Global Journal of Engineering and Technology Advances
eISSN: 2582-5003
Cross Ref DOI: 10.30574/gjeta
Journal homepage: https://gjeta.com/

(RESEARCH ARTICLE)

Check for updates

Smart scan and go shopping system using flask and QR/Barcode Technology

Veeresh J D [1], Kumar Siddamallappa U [1] and Anusha Jajur. J [2,*]

[1] *Department of Studies in Computer Applications (MCA), Davangere University, Davangere, Karnataka, India*
[2] *Department of Studies in Computer Science, Davangere University, Davangere, Karnataka, India.*
Kumar Siddamallappa U; ORCID: 0000-0002-1975-3868
Anusha Jajur. J; ORCID: 0009-0003-9835-624X

Global Journal of Engineering and Technology Advances, 2026, 28(02), 039–046

Publication history: Received on 26 June 2026; revised on 02 August 2026; accepted on 04 August 2026

Article DOI: https://doi.org/10.30574/gjeta.2026.28.2.0191

Abstract

The retail sector continues to face persistent challenges related to billing delays, long checkout queues, and inefficient customer handling during peak shopping periods. Traditional billing depends heavily on cashier-operated counters, where each product must be scanned and processed sequentially, often creating congestion and reducing customer satisfaction. This paper presents ScanandGo, a smart self-checkout shopping system that combines a Flask-based backend, a Flutter mobile application, a web-based administrative portal, MySQL database management, QR code product identification, and secure exit verification. The proposed system enables customers to scan product QR codes while shopping, add products to a virtual cart, monitor the bill in real time, and complete checkout using online or cash-based payment workflows. After successful payment, the system generates a digital receipt encoded as a QR token, which is later verified at the store exit. The system is intended to reduce billing time, improve transaction accuracy, lower cashier dependency, and modernize the shopping experience in a cost-effective way. The paper discusses the motivation, objectives, literature background, system architecture, methodology, database design, implementation modules, expected results, limitations, and future scope of the proposed model.

Keywords: Scan and Go; Smart Shopping; QR Code; Mobile Self-Checkout.

1 Introduction

Retail automation has become increasingly important due to changing customer expectations and the need for faster service in supermarkets and stores. Customers no longer view shopping as only the process of selecting goods; they also expect fast checkout, reduced waiting time, transparent billing, and convenient payment options. However, many retail environments still rely on conventional billing counters that become bottlenecks when customer volume increases, especially during holidays, weekends, and discount campaigns. As a result, store operations slow down, staff workload increases, and the shopping experience becomes less efficient.

Digital self-checkout systems offer a practical way to address these challenges by shifting part of the transaction workflow from the cashier to the customer. By allowing product identification, cart management, and billing to occur through mobile or web systems, retailers can improve throughput without significantly increasing the number of counters or staff. QR code and barcode technologies are especially suitable for this purpose because they are simple, low-cost, and widely compatible with smartphone cameras and web-connected devices.

Image processing is the use of computational techniques to analyze, enhance, and extract useful information from images. In general, it takes an image as input and produces either an improved image or meaningful data about that image. In the context of the ScanandGo system, the mobile camera captures the QR code image, and the system detects

* Corresponding author: Veeresh. J. D

and decodes the encoded data to identify the product or transaction details. This process is related to image-based analysis, but it is more specifically called QR code detection and decoding rather than full image processing.

The proposed ScanandGo system is designed as a smart shopping platform that supports end-to-end self-checkout with customer-side scanning, server-side bill generation, and exit-side receipt verification. It uses Flask for API and business logic, Flutter for the customer mobile app, HTML/CSS/JavaScript for the admin dashboard, and MySQL for centralized data management. Through this integration, the system aims to provide a scalable retail solution that is practical for small and medium stores while also supporting future upgrades such as AI-based recommendations, barcode expansion, and smart anti-theft measures.

2 Literature Review

The evolution of smart retail systems is closely associated with the broader growth of automation in commerce and service environments. Several studies in the domain of smart shopping emphasize the importance of faster billing, reduced congestion, and improved customer convenience through automated checkout processes.

