Integration of Internet of Things (IoT) and React Native-Based Mobile Application for Smart Trolley System Using ESP32 and Firebase

Glenn Putra Laymando
School of Computer Sciences
Bina Nusantara University
Jakarta, Indonesia
glenn.laymando@binus.ac.id

Theo Filus Iglesias Triadi
School of Computer Sciences
Bina Nusantara University
Jakarta, Indonesia
theo.triadi@binus.ac.id

Justin Sufrapto
School of Computer Sciences
Bina Nusantara University
Jakarta, Indonesia
justin.sufrapto@binus.ac.id

Abstract — **Along with the development of the era and technology, practicality has become a primary need sought by many people. This drives innovation in various sectors, including the retail and shopping sectors. Shopping trolleys that have been used in supermarkets are still conventional, where consumers must estimate the total price of their purchases themselves and still have to queue at the cashier to make payments. Therefore, innovation is needed in the form of smart trolleys that can increase efficiency and convenience in shopping. Smart Trolley is here as a solution to this problem.**

The Smart Trolley automates the traditional shopping process by allowing users to control the movement of the trolley through a mobile application and scan product QR codes. The scanned product data is displayed in real time on the app, along with the total accumulated shopping cost.

This device uses ESP32-CAM as the main controller, which is connected to the 4WD motor via the L298N motor driver and powered by a 2S LiPo battery. This trolley is connected to the mobile application via a Wi-Fi hotspot connection from a smartphone. The QR code scanning process is carried out using the ESP32-CAM's built-in camera. The firmware is developed using the Arduino IDE, while the mobile application serves as the user interface.

The test results show that the Smart Trolley can perform basic movements and barcode scanning functions well, although there are some obstacles such as unstable connections and occasional scan failures. Future developments are focused on improving scanning speed and accuracy, as well as adding security features in the form of an automatic cover that only opens after the product has been successfully scanned, to support a safer and more independent transaction system.

Keywords—*Smart Trolley*, *Internet of Things (IoT)*, *ESP32*, *React Native*, *Firebase*

I. Introduction

The development of Internet of Things (IoT) technology has presented innovative solutions in everyday life, including in the retail and shopping sector. One of the main challenges in supermarkets is the long queues at the cashier and the lack of efficiency in the process of recording and paying for goods. Customers often have to push the trolley manually which can be difficult for the elderly and parents with small children. In addition, customers also have to estimate the total shopping manually and queue for the payment process. In the era of automation, these challenges can be overcome by integrating intelligent technology, such as IoT-based smart trolleys whose movements can be controlled via mobile applications and automatically scan products.

This project aims to design and implement a Smart Trolley, an IoT-based shopping trolley that can be controlled via a mobile application and is equipped with a product qr code scanning feature. The main motivation of this project is to increase efficiency and convenience in shopping, reduce dependence on cashiers, and introduce an affordable retail automation solution that can be integrated into a modern supermarket environment.

This project includes the development of a smart trolley system consisting of several Key features:

- Movement Control: The trolley can be moved forward, backward, turn left and right via the mobile application
- QR code scanning: Using the camera from the ESP32-CAM module to scan the product QR code
- Wi-Fi connection: The system uses a Wi-Fi hotspot connection to connect the trolley to the application
- Mobile App Integration: The mobile application displays a list of scanned products and displays the total price, and can be used for trolley control
- Hardware Platform: The system is built using ESP32-CAM, DC motors with l298N drivers, and a 4WD car kit powered by a LiPo battery

II. Related Work

The use of Internet of Things technology in the retail sector continues to grow over time. Early studies such as [1], [2], [12], [14], and [16] using RFID technology can automate product identification which can speed up the checkout process with information displays on the LCD trolley. Another study [4] tried to integrate this trolley system with QR code scanning, weight sensors, and automatic trolley control using NodeMCU ESP8266. However, the challenges of Wi-Fi connection stability and security still need to be a major concern. Other studies [6] and [7] also support this by emphasizing that IoT can optimize cross-sector businesses such as retail by increasing efficiency, cost savings, and improving customer experience, but still requires strict security policies, human resource training, and integration with complex legacy infrastructure. Research [13] proposed adding security so that the trolley cannot be taken out before the payment process, as well as verifying the weight of the goods to prevent errors or fraud.

The use of IoT-based multi-motor servo control systems and controlled via an Android application has also been carried out in a study [5]. The use of the MQTT HiveMQ protocol as a communication broker allows distance control without physical limitations. Although there are hardware limitations that affect accuracy, this approach opens up opportunities for the development of IoT trolley systems. The use of IoT in studies [9], [10], and [11] also shows that IoT technology is able to provide effective, flexible monitoring control solutions, and can also operate optimally up to several kilometers with fairly stable WiFi.

