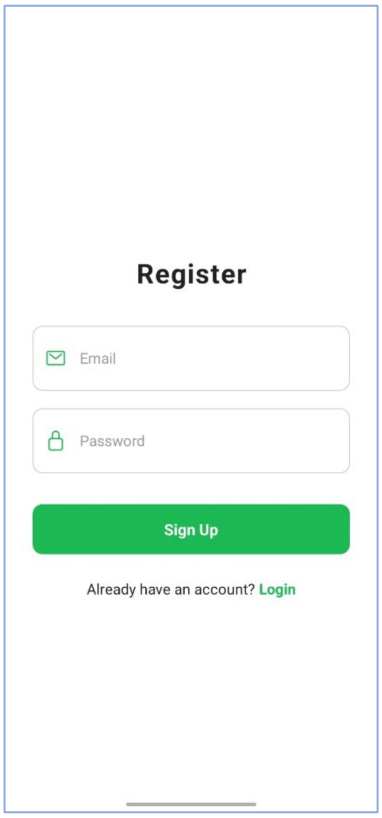


Fig 5.2 Register Screen

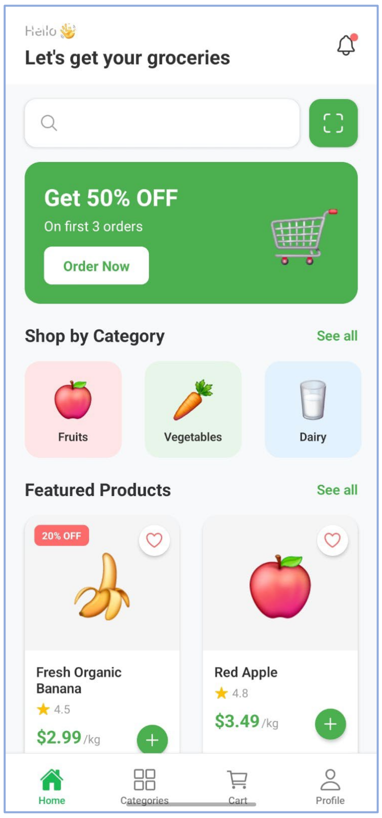
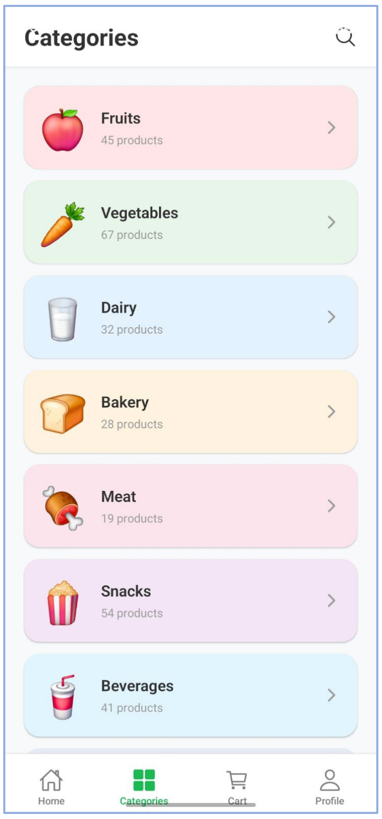
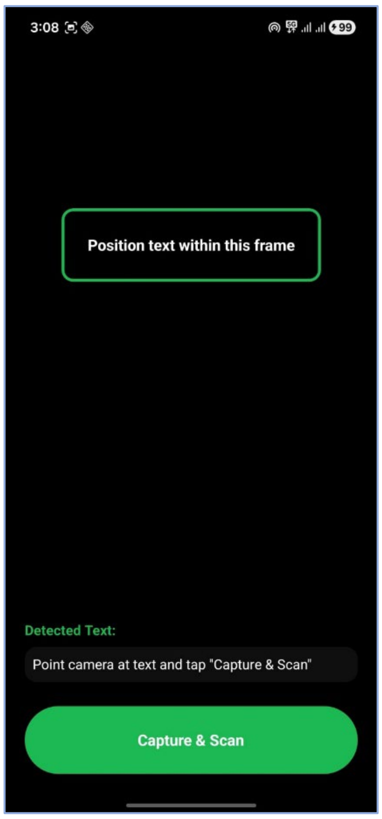


Fig 5.3 Home Screen




3:08
Position text within this frame
Detected Text:
Point camera at text and tap "Capture & Scan"
Capture & Scan