Google [1] introduced Flutter as a cross-platform application development framework that enables developers to build high-performance mobile applications for Android and iOS using a single codebase. D. Kato and H. Tanaka [2] developed a QR code-based mobile shopping system that simplified product identification and enabled faster checkout processes in retail environments. S. Kumar and M. Ramesh [3] designed an IoT-enabled smart shopping cart that automated billing operations and minimized customer waiting time in supermarkets. A. Patel and R. Shah [4] proposed a mobile self-checkout system that reduced dependency on cashiers and enhanced shopping efficiency through smartphone-based billing. M. Grinberg [5] explained the Flask web framework and demonstrated its suitability for developing lightweight, secure, and scalable web applications. International Organization for Standardization [6] established the ISO/IEC 18004 standard for QR code generation and decoding, ensuring reliable data encoding and interoperability across systems. S. Preradovic and N. C. Karmakar [7] presented a comprehensive survey of RFID technology and discussed its applications in retail automation, inventory management, and supply chain systems. K. Finkenzeller [8] described the fundamentals and practical applications of RFID technology for product identification and retail automation. M. Weiser [9] introduced the concept of ubiquitous computing, providing the foundation for intelligent and technology-enabled retail environments. E. Gamma et al. [10] introduced software design patterns that improve software maintainability, modularity, and code reusability in application development. M. Fowler [11] presented enterprise application architecture patterns that support scalable and maintainable software system design. P. Barry and D. Griffiths [12] explained Python programming concepts and development techniques that are widely used for backend application development. A. Hunt and D. Thomas [13] discussed software engineering best practices that improve software quality, maintainability, and project success. Oracle Corporation [14] described MySQL database management features for efficient data storage, retrieval, and transaction processing in web applications. Google Developers [15] introduced ML Kit Barcode Scanning technology for accurate and real-time barcode and QR code recognition in mobile applications. A. S. Tanenbaum and H. Bos [16] explained modern operating system concepts that support efficient execution of mobile and distributed software applications. R. Pressman and B. Maxim [17] presented software engineering methodologies covering software design, development, testing, and maintenance processes. I. Sommerville [18] discussed software engineering principles that support the development of reliable, scalable, and user-friendly software systems. M. J. Hernandez [19] explained database design principles that ensure data integrity, consistency, and efficient information management. G. Booch, J. Rumbaugh, and I. Jacobson [20] introduced UML as a standardized modeling language for designing, documenting, and visualizing software systems. World Wide Web Consortium [21] provided web architecture standards that improve interoperability, accessibility, and security in web application development. P. Sadalage and M. Fowler [22] discussed NoSQL database technologies and their advantages in handling scalable and high-performance data management. D. K. Panda and S. Mishra [23] proposed a QR code-based smart shopping system that enhanced customer convenience and improved retail billing efficiency. J. Nielsen [24] introduced usability engineering principles that enhance user interface design and improve overall user experience in software applications.

3 Proposed System

ScanandGo is designed as an integrated retail checkout platform that shifts product billing from the traditional cashier desk to a distributed self-checkout process. In the proposed model, each product is linked to a QR code that can be scanned by the customer using the Flutter mobile application. The mobile app reads the product identifier, sends the request to the Flask backend, and retrieves product information such as name, category, and price from the MySQL database. Once identified, the item is added to the customer's digital cart, where quantity and total value can be managed in real time.

At the completion of shopping, the customer chooses either online payment or cash payment. In the online mode, the system connects to Razorpay and confirms payment digitally, while in the cash mode, the order is placed in a pending state until approved through the admin cash counter interface. Once the payment is validated, the backend generates an order record and provides a QR-based receipt in the format ORDER:<id>. This receipt is later scanned by a security module at the exit gate to confirm whether the order is paid and eligible for clearance. The proposed architecture therefore combines convenience with control. Customers benefit from queue-free or reduced-queue shopping, and retailers gain administrative oversight through centralized product management, order monitoring, payment handling, and security verification. This makes the system both customer-oriented and operationally useful for store management.