Integration of mobile applications with IoT-based devices is also widely used, such as the use in research [3]. In this study, smartphones as barcode and Bluetooth scanning interfaces are combined with the use of RFID. However, although effective, it has constraints in cost and implementation flexibility. In research [20], an application was developed that uses a wheel system by utilizing NodeMCU ESP8266 for internet connectivity, Arduino Uno, and L298N Motor Driver. The integration of Android applications that display joysticks to perform movement can be integrated into the movement of smart trolleys.

The combination of ESP32, Websocket, and Firebase technology can be a solution for a low-cost, portable, and easy-to-implement real-time communication platform and data storage on a smart trolley system [8]. The use of the Firebase real-time database and ESP32 was also successful in a study [17] that successfully sent data in real time to the cloud, allowing Android applications to display automatic updates without distance limitations.

Several studies have tried to add various other features to their smart trolleys. In Research [18] tried to add a machine learning system for stock prediction, automation when putting products into the basket with proximity sensors and IR sensors. Then Research in [15] added an RF module, which provides a warning if the child moves away from the trolley which can increase the safety aspect of shopping in crowded places.

III. System Design And Architecture

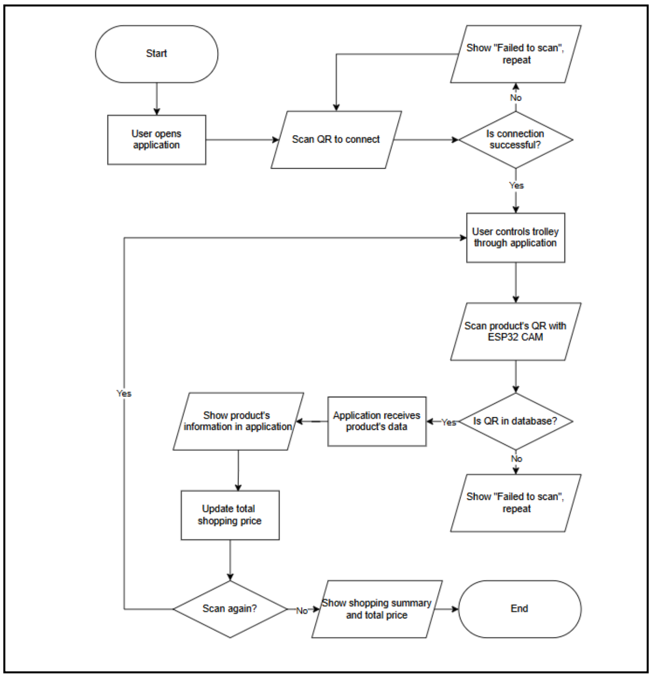

Figure 1. IoT Smart Trolley flowchart

3.1 Flowchart of IoT Smart Trolley

The flowchart in Figure 1 shows the operational sequence of the IoT Smart Trolley system, which integrates mobile app control and QR code-based product scanning. At the beginning, the user opens the mobile application. The first step is to scan a QR code to initiate a connection between the mobile app and the trolley system. If the connection fails, an error message is shown: "Failed to scan", prompting the user to try again. Once connected successfully, the user can control the movement of the trolley via the application. The next step is to scan the product's QR code using the ESP32-CAM camera module on the trolley.

The system then checks whether the scanned QR code is valid and exists in the database. If it does not, the system once again displays "Failed to scan", and the process is repeated. If the QR code is valid, the product data is received by the application and the product's information is displayed. The system updates the total shopping price accordingly. The user is then asked whether they want to continue scanning more items. If yes, the scanning process is repeated. If not, the application proceeds to show the complete shopping summary and the final total price, marking the end of the transaction. This flowchart effectively demonstrates how the Smart Trolley automates traditional shopping by combining mobility control, QR scanning, and real-time product tracking and pricing in a seamless loop via IoT and mobile technologies.

