IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141268

Smart Billing Application

Ms. Vidyasre N [1], Pavan T L [2], Niveda B [3], Mansi M [4], and Marineni Hansika [5]

Assistant Prof., Dept of Computer Science, K.S.School of Engineering & Management, Bengaluru, India[1]

Student, Dept of Computer Science, K.S.School of Engineering & Management, Bengaluru, India[2-5]

Abstract: Traditional retail checkout processes are often characterised by inefficiencies, leading to long queues and a decrease in customer satisfaction. This paper proposes "Smart Cart," an end-to-end automated billing system to streamline the in-store shopping experience. The proposed solution utilises a modern web technology stack-MongoDB, Express.js, React, Node- to create a responsive user interface and a robust backend. Key features include real-time barcode scanning through a device camera or hardware integration, dynamic cart management with budget tracking, and a secure wallet-based payment system. To guarantee transaction integrity, the system implements a checkout verification algorithm that cross-checks the software cart data against the hardware sensor data, such as weight and item count. This system also integrates accessibility options and an AI-powered chatbot for nutritional analysis, thus demonstrating a highly scalable and user-centric approach toward the automation of retail.

Keywords: Smart Cart, Automated Billing, Internet of Things (IoT), React.js, Node.js, Generative AI, Web Application, Retail Automation.

I. INTRODUCTION

The retail sector is witnessing a paradigm shift towards automation, driven by the need to enhance customer experience and operational efficiency. Traditional Point of Sale (POS) systems, while reliable, often create bottlenecks during peak hours, resulting in long wait times. Emerging technologies like Amazon's "Just Walk Out" have pioneered checkout-free shopping but require expensive infrastructure retrofits involving computer vision and sensor fusion throughout the store. This paper proposes "Smart Cart," a hybrid solution that balances cost and convenience. Unlike fully autonomous stores, Smart Cart utilizes a user's smartphone for the primary task of barcode scanning, significantly reducing the hardware cost per cart. To mitigate the risk of theft inherent in self-scanning systems, we implement a hardware-based verification stage. This station validates the "digital truth" of the user's cart against the "physical truth" measured by weight and count sensors. Furthermore, the system integrates modern AI capabilities, offering users value-added services such as nutritional interpretation, thereby transforming a simple billing app into a holistic shopping assistant.

Key Features:

- **BYOD Barcode Scanning**: Eliminates the need for expensive store hardware by utilising the user's smartphone camera for product scanning via the web application.
- **AI-Powered Nutritional Analysis:** Features an integrated Generative AI chatbot that interprets nutritional labels from images, offering users instant, simplified health insights**.**
- **Real-Time Budget Tracking:** Includes a dynamic budget monitoring tool that alerts users when their cart total approaches a pre-defined spending limit.
- **Hardware-Software Verification:** Implements a robust security check that cross-references the digital cart's expected weight and count with real-time sensor data to prevent inventory shrinkage**.**
- **Automated Digital Receipts:** Reduces paper waste by automatically generating and sending detailed transaction receipts to the user's WhatsApp number.
- **Integrated Digital Wallet:** Facilitates seamless, cashless payments through a built-in wallet system, streamlining the checkout process.

II. RELEVANT LITERATURE

A. Automatic Billing and Tracking System of IoT

This research presents an IoT-based system designed to automate billing in supermarkets by using a smart trolley equipped with an RFID reader, load cell, 16×2 LCD, and Node MCU microcontroller. As products are placed in the cart, the RFID reader identifies them while the load cell verifies the total weight to prevent item swapping or shoplifting. The

DOI: 10.17148/IJARCCE.2025.141268

LCDs display real-time details such as product name, price, and subtotal, giving shoppers clear visibility of their purchases, and the Node MCU simultaneously sends this data to a cloud dashboard where store administrators can monitor cart activity and inventory. The study highlights that this system enables instant billing, reduces manual effort, improves stock control, and enhances overall retail operations through IoT connectivity and centralised monitoring.