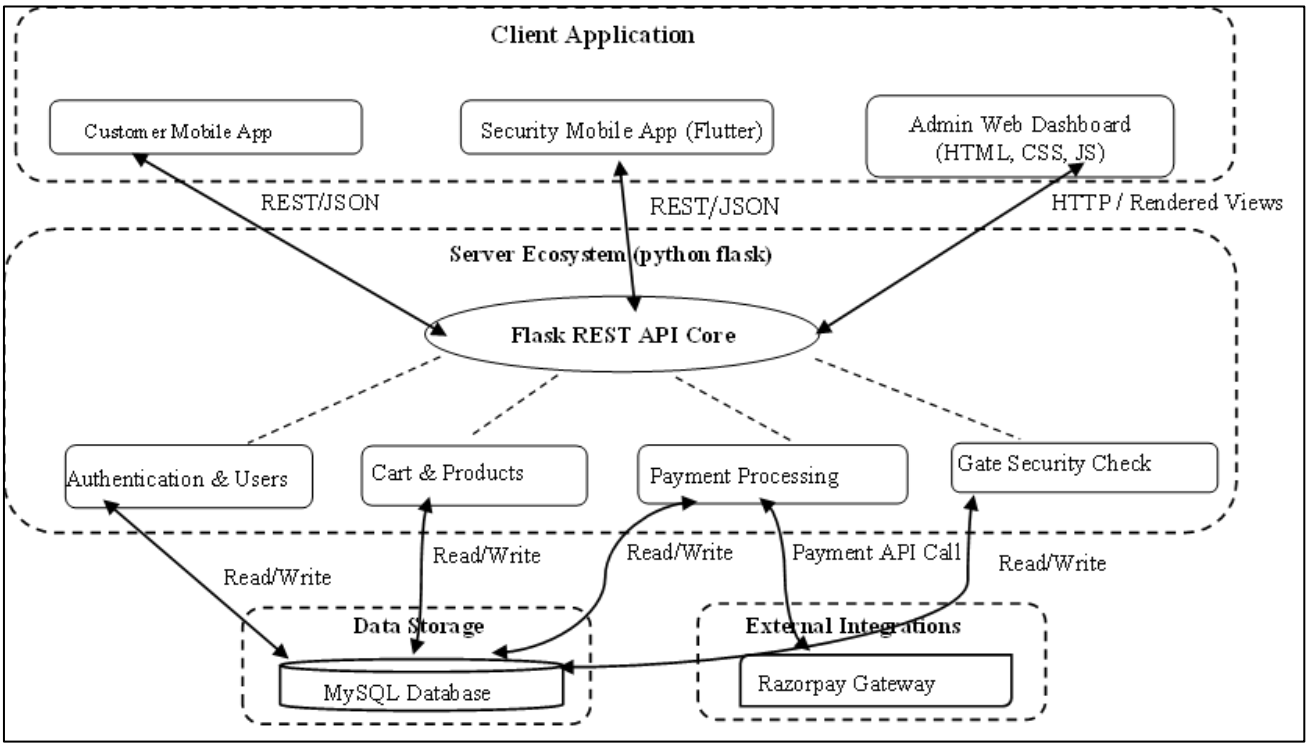

Figure 1 System Architecture

The overall architecture of the proposed ScanandGo Smart Shopping System follows a three-tier architecture consisting of the Client Layer, Server Layer, and Data and External Integration Layer. The Client Layer includes three independent applications: the Flutter Customer Mobile Application, Flutter Security Mobile Application, and the Admin Web Dashboard. The Customer Application enables users to authenticate, scan product QR codes, manage their shopping cart, complete online or cash payments, and receive digital receipts with Exit QR codes. The Security Application is designed exclusively for security personnel to verify customer Exit QR codes before permitting store exit. The Admin Web Dashboard, developed using HTML, CSS, JavaScript, and Jinja2 templates, allows administrators to manage categories, products, inventory, customer orders, cash transactions, and system operations. Communication between these client applications and the backend is achieved through RESTful APIs using HTTP and JSON, while the Admin Dashboard communicates directly with the Flask server using server-side rendered web pages.

