VIVA-TECH IJRI
ANNUAL RESEARCH JOURNAL
VIVA-TECH INTERNATIONAL JOURNAL FOR RESEARCH AND INNOVATION
ANNUAL RESEARCH JOURNAL
ISSN(ONLINE): 2581-7280

# Smart Shopping Cart Using Raspberry Pi and Computer Vision

Mr. Eshaan Bangera[1], Mr. Ali Firoz[2], Mr. Bhalchand Gaund[3]
Prof. Jyoti Deore[4]

*(Electronics and Telecommunications, Viva institute of technology /Mumbai University, India)*
*(Electronics and Telecommunications, Viva institute of technology /Mumbai University, India)*
*(Electronics and Telecommunications, Viva institute of technology /Mumbai University, India)*
*(Electronics and Telecommunications, Viva institute of technology /Mumbai University, India*

***Abstract :*** *The rapid advancements in computer vision and edge computing have catalyzed the development of innovative retail solutions. This research presents a smart shopping cart system leveraging Raspberry Pi and computer vision technologies to streamline the shopping experience by automating item detection and checkout processes. Using a Raspberry Pi, This system employs a camera module and machine learning algorithms to recognize products in real-time. Experimental results demonstrate the feasibility and accuracy of this system in a controlled retail environment. The implementation addresses common challenges such as lighting variability and computational constraints, paving the way for scalable and cost-effective retail automation solutions.*

***Keywords*** *- Automation, Computer Vision, Edge Computing, Retail Technology, Smart Shopping Cart*

## I. INTRODUCTION

The retail industry has long faced challenges related to inefficiencies such as lengthy checkout lines, manual inventory tracking, and high labor costs. These issues are exacerbated by increasing consumer expectations for speed and convenience, which traditional systems often fail to meet. Although self-checkout systems have been introduced to address these problems, they typically rely on barcode scanning or RFID technology. Barcode-based systems require manual intervention, while RFID solutions entail significant infrastructure costs, making them less accessible to smaller retailers.

Recent advancements in computer vision and machine learning offer promising alternatives. Systems like Amazon Go have demonstrated the potential of computer vision in creating cashier-less retail environments. However, such solutions rely on expensive hardware and extensive cloud-based processing, limiting their scalability and affordability. Edge computing devices like Raspberry Pi present an opportunity to bring computer vision-based automation to a broader market by enabling real-time processing at a lower cost.

This research introduces a smart shopping cart system that integrates computer vision, edge computing, and machine learning to automate product recognition and billing. The proposed solution eliminates the need for barcode scanning, enhances the shopping experience, and reduces operational costs.

## II. LITERATURE REVIEW

The inefficiencies in traditional retail systems have spurred extensive research into automating the shopping experience. Self-checkout systems, which aim to reduce customer wait times and operational costs, have been a

focal point of these studies. Below, we review key approaches that have been proposed and their implications for smart shopping cart systems. RFID technology has been widely adopted for inventory management and automated checkout. Research highlights several advantages:

**Efficiency and Accuracy:** RFID systems enable automated item identification without requiring a direct line of sight, reducing errors associated with manual barcode scanning. Studies such as those by Sudhakar and Chaitanya demonstrate significant improvements in inventory accuracy and tracking efficiency. [1]

**High Costs:** However, RFID tags and readers remain expensive, particularly for small retailers. The high initial investment and ongoing maintenance costs make RFID systems less accessible, as shown in studies focusing on the economic challenges of deploying RFID at scale.

**Integration Complexity:** Implementing RFID requires overhauling existing infrastructure, including backend inventory systems and point-of-sale terminals. This challenge has been extensively documented in technical reports and case studies on RFID adoption in retail environments.

The emergence of deep learning has transformed computer vision applications, enabling real-time object detection and classification. Research in this area has been driven by advances in Convolutional Neural Networks (CNNs) and their deployment in retail systems.