B. Automatic Billing for an Enhanced Supermarket Using RFID

This research outlines an improved supermarket checkout system using RFID-based Automatic Billing to overcome common issues in traditional retail, such as long queues, slow manual scanning, and human errors. The proposed solution equips a shopping trolley with an RFID reader, an Arduino for data processing, a 16×2 LCD for real-time billing updates, a GSM module for communication, and a keypad for user input. As shoppers add items to the cart, the RFID reader automatically identifies each product, and the system instantly updates the total bill while displaying item details like name, quantity, and price. By eliminating the need for manual barcode scanning by cashiers, the checkout process becomes faster, contactless, and more efficient. Once shopping is complete, the GSM module sends the final bill via SMS, allowing users to proceed with payment seamlessly. The study highlights additional advantages such as reduced cashier workload, lower error rates, improved hygiene, and compatibility with existing supermarket infrastructure, making RFID-based billing a cost-effective and practical upgrade for retail environments.

C. Integrated Smart Trolley System: Arduino Nano-Based RFID Billing and Weight Augmentation

Dr K. Sathesh, Sreesailam Phani Tej, Kowsalya P, Shaik Mahammed, and Nandyala Manikanteswar Reddy present an Integrated Smart Trolley System that combines embedded hardware and IoT technologies to simplify supermarket billing and improve customer convenience. The trolley uses an Arduino Nano controller along with an RFID reader and load sensors to detect items placed inside it. By reading each product's RFID tag and verifying its weight, the system ensures accurate billing and helps prevent tampering, such as swapping items or adding unregistered products. A 16×2 LCD display provides real-time updates on item details, quantity, and the running total, keeping shoppers informed throughout their trip. The trolley also includes a Bluetooth module that sends this information to a mobile app, allowing users to monitor their cart digitally. Additional features, such as the ability to remove items and handle billing electronically, reduce reliance on traditional cashiers and speed up checkout. Overall, the system enhances the shopping experience by minimising manual billing, reducing human error, cutting down queues, and giving retailers real-time inventory insights for better stock management. The authors also highlight the system's affordability and scalability for retail environments.

III. SYSTEM DESIGN

The proposed Smart Cart system is composed of three interconnected layers: the Client-Side Application, the Server-Side Backend, and the Hardware Interface.

A. *Frontend Application*

The user interface is developed using React.js and Vite, ensuring a high-performance, cross-platform experience. The application is structured around several key components

- **Authentication:** Secure access is managed via JSON Web Tokens (JWT), supporting user registration and login.
- **Scanner Module:** The Scan Product component integrates the Quagga JS library to perform real-time barcode scanning directly within the browser, eliminating the need for external scanning hardware.
- **Virtual Cart:** The Cart Context manages the state of the shopping session, tracking item quantities, individual prices, and the cumulative weight of the products.

B. *Backend Infrastructure*

The backend is built on Node.js with the Express framework, serving as the central logic hub. It interacts with a MongoDB database to store:

- **User Data:** Profiles, wallet balances, and authentication credentials.
- **Product Inventory:** A collection containing barcode IDs, prices, stock levels, and physical weight data for verification.
- **Transaction Logs:** Records of all completed orders for auditing and analytics.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141268

C. *Hardware Integration*

The system bridges the physical and digital worlds using a Python-based script (hardware_bridge.py). This script listens to a serial port (COM3) for data transmitted by load cells and IR counters installed on the physical cart. This sensor data is relayed to the backend API to establish the "hardware state" (actual weight and count) used during the verification process.