The Server Layer is implemented using the Python Flask framework, which acts as the central controller of the entire system. The Flask REST API Core manages authentication, product retrieval, shopping cart operations, payment processing, inventory management, and exit verification. Dedicated backend modules such as Authentication and Users, Cart and Products, Payment Processing, and Gate Security Check execute the business logic and interact with the MySQL relational database using the mysql-connector-python library for persistent data storage. The backend also integrates with the Razorpay Payment Gateway to generate secure payment orders and verify online transactions before updating the payment status. Since all system modules communicate through centralized REST APIs while maintaining independent functionality, the architecture ensures loose coupling, improved scalability, simplified maintenance, secure data management, and efficient coordination among all components of the ScanandGo ecosystem.

4 Research Methodology

The project follows a systematic software development methodology consisting of analysis, design, implementation, testing, and deployment. In the analysis phase, the needs of customers, store administrators, and security personnel are studied to identify the essential functions of a smart self-checkout environment. This includes studying queue-related delays, billing inefficiencies, security concerns, and the need for real-time product retrieval. During the design phase, the project architecture, API routes, interface behavior, module interactions, and database schema are planned. Particular attention is given to the flow of information between the mobile application, Flask APIs, MySQL data storage, and the security scanning module. Design also includes the generation of product QR codes and digital receipt QR codes for downstream verification.

In the implementation phase, the backend is developed using Python Flask, and the database is connected through MySQL integration. The customer-facing application is created in Flutter to support mobile interaction, while the admin dashboard is developed using HTML, CSS, and JavaScript templates. This phase also includes cart API development, payment workflow integration, and stock deduction logic. Testing is carried out through multiple methods, including backend API validation, functional testing, and integration testing. Functional tests verify whether users can log in, scan products, modify the cart, complete payment, and pass exit verification correctly. Integration tests examine the interaction between payment services, mobile devices, cart logic, and order validation. The final deployment stage makes the application suitable for controlled practical use in a retail environment.

4.1 Methods and Module Description

- **User Authentication Module:** This module provides registration, login, and user verification functions. It supports secure access and helps associate cart and order activity with authenticated users.
- **Product Management Module:** This module allows administrators to manage product information including identifiers, category mapping, pricing, descriptions, stock quantity, and QR code assignment. It is essential for maintaining an up-to-date and searchable product catalog.
- **QR Scanning Module:** This module reads product identifiers through the mobile camera and fetches associated details from the backend. It is the primary mechanism through which the customer interacts with the shopping system while moving through the store.
- **Cart Management Module:** This module enables customers to add products, update quantities, remove unwanted items, and review their selections before payment. Server-side cart storage helps maintain reliability and supports accurate price calculation.
- **Billing and Checkout Module:** This module calculates subtotal, GST, discounts, and final payable amount. It also handles payment method selection and creates final orders once the checkout process is complete.
- **Security Verification Module:** This module verifies whether a paid order corresponds to a valid exit receipt. It strengthens the retail workflow by introducing a final control point before the customer leaves the store.
- **Report and Monitoring Module:** This module helps store management analyze order activity, product movement, and operational trends through administrative reporting and dashboard summaries. It supports better oversight and decision-making.
- **Billing Logic**: Billing is handled on the server side to maintain consistency and reduce the risk of client-side manipulation. For each cart item, the system multiplies the unit price by the selected quantity to determine line totals. The subtotal is calculated as the sum of all line totals, after which GST is applied at 5 percent. A generalized billing equation used in the system is shown below:

$$\text{Grand Total} = \text{Subtotal} + (\text{Subtotal} \times 0.05) - \text{Discount Amount}$$

This structure provides clarity, supports transparent billing, and makes it easy to introduce later additions such as promotional discounts or coupon codes. Once payment is confirmed, the order details are inserted into the database, and a digital receipt QR is generated for post-payment verification.