**Cashier-less Stores:** Amazon Go exemplifies the use of computer vision in creating seamless shopping experiences. Academic studies of this technology, such as Rajalakshmi et al., detail its reliance on overhead cameras, weight sensors, and machine learning algorithms to identify items and update virtual carts automatically. Despite its innovation, the cost and complexity of this approach limit its scalability. [2]

**Shelf Monitoring and Analytics:** Beyond checkout automation, computer vision has been applied to monitor stock levels and analyse customer behaviour. These applications, often reliant on cloud-based processing, face challenges related to latency and data security, as identified in studies by Kumar and Gupta. Edge computing offers a cost-effective alternative by processing data locally on devices such as the Raspberry Pi. This approach has gained traction due to its ability to reduce dependency on cloud infrastructure. [3]

**Real-time Processing:** Studies show that edge-based systems achieve significantly lower latency compared to cloud-based counterparts, making them ideal for applications requiring immediate feedback, such as smart shopping carts.

**Cost and Scalability:** Research by Salam et al. highlights the affordability and scalability of edge devices in retail settings. Lightweight neural networks optimized for edge hardware, using techniques such as quantization and pruning, have demonstrated near-cloud performance in terms of accuracy and speed. [4]

**Energy Efficiency:** Edge devices consume less power, further reducing operational costs. These findings align with industry reports emphasizing the environmental and economic benefits of edge computing for small and medium-sized retailers.

While RFID, computer vision, and edge computing each address specific aspects of retail automation, combining these technologies can yield comprehensive solutions. Studies by Chowdhury et al. propose hybrid systems that leverage RFID for inventory tracking and computer vision for checkout automation. These systems balance cost, efficiency, and scalability, although their real-world implementation remains a challenge. This review underscores the potential of smart shopping cart systems to revolutionize the retail experience. By addressing the limitations of existing solutions and integrating advancements in edge computing and computer vision, this research aims to develop a cost-effective, scalable alternative suitable for diverse retail environments. [5]

# III. Methodology

### 3.1 Hardware Components

The smart shopping cart system is composed of several critical hardware components designed to provide robust functionality and reliability in a retail environment.

- **Raspberry Pi 5 B**: This serves as the central processing unit, managing image capture, object detection, and communication with peripheral devices. The Raspberry Pi 5 B features a quad-core processor clocked at 2.4 GHz and 8 GB of RAM, which enables it to run lightweight machine learning models efficiently. Additionally, its GPIO pins facilitate seamless integration with sensors and actuators, making it an ideal choice for edge computing tasks.
- **Camera Module**: A Raspberry Pi-compatible 8-megapixel web camera module is used to capture high-resolution images of items placed in the cart. The camera is mounted on a retractable arm to provide a clear and unobstructed view of the cart's interior. Adjustable focus and exposure settings ensure optimal image quality under varying lighting conditions, a critical factor for accurate object detection.
- **Load Sensors**: Load sensors are installed beneath the cart's basket to measure the weight of items as they are added. These sensors provide real-time weight data, which is cross-referenced with the expected weight of detected items from the product database. This dual validation mechanism reduces errors caused by misclassification or accidental omissions.
- **Display Module**: A 7-inch capacitive touchscreen display serves as the user interface. The display is mounted on the cart handle for easy accessibility and provides real-time feedback to the user. It displays details about detected items, their prices, and the running total of the bill. Additionally, it offers interactive options to remove items or initiate the checkout process.
- **Power Supply**: A rechargeable lithium-ion battery pack powers the system. The battery pack includes a voltage regulator to ensure stable operation, with a runtime of up to 8 hours under typical usage conditions.


Fig.1 Hardware architecture

### 3.2 Software Architecture

The software architecture of the smart shopping cart integrates multiple subsystems for object detection, data management, and user interaction.