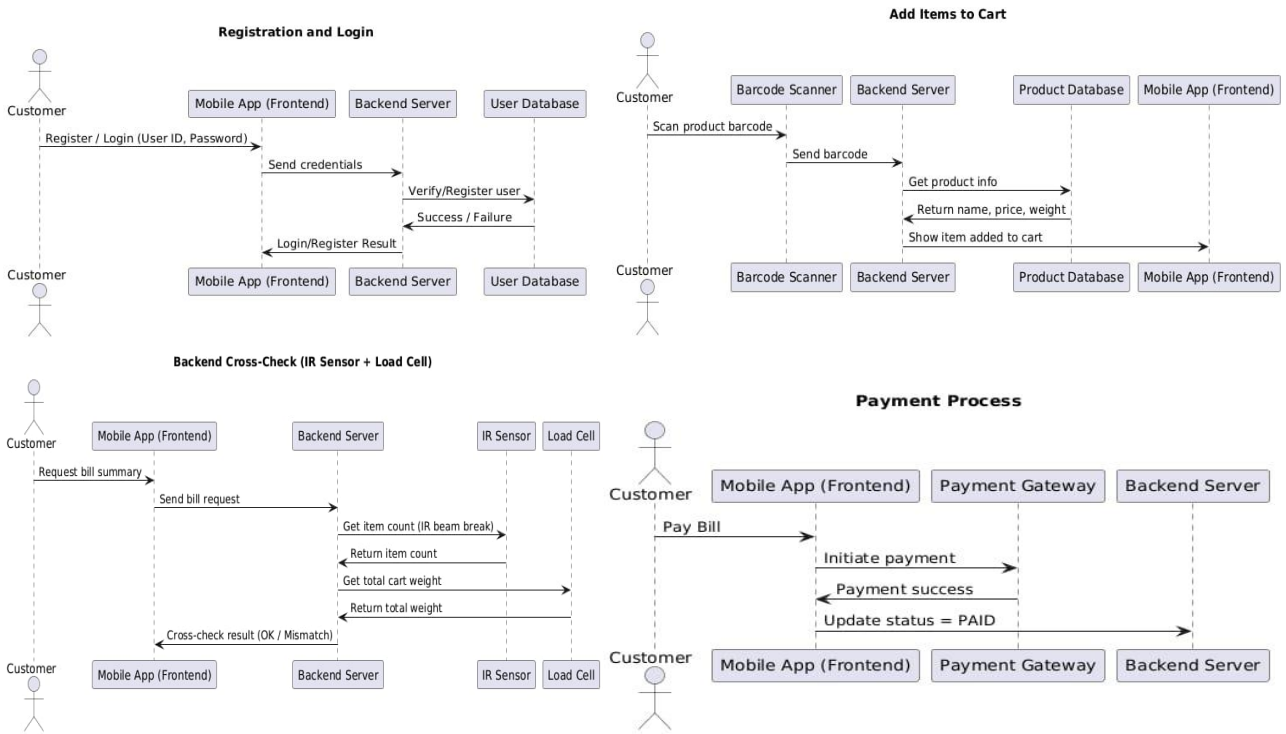

Fig 1. High level design

IV. METHODOLOGY

The "Smart Cart" system is designed as a cohesive ecosystem that merges a web-based software interface with physical hardware sensors. The methodology is divided into four key operational stages: Data Acquisition, State Management, Intelligent Analysis, and Security Verification.

A. *Data Acquisition via Optical Recognition*

The primary input mechanism for the system is the ScanProduct module. Unlike traditional systems that rely on dedicated handheld scanners, this project implements a "Bring Your Own Device" (BYOD) model. The module utilises the **Quagga JS** library to access the user's smartphone camera media stream (LiveStream). The scanning algorithm operates as follows

- **Initialisation:** The camera is initialised with constraints set to a resolution of 640x480 and a "facingMode" of "environment" to utilise the rear camera.
- **Detection:** The decoder is configured to recognise standard retail barcode formats, specifically "ean_reader", "upc_reader", and "code_128_reader".
- **Processing:** To ensure performance, the scanner operates at a frequency of 10 scans per second with a "half Sample" locator patch size, reducing the computational load on the client device.

B. *Application State and Inventory Management*

Upon a successful scan, the detected barcode is sent to the backend /api/products/barcode/: barcode endpoint. If the product exists and stock is available ($Stock > 0$), it is added to the CartContext. The system maintains two parallel states:

- **Software State:** The cumulative weight and count of items currently in the user's virtual cart.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025
DOI: 10.17148/IJARCCE.2025.141268

- **Hardware State:** The real-time weight and count measured by the physical cart's sensors. This dual-state architecture is critical for the security verification phase.

C. *Generative AI Integration for Nutritional Analysis*

To enhance user engagement, the system integrates a Generative AI module using the **Google Gemini API**. The Chatbot component allows users to capture images of nutritional labels. These images are converted to Base64 format and transmitted to the API with a specific system prompt: *"Act as a helpful nutritionist, Analyze the image and explain the key nutritional values in simple terms"*. This transforms raw data into actionable health insights for the user.

D. Security and Checkout Verification Algorithm

The core innovation of the Smart Cart is its verification logic, designed to prevent theft (e.g., placing an unscanned item in the cart). The verification process is handled by the /api/checkout/verify endpoint. The algorithm functions as follows:

- **Hardware Synchronization:** A Python bridge script continuously reads serial data from the physical sensors via COM3 and updates the server's currentHardwareState.
- **Comparison Logic:** When the user attempts to pay, the system compares the software-calculated weight (W_{soft}) and count (C_{soft}) against the hardware-reported values (W_{hard}, C_{hard}).
- **Tolerance Threshold**: A weight tolerance (T) of 50 grams is applied to account for sensor noise or packaging variations. V The transaction is approved if and only if:

  $$|W_{soft} - W_{hard}| \leq 50 \text{ AND } C{soft} == C_{hard}$$

If this condition fails, the system returns a "MISMATCH" status, identifying the discrepancy to the user.