5 Results

Table 1 Testing and Results

Aspect	Details
Purpose of Testing	Testing is essential to validate whether the ScanandGo system meets its functional and operational goals.
Backend API Testing	Used to verify product lookup, cart additions, updates, removals, billing calculations, stock adjustment, and order generation.
Functional Testing	Performed across the customer app, admin workflow, and security verification process to ensure that each component behaves correctly under expected usage conditions.
Integration Testing	Important because the system depends on coordination among multiple layers including mobile scanning, database synchronization, payment workflow handling, and QR-based receipt verification.
Payment Testing	Confirms whether online transactions are processed successfully, and cash orders are traceable through the pending-to-approved state transition.
Exit Validation Testing	Ensures the system correctly distinguishes between valid, unpaid, already-used, and invalid receipt conditions.
Expected Result	Improved billing speed and better customer convenience because much of the checkout activity occurs during the shopping journey rather than at a fixed billing counter.
Retailer Benefits	Better order visibility, real-time stock updates, and reduced cashier dependency.
Cash Payment Impact	Even when cash payments are involved, the structure improves order organization and creates a more modern and traceable shopping process.
Future Improvements	Loyalty and rewards systems, smart anti-theft mechanisms using weight sensors, real-time analytics dashboards, integrated promotional campaigns, and advanced inventory intelligence.
Overall Outcome	The platform could evolve from a self-checkout system into a more comprehensive smart retail management ecosystem.