- **Object Detection**: The RNN (Recurrent Neural Network) algorithm was selected for its ability to deliver real-time object detection with high accuracy. RNN's trained weights were fine-tuned using a custom dataset to recognize a wide range of retail items. The model runs on the Raspberry Pi using TensorFlow, an optimized framework for edge devices.
- **Data Management**: A local SQL database stores product information, including item names, prices, and expected weights. The database is designed for quick lookups and efficient data retrieval, ensuring minimal latency during object detection and validation processes.
- **User Interface**: The Flask web framework was employed to develop a lightweight graphical user interface (GUI). This interface handles user inputs, displays item details, and facilitates seamless interaction with the system.
- **Integration Framework**: Python was used as the primary programming language due to its extensive library support for image processing (OpenCV), machine learning (TensorFlow), and hardware interfacing (RPi. GPIO).
- **Mobile Application** The system includes a companion mobile application for users to view their real-time cart total, access product details, and make payments. The application communicates with the cart via a secure cloud API.
- **User Authentication:** Secure login and account management.
- **Cart Sync:** Real-time synchronization with the cart's product list and total.
- **Payment Gateway:** Seamless integration with popular payment methods for a smooth checkout experience.

- **Admin Panel Website** An admin panel website provides retailers with tools to manage inventory, monitor cart usage, and analyse sales data. The admin panel is built using modern web technologies and includes:
- **Dashboard:** Visualizations of key metrics such as sales trends and active carts.
- **Product Management:** Tools to add, update, or remove products, including their images and prices.
- **User Management:** Access controls and activity logs for both customers.
- **Reports:** Detailed reports on transactions, product popularity, and revenue.