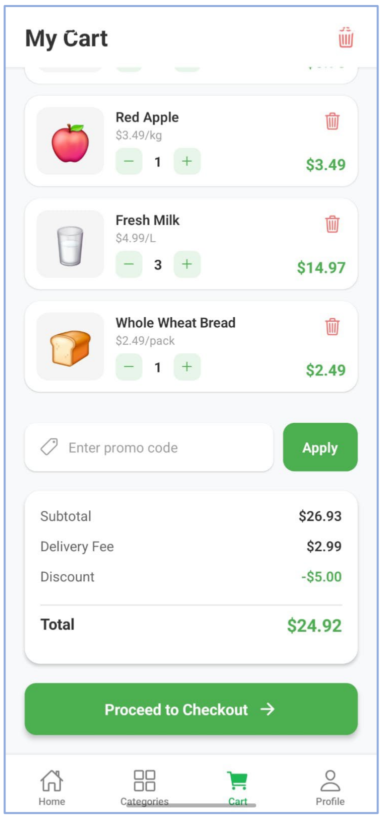


Fig 5.4 Category Screen **Fig 5.5 Scanner Screen** **Fig 5.3 Cart Screen**

6. OCR MODULE

6.1 OCR Engines Evaluated at Backend

6.1.1 Multiple OCR engines were tested at the backend level using Python scripts, including PaddleOCR, EasyOCR, and Tesseract OCR, to evaluate their text recognition accuracy, speed, and robustness on varied product packaging images.

6.1.2 Each engine demonstrated strengths and weaknesses: PaddleOCR offered a lightweight and fast solution, Tesseract delivered broad language support but was slower, and EasyOCR showed promise with certain fonts but struggled with noisy images.

6.1.3 Despite reasonable performance from these engines in controlled tests, none fully met the speed and accuracy requirements for real-time, user-facing scanning in the mobile app environment.

6.2 ML Kit Integration in React Native App

6.2.1 Paddle OCR

6.2.1.1 Advantages:

6.2.1.1.1 Highly optimized for lightweight and fast text detection, making it suitable for real-time applications.

6.2.1.1.2 Supports multiple languages and handles stylized fonts fairly well.

6.2.1.1.3 Modular design allows integration of text detection and recognition components independently.

6.2.1.2 Drawbacks:

6.2.1.2.1 Requires environment setup and dependencies that can be complex for some backend deployments.

6.2.1.2.2 Although fast, in practical usage it may struggle with highly distorted or curved texts on uneven packaging surfaces.

6.2.1.2.3 Accuracy can vary significantly depending on image quality and preprocessing steps.

6.2.2 EasyOCR

6.2.2.1 Advantages:

6.2.2.1.1 Open-source with support for numerous languages.

6.2.2.1.2 Performs reasonably well on clean, clear images with standard fonts.

6.2.2.2 Drawbacks:

6.2.2.2.1 Less robust when dealing with noisy backgrounds or complex packaging fonts.

6.2.2.2.2 Can be slower compared to more optimized OCR engines, affecting real-time usability.

6.2.2.2.3 Offers limited customization or fine-tuning capabilities.

6.2.3 Tesseract OCR

6.2.3.1 Advantages:

6.2.3.1.1 One of the oldest and widely adopted open-source OCR engines with extensive language and script support.

6.2.3.1.2 Highly customizable with training options for specialized fonts.

6.2.3.2 Drawbacks:

6.2.3.2.1 Processing speed is slower, which reduces feasibility for real-time mobile applications.

6.2.3.2.2 Susceptible to errors in low-quality images or complex backgrounds without significant preprocessing.

6.2.3.2.3 Integration and tuning can be non-trivial, requiring expertise for best results.

6.3 Google ML Kit (Used in React Native App)

6.3.1 ML Kit offers an easy-to-integrate OCR solution designed specifically for mobile platforms, providing fast, on-device text recognition without the latency associated with server-side processing.

6.3.2 It effectively handles multiple languages, diverse fonts, and various orientations of text on product packaging, delivering superior accuracy compared to backend OCR alternatives tested.

6.3.3 The on-device nature of ML Kit enhances privacy, as images do not need to leave the user's device.

6.3.4 Integration with React Native through libraries like react-native-mlkit-ocr simplifies development, enabling quick deployment and iteration.

6.3.5 Overall, ML Kit balances ease of use, speed, and accuracy, making it the optimal OCR choice for AI Shop Sync’s real-time product scanning needs.

7 REFERENCES

[1] R. Smith, “An Overview of the Tesseract OCR Engine,” Proc. Int. Conf. Document Analysis and Recognition (ICDAR), 2007.

[2] Duan, Y., et al. “PaddleOCR: A Practical Ultra Lightweight OCR System,” *arXiv preprint arXiv:2009.09941*, 2020.

[3] Google ML Kit, "On-device Text Recognition," 2023.

[4] Li, Y., et al. “Mobile Price Comparison Systems: A Survey,” *Journal of Retail Technology*, 2019.

[5] Wagh, A., et al. “Comparison of Image-based vs. Barcode-based Product Recognition in Retail,” *International Journal of Computer Applications*, 2021.

[6] Multimodal Fine-Grained Grocery Product Recognition, 2024.

[7] Retail-786k: Large-Scale Dataset for Visual Entity Matching, 2023/2024

[8] Actowiz Solutions, “Grocery Price Comparison and Scraping,” 2025.

[9] Daltix, “Retail Product Matching: Challenges & Solutions,” 2025.

[10] Deep Learning-Based Product Recognition for Retail, 2023.

[11] Real-Time Mobile Price Comparison Using OCR and Web APIs, 2022