5.1 Screenshots

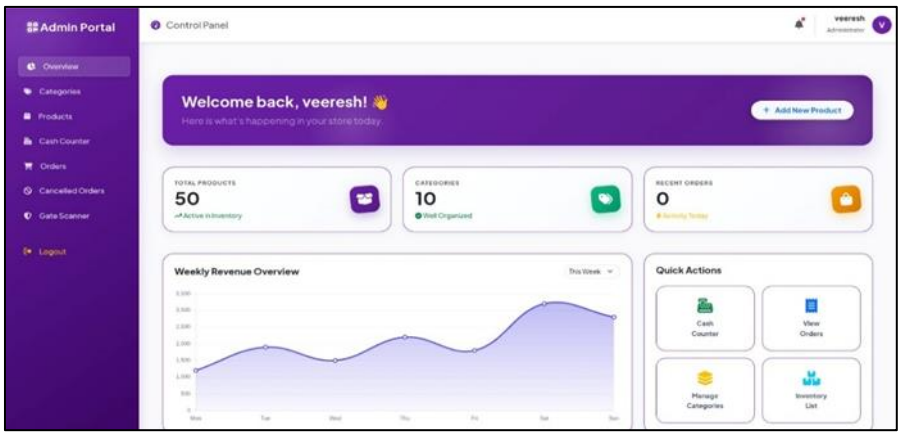

Figure 2 Admin Dashboard

Figure 3 Manage Categories

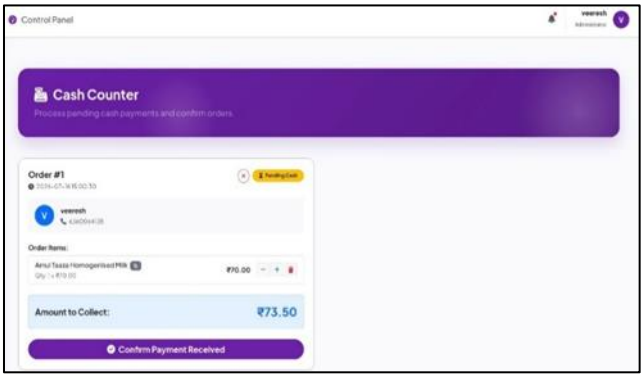

Figure 4 Cash Counter

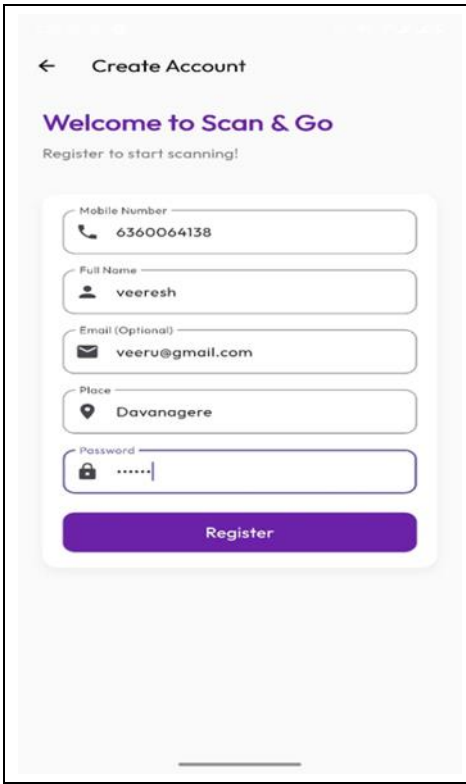

Figure 5 User Registration

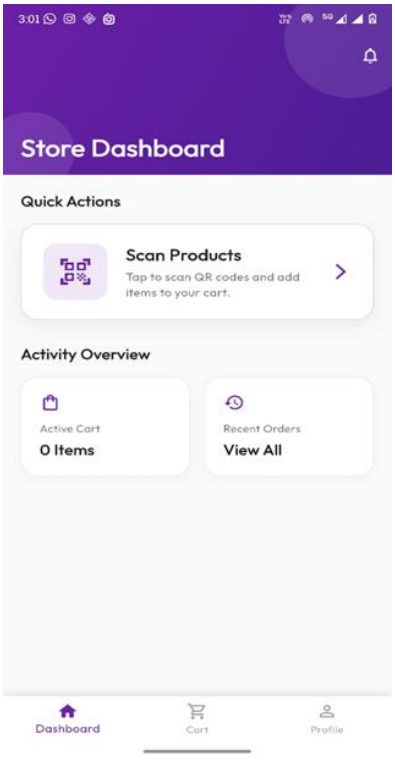

Figure 6 User Home Screen

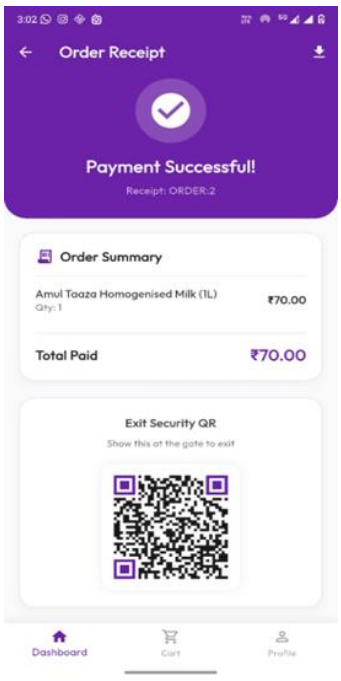

Figure 7 User Generated bill

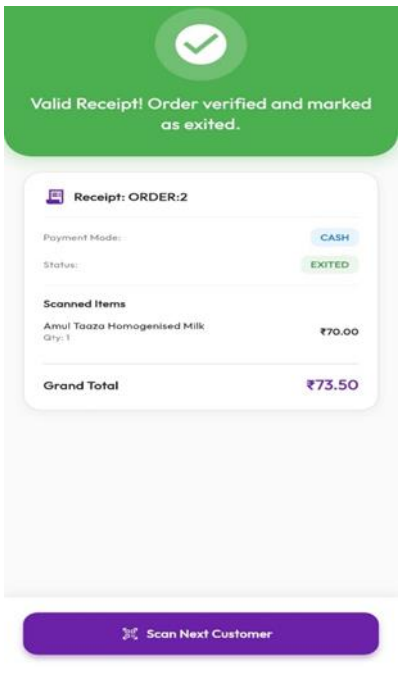

Figure 8 Security Scan Success

6 Conclusion

The Smart Scan and Go Shopping System provides a practical and technology-driven solution to one of the major challenges faced by retail stores—long billing queues and checkout delays. By integrating a Flask-based backend, Flutter mobile application, centralized database, QR code product scanning, digital cart management, and secure exit verification, the system enables customers to complete most of the checkout process while shopping. This distributed self-checkout approach significantly reduces waiting time, improves transaction efficiency, and enhances the overall shopping experience. The software-oriented architecture makes the system a cost-effective alternative to RFID- or smart cart-based solutions, making it particularly suitable for small and medium-sized retail stores. Its modular design, which separates customer, administrator, and security functions, along with support for both online and cash payments, provides a scalable and well-organized framework for efficient retail management.