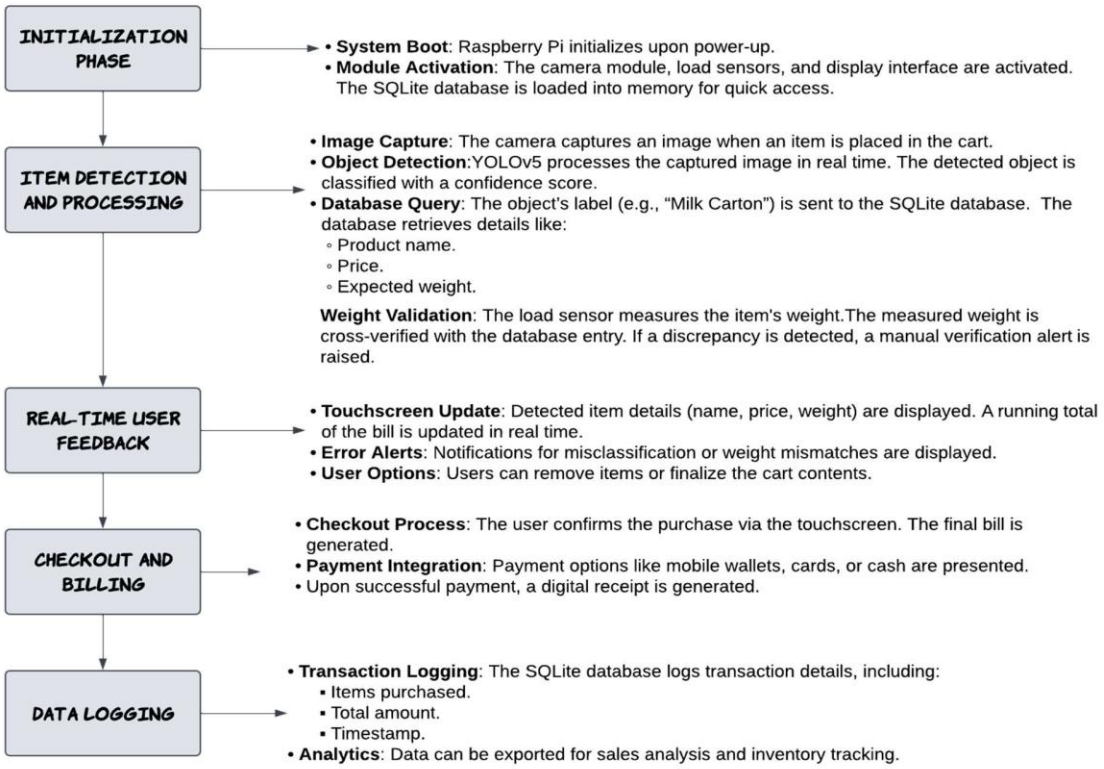


Fig.2 Software architecture

### 3.3 Workflow

1. **Item Detection**: When an item is placed in the cart, the camera captures an image. The RNN's model processes the image, identifies the object, and assigns a classification label.
2. **Database Query**: The detected object's label is used to query the SQL database. The system retrieves the item's name, price, and expected weight.
3. **Weight Verification**: The load sensors measure the weight of the added item. This weight is compared with the database's expected weight for the detected item. Discrepancies trigger an alert for manual verification, ensuring accuracy.
4. **Display Update**: Detected items and their details are displayed on the touchscreen interface, along with the updated bill total. Users can interact with the interface to review or remove items.
5. **Checkout Process**: Once shopping is complete, users can initiate checkout via the touchscreen. The system generates a detailed bill and provides payment options, including mobile wallets and card readers.

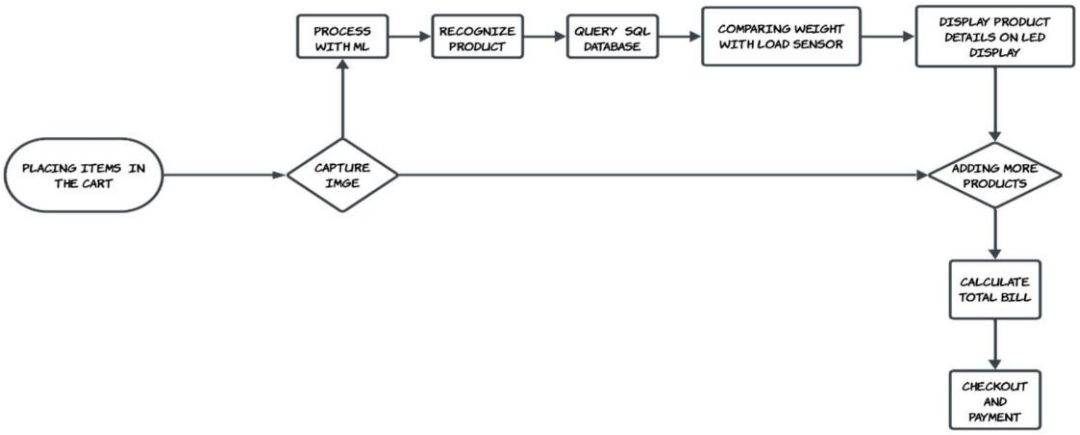


Fig.3 flowchart of Smart Shopping cart

### 3.4 Dataset Preparation

A custom dataset was created to train the RNN's model, consisting of over 1,000 images representing 10 common retail items. Images were collected under diverse lighting conditions and from multiple angles to improve the model's robustness. Data augmentation techniques such as rotation, scaling, flipping, and cropping were applied to enhance dataset diversity and reduce overfitting. Labels were annotated manually using tools like Labelling to ensure high-quality training data.

### 3.5 System Deployment

The system was deployed on a prototype shopping cart. The Raspberry Pi, Web camera module, and load sensors were securely mounted on the cart, with the touchscreen display positioned for optimal usability. The entire setup was encased in a weather-resistant housing to protect the electronics from environmental factors. During deployment, the system was subjected to extensive testing in a controlled retail simulation environment, replicating real-world conditions such as variable lighting and high shopper activity. The prototype demonstrated stable operation and high accuracy, validating the design choices and integration approach.

## IV. Results

### 4.1 Experimental Setup

A prototype cart was tested in a controlled environment simulating a retail store. The database included 10 commonly purchased items, such as packaged goods, beverages, and fresh produce. Performance metrics were evaluated based on detection accuracy, processing speed, and system reliability under various conditions.