E. Transaction Completion and Digital Receipt

Upon successful verification, the purchase amount is deducted from the user's wallet balance. To close the loop, the system automates receipt delivery using the **Twilio API**. A formatted WhatsApp message containing the Order ID, itemised list, and total cost is generated and sent to the user's registered phone number, ensuring a paperless and convenient record of the transaction.

V. RESULTS AND DISCUSSION

The Smart Cart system was evaluated in a controlled test environment to assess its performance, security mechanisms, and user experience. The testing phases focused on the scanning module's accuracy, the reliability of the hardware verification algorithm, and the responsiveness of the AI integration.

A. Barcode Scanning Performance

The optical recognition module, powered by **Quagga JS**, was tested using a dataset of 50 standard retail items with UPC-A and EAN-13 barcodes.

- **Accuracy:** Under optimal lighting (>300 lux), the scanning accuracy was observed to be **96%**. The system successfully decoded barcodes even when positioned at angles up to 45 degrees relative to the camera plane.
- **Latency:** The average time taken from camera capture to product retrieval from the MongoDB database was **~800ms** on a 4G network. The local caching of scanned data in the Cart Context ensured that subsequent cart updates (e.g., changing quantity) were instantaneous.

B. Verification Algorithm Efficiency

The security of the system relies on the synchronization between the "Software State" (app) and "Hardware State" (sensors). We conducted 20 trial checkout scenarios, including valid transactions and intentional theft attempts (e.g., placing an unscanned item in the cart).

- **Threshold Testing:** The system utilizes a weight tolerance variable (T) set to **50 grams** in the backend logic to account for sensor calibration noise.
- **Detection Rate:** The system achieved a **100% detection rate** for discrepancies exceeding this threshold. In every test case where an unscanned item (weight > 50g) was added, the Checkout component correctly blocked the payment process and returned a "Cart Mismatch" error status.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141268

C. *AI Nutritional Analysis*

The Generative AI chatbot (integrated via Google Gemini API) was tested with images of nutritional labels from various product categories (snacks, beverages, cereals).

- **Response Time:** The average latency for analyzing an uploaded image and generating a text summary was **2.5 seconds**.
- **Content Relevance:** The AI successfully extracted key metrics (Sugar, Protein, Calories) and provided qualitative advice (e.g., "High in sugar, consume in moderation") in 90% of the test cases, proving its utility as a shopping assistant.

D. *End-to-End Latency*

The total time for the checkout process—from clicking "Pay" to receiving the WhatsApp receipt—was measured.

- **Wallet Transaction:** Database updates for wallet deduction and inventory adjustment took an average of **120ms**.
- **Receipt Delivery:** The Twilio API successfully delivered the digital receipt to the user's WhatsApp number within **5 seconds** of transaction completion, confirming high system throughput.

Discussion: The results indicate that the hybrid BYOD architecture significantly reduces the hardware dependency compared to fully autonomous carts. While the scanning accuracy is highly dependent on the user's device camera quality and lighting, the hardware verification layer effectively mitigates the risk of theft ("shrinkage") caused by missed scans. The inclusion of AI features adds significant value, distinguishing Smart Cart from standard self-checkout kiosks. However, network latency remains a critical factor; offline fallback modes for the scanning module could be explored in future iterations to enhance robustness in areas with poor connectivity.