A key strength of the proposed system is its integration of post-purchase validation through QR code-based digital receipts and exit-side verification, ensuring secure item clearance and better transaction accountability. Unlike many basic self-checkout systems that end with payment confirmation, this solution supports complete order tracking, approval management, and controlled store exit, reducing dependence on manual cashier intervention while improving operational transparency. Furthermore, the project demonstrates how lightweight web and mobile technologies can effectively automate retail operations without requiring expensive hardware infrastructure. With future enhancements such as support for multiple barcode formats, AI-based product recognition, customer loyalty programs, advanced analytics, and cloud-based inventory integration, the system has strong potential to evolve into a comprehensive next-generation retail automation platform suitable for real-world deployment.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] Google, Flutter: Build Apps for Any Screen, 2024. [Online]. Available: https://flutter.dev. [Accessed: Jul. 20, 2026].

[2] D. Kato and H. Tanaka, "QR Code-Based Mobile Shopping System for Retail Applications," International Journal of Computer Applications, vol. 177, no. 12, pp. 20–26, 2020.

[3] S. Kumar and M. Ramesh, "Smart Shopping Cart Using IoT Technology," International Journal of Innovative Research in Science, Engineering and Technology, vol. 8, no. 5, pp. 5500–5506, 2019.

[4] A. Patel and R. Shah, "Mobile Self-Checkout System for Smart Retail Stores," International Journal of Advanced Research in Computer Science, vol. 10, no. 4, pp. 55–61, 2019.

[5] M. Grinberg, Flask Web Development: Developing Web Applications with Python, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2018.

[6] International Organization for Standardization, ISO/IEC 18004:2015—Information Technology—Automatic Identification and Data Capture Techniques—QR Code Bar Code Symbology Specification. Geneva, Switzerland: ISO, 2015.

[7] S. Preradovic and N. C. Karmakar, "RFID Technology for Retail Automation: A Survey," IEEE Transactions on Automation Science and Engineering, vol. 9, no. 3, pp. 561–572, Jul. 2012.

[8] K. Finkenzeller, RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification, 3rd ed. Hoboken, NJ, USA: Wiley, 2010.

[9] M. Weiser, "The Computer for the 21st Century," Scientific American, vol. 265, no. 3, pp. 94–104, Sep. 1991.

[10] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Boston, MA, USA: Addison-Wesley, 1995.

[11] M. Fowler, Patterns of Enterprise Application Architecture. Boston, MA, USA: Addison-Wesley, 2003.

[12] P. Barry and D. Griffiths, Head First Python, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2016.

[13] A. Hunt and D. Thomas, The Pragmatic Programmer: Your Journey to Mastery, 2nd ed. Boston, MA, USA: Addison-Wesley, 2019.

[14] Oracle Corporation, MySQL 8.0 Reference Manual, 2023. [Online]. Available: https://dev.mysql.com/doc/. [Accessed: Jul. 20, 2026].

[15] Google Developers, ML Kit Barcode Scanning Documentation, 2024. [Online]. Available: https://developers.google.com/ml-kit. [Accessed: Jul. 20, 2026].

[16] A. S. Tanenbaum and H. Bos, Modern Operating Systems, 5th ed. Boston, MA, USA: Pearson, 2022.

[17] R. Pressman and B. Maxim, Software Engineering: A Practitioner's Approach, 9th ed. New York, NY, USA: McGraw-Hill, 2019.

[18] I. Sommerville, Software Engineering, 10th ed. Boston, MA, USA: Pearson, 2016.

[19] M. J. Hernandez, Database Design for Mere Mortals, 4th ed. Boston, MA, USA: Addison-Wesley, 2021.

[20] G. Booch, J. Rumbaugh, and I. Jacobson, The Unified Modeling Language User Guide, 2nd ed. Boston, MA, USA: Addison-Wesley, 2005.

[21] World Wide Web Consortium (W3C), Web Architecture, 2023. [Online]. Available: https://www.w3.org/. [Accessed: Jul. 20, 2026].

[22] P. Sadalage and M. Fowler, NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence. Boston, MA, USA: Addison-Wesley, 2013.

[23] D. K. Panda and S. Mishra, "Mobile-Based Smart Shopping System Using QR Code Technology," International Journal of Scientific Research in Computer Science, Engineering and Information Technology, vol. 6, no. 2, pp. 115–121, 2020.

[24] J. Nielsen, Usability Engineering. San Francisco, CA, USA: Morgan Kaufmann, 1994.