### 4.2 Performance Metrics

**Accuracy:** The system achieved an item recognition accuracy of 94%, with higher accuracy observed for packaged goods compared to fresh produce.
**Processing Time**: The average detection and classification time per item was 1.2 seconds, allowing for near real-time operation.
**Error Rate:** Misclassification occurred in 6% of cases, primarily due to poor lighting, occlusion, or similarities between items (e.g., two different brands of beverages with similar packaging).

### 4.3 Observations

Real-time detection was seamless under optimal lighting conditions. The system's performance degraded slightly in low-light environments, necessitating the use of preprocessing techniques or additional lighting. Computational constraints of the Raspberry Pi limited the frame rate when processing high-resolution images. This issue could be mitigated by using a more powerful edge device or further optimizing the model.

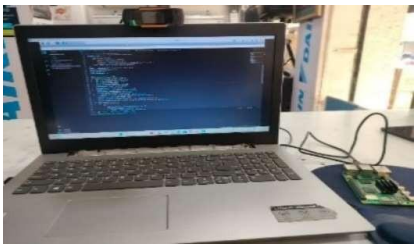

Fig.4 Runing ML model in Raspberry pi

Figure 4 illustrates the execution of a machine learning (ML) model on a Raspberry Pi for the smart shopping cart system. The setup includes:

- **Raspberry Pi**: Positioned on the right, serving as the edge device responsible for running lightweight ML models to detect and classify items in real-time.
- **Camera Module**: Attached to capture images of items for processing by the ML model.


Fig 5. Working of ML model


Fig 6. Recognizing the product using ML model

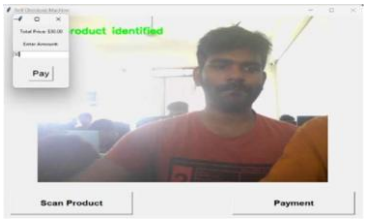


Fig 7. Payment of the product

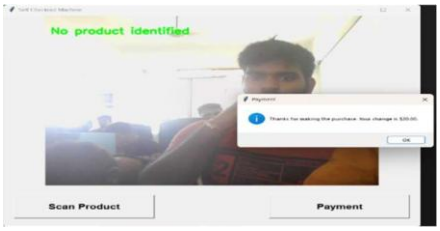


Fig 8. Checkout and taking the change

This figures illustrates the process of product recognition as implemented in the smart shopping cart system. The setup consists of the following key components and operations:

1. **Camera Module**: Positioned to capture high-resolution images of the cart's interior. When an item is placed in the cart, the camera takes a snapshot, ensuring optimal angles for object detection.
2. **Image Processing**: The captured image is sent to the Raspberry Pi, where it is processed using pre-trained machine learning models. The **Custom RNN** object detection algorithm is employed for its high accuracy and real-time processing capability.

3. **Object Detection**: The Custom RNN model identifies the product by detecting its label or distinguishing features in the image. Detected objects are classified into predefined categories stored in the database.
4. **Database Lookup**: Upon recognition, the system queries the local SQLite database to retrieve corresponding product details, such as the item's name, price, and expected weight.
5. **Display Output**: The recognized product information is displayed on the shopping cart's interface .This includes the product name, price,discount and a running total of the bill.

The figures effectively demonstrates the integration of **computer vision and machine learning** to automate product recognition, eliminating the need for manual barcode scanning. This approach ensures a seamless and efficient shopping experience while operating on an edge computing framework using Raspberry Pi.