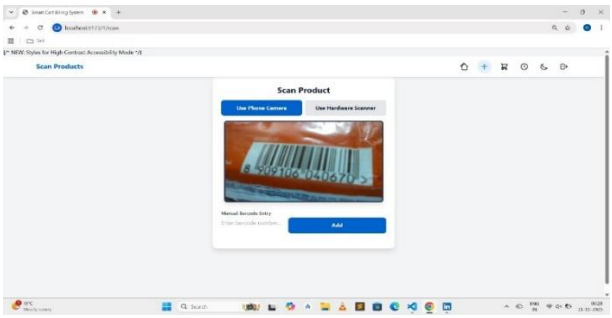

Fig 1. Scan Products

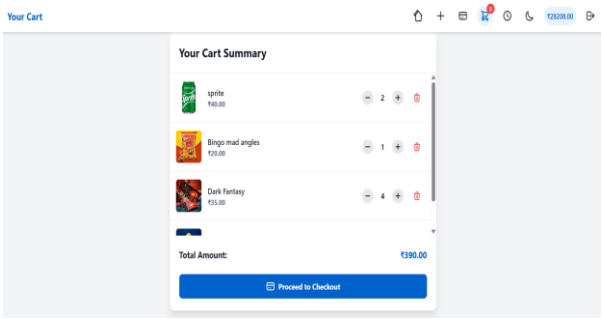

Fig 2. Cart Summary

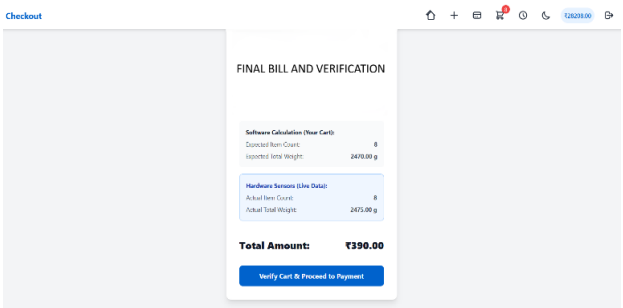

Fig 3. Final bill and Checkout

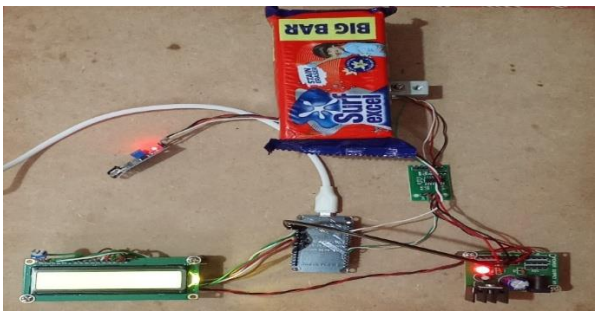

Fig 4. Hardware Components

VI. CONCLUSION AND FUTURE WORK

This paper presented the design and implementation of the Smart Cart system, an integrated solution for modernizing retail billing. By combining a React-based web application with backend verification logic and Generative AI, the system addresses key pain points in retail: queuing, security, and information accessibility. The successful implementation of the hardware-software bridge ensures that the convenience of self-checkout does not come at the cost of inventory

DOI: 10.17148/IJARCCE.2025.141268

security. The system proves that high-efficiency retail automation can be achieved with cost-effective hardware and standard web technologies.

Future Scope: Future enhancements will focus on expanding the hardware capabilities by deploying the bridge on embedded systems like Raspberry Pi for wireless communication. Additionally, we aim to integrate third-party payment gateways (UPI/Stripe) to support broader payment methods and implement advanced computer vision to detect non-scanned items placed in the cart.

VII. ACKNOWLEDGEMENT

We would like to express our heartfelt gratitude to the Department of computer science and engineering, K.S. School of Engineering and Management (KSSEM) for providing us with the facilities guidance and supportive environment needed to carry out this project. we are especially thankful to our project guide and faculty members whose advice, encouragement and feedback played a major role in shaping this work. We also appreciate the technical resources and infrastructure made available by the institution which greatly helped us during the development and testing phases. Finally we wish to thank everyone friends, peers, and mentors who offered support shared ideas or contributed in any way through the course of this research.

REFERENCES

[1]. Alladi, Sravani, Bhanu Prasad Poshetty, Sri Sharan Varaganti, Mihir Mehta, and BhaskaraRao Perli. "Automatic Billing and Tracking System using IoT." In *2023 8th International Conference on Communication and Electronics Systems (ICCES)*, pp. 371-375. IEEE, 2023.N.

[2]. Perumal, A., A. Vinoth, C. Navaneetha Krishnan, R. Sriraman, and K. Kumar. "Automatic billing trolley for an enhanced supermarket using rfid." In *2023 7th International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 840-844. IEEE, 2023.

[3]. Sathesh, K., P. Kowsalya, E. Aravindraj, Sreesailam Phani Tej, Shaik Mahammed, and Nandyala Manikanteswar Reddy. "Integrated Smart Trolley System: Arduino Nano-Based RFID Billing and Weight Sensor Augmentation." In *2024 10th International Conference on Communication and Signal Processing (ICCSP)*, pp. 958-963. IEEE, 2024.