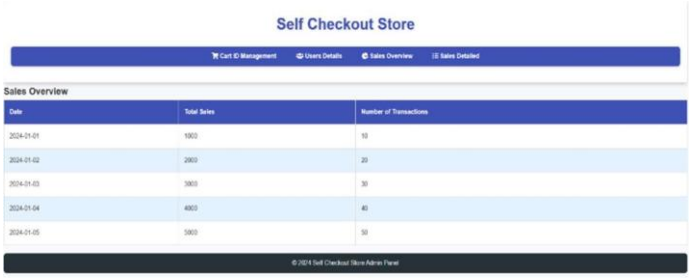


Fig 9. Admin panel for Sales Detailed

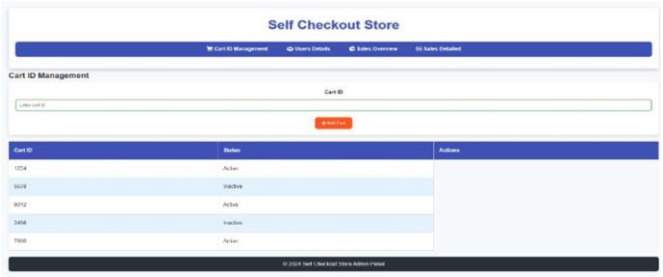


Fig 10. Admin panel for Cart Management

This figures represents the interface and architecture of the admin panel used in the smart shopping cart system. The admin panel is a central hub for managing retail operations, providing the following functionalities:

1. **Sales Management**: Displays real-time and historical sales data, including total revenue, sales trends, and frequently purchased products. Offers filtering and sorting options to analyze performance by product category, time periods, or customer segments.
2. **Cart Management**:Tracks individual cart statuses during active shopping sessions. Displays cart contents, total prices, and any flagged errors (e.g., unrecognized products or weight mismatches).Allows manual intervention by administrators to resolve discrepancies or update product information.
3. **Analytics Dashboard**: Visualizes key performance indicators (KPIs) using graphs and charts for better decision-making. Provides insights into customer behavior, inventory usage, and product popularity.Generates predictive analytics using machine learning models to forecast demand or optimize stock levels.
4. **Inventory Management Integration**: Syncs product information with inventory databases to ensure accurate stock tracking and automated replenishment alerts.

This admin panel, hosted on a web-based application, bridges the gap between the smart cart's operational data and business insights, facilitating streamlined management and strategic decision-making. The figure showcases how technology drives efficiency and enhances the overall shopping experience for both customers and retailers

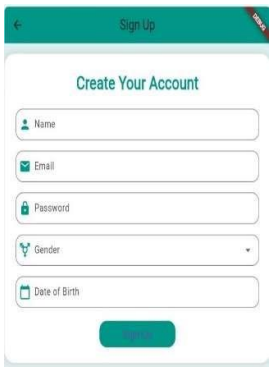


Fig.10 Application Signup Page


Fig.11 Application Checkout page with Advertisements

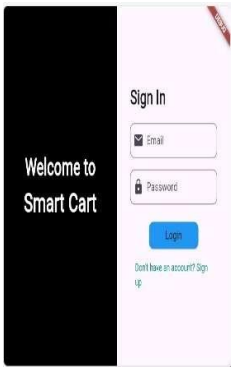


Fig.12 Application Sign in page

The figure illustrates key UI designs for the mobile application associated with a smart shopping cart system. It highlights three critical functionalities: advertisement display, checkout process, and sign-in/sign-up features, which collectively enhance the shopping experience. Below is a detailed explanation of each component:

**Sign-In/Sign-Up Interface:**

1. **User Authentication:** Provides secure login for existing users via email, phone number, or social media accounts. New users can create an account by signing up with minimal information, such as name, email, phone number, and a password.
2. **Account Personalization:** Allows users to set preferences, such as favourite product categories, dietary restrictions, or shopping goals. Ensures the application tailors advertisements and recommendations to individual needs.
3. **Seamless Integration:** Once logged in, the user's data (e.g., shopping history, saved items, loyalty points) synchronizes across devices.
4. **Password Recovery:** Offers an easy way to reset passwords via email or SMS in case of forgotten credentials.

**Advertisement Interface:**

1. **Personalized Promotions:** Delivers targeted ads based on the user's shopping history, preferences, or account settings. Displays discounts, product recommendations, or exclusive offers.
2. **Dynamic Updates:** Real-time updates to ads based on the user's cart, store promotions, or seasonal campaigns.
3. **Interactive Features:** Users can engage with ads by adding products to the cart, viewing product details, or exploring alternatives.

**Checkout Interface:**

1. **Cart Summary:** Presents an organized list of products, including quantities, prices, and discounts applied.
2. **Billing Options:** Supports multiple payment methods, such as credit/debit cards, digital wallets, and UPI.
3. **QR Code Scanning:** Simplifies the checkout process by syncing cart contents with the app through a QR code.
4. **Final Confirmation:** Displays the total bill for review and generates an electronic receipt after successful payment.

## V. Conclusion

This paper provides a detailed exploration of the development and implementation of a self-shopping cart powered by Raspberry Pi and computer vision. The integration of hardware components with state-of-the-art computer vision algorithms enables real-time product recognition, accurate billing, and streamlined checkout processes. By addressing common challenges such as lighting variability and computational limitations, the prototype demonstrates significant potential for practical deployment in retail environments. Future enhancements, including more robust anomaly detection and expanded hardware capabilities, could further improve the system's reliability and scalability. Ultimately, this self-shopping cart serves as a testament to the transformative potential of affordable AI and IoT technologies in modernizing retail operations, paving the way for a more seamless and efficient shopping experience.

## Acknowledgements

We like to share our sincere gratitude to all those who help us in completion of this project. During the work we faced many challenges due to our lack of knowledge and experience but those people helped us to get over all the difficulties and in final compilation of our idea to a shaped and completed project. We would like to thank Prof. Jyoti Deore for her governance and guidance, because of which our whole team was able to learn the minute aspects of a project work. We would also like to show our gratitude to all the AI/ML developers and researchers who have provided us with the required literature to refer to. All of our team members are thankful to all the Professors, HOD, Principal and all the Staff of Department of Electronics and Telecommunication Engineering, VIVA Institute of Technology, for their sincere help and support towards this project and our team. We are also thankful to our whole class and most of all to our parents and friends who have inspired us to face all the challenges and overcome all the hurdles in life. "We extend our heartfelt appreciation to our classmates for their constant encouragement and collaborative spirit. Their enthusiasm fuelled our determination to succeed. Special thanks to our families for their unwavering support, patience, and belief in our abilities. Their love and encouragement gave us the strength to persevere. Lastly, we express

our deep gratitude to the entire campus community for fostering an environment of learning and innovation. This project wouldn't have been possible without the collective support and belief in our vision. Thank you all for being an essential part of our journey!"

## References

### Journal Papers:

[1] R. P. Sudhakar and P. S. Chaitanya, "Automated Smart Trolley Using RFID and IoT," in *Proceedings of the International Conference on Electronics and Communication Systems (ICECS)*, pp. 847-850, 2020.
[2] K. Rajalakshmi, R. S. Ranjani, and M. S. Venkatesan, "Smart Shopping Cart Using Computer Vision and IoT," *IEEE Access*, vol. 9, pp. 54675-54684, 2021.
[3] A. Kumar and R. Gupta, "A Vision-Based Smart Shopping Cart System for Automated Billing Using Raspberry Pi," *Journal of Computing and Information Technology*, vol. 28, no. 4, pp. 335-342, 2020.
[4] M. A. Salam, A. S. Iqbal, and M. A. Khan, "Optimized Smart Shopping Cart Using Edge Computing," *IEEE Sensors Journal*, vol. 22, no. 1, pp. 567-574, 2022.
[5] M. H. Chowdhury et al., "IoT-Based Smart Cart System for Modern Retail Environments," in *Proceedings of the IEEE Internet of Things Conference (IoTConf)*, pp. 211-216, 2022.

### Books:

[6] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
[7] Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. arXiv preprint.
[8] Raspberry Pi Foundation. (2023). Raspberry Pi Documentation. [Online]. Available: https://www.raspberrypi.org/documentation
[9] OpenCV Documentation. (2023). Open-Source Computer Vision Library. [Online]. Available: https://opencv.org/documentation