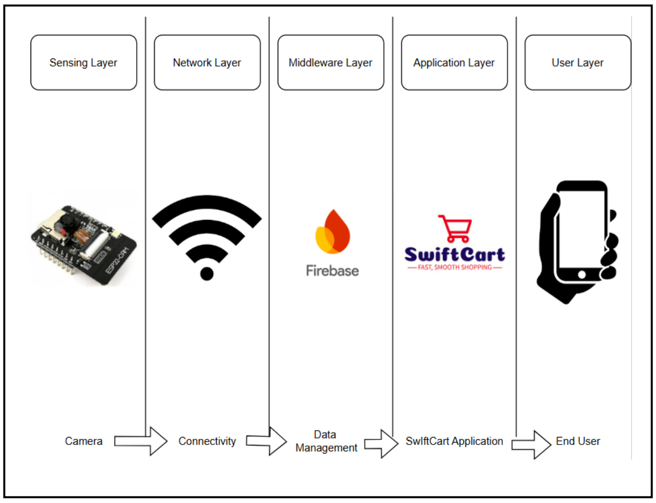

Figure 2. The architecture of IoT Smart Trolley

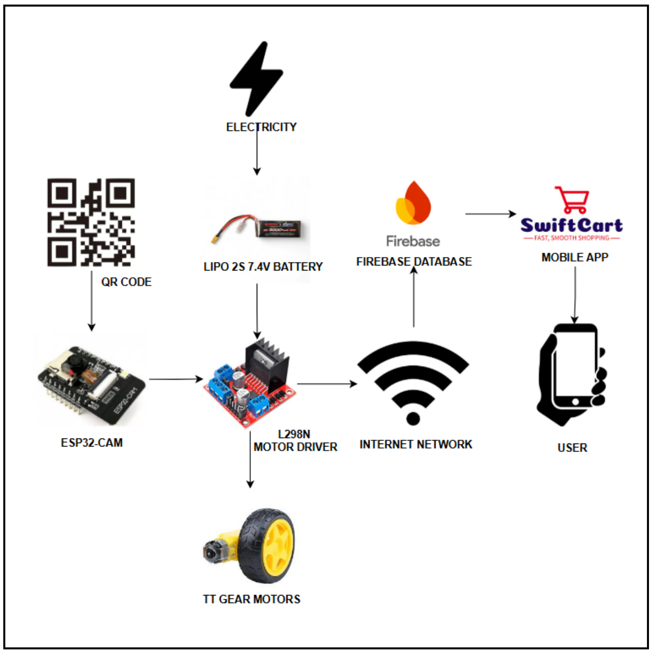

Figure 3. Sketch of improved Smart Trolley

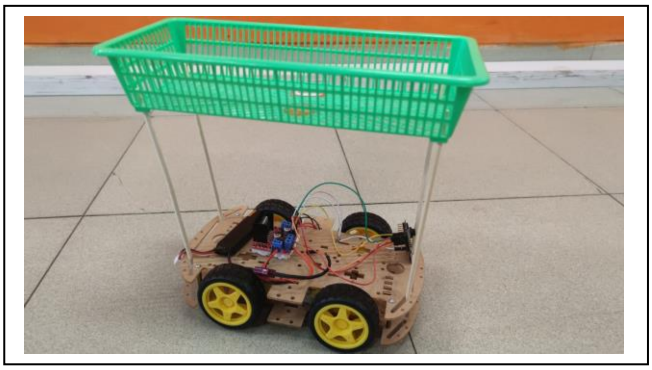

Figure 4. Smart Trolley device prototype

3.2 Architecture of IoT Smart Trolley

The architecture of the IoT Smart Trolley system, shown in Figure 2, is divided into five main layers: the Sensing Layer, Network Layer, Middleware Layer, Application Layer, and User Layer. In the Sensing Layer, the ESP32-CAM module is used as the core sensor, equipped with a built-in camera to scan QR codes placed on products. This module acts as the main input component, gathering data from the physical environment. Next, in the Network Layer, the data collected from the ESP32-CAM is transmitted via a Wi-Fi connection, which serves as the communication medium between the device and the cloud infrastructure. The Middleware Layer handles data management using Firebase, a real-time cloud-based database that stores and synchronizes scanned product data. Firebase also facilitates efficient and scalable backend communication between the IoT device and the mobile application.

In the Application Layer, a mobile app named SwiftCart is used. It receives real-time product data, displays the list of scanned items, shows the total shopping cost, and allows the user to control the movement of the trolley. Finally, the User Layer represents the interaction between the end-user and the system, allowing users to receive processed data and control the trolley directly from their smartphones This layered architecture provides a modular and scalable foundation for smart retail systems, ensuring real-time interaction, seamless connectivity, and effective data management between physical devices and digital interfaces.

3.3 Advanced Sketching and Prototype

The sketch and prototype of the IoT Smart Trolley are presented in Figure 3 and Figure 4, respectively. This system integrates both hardware and software components to automate the shopping experience through remote control and product scanning features. At the core of the design is the ESP32-CAM, which acts as the main controller. It is responsible for scanning product QR codes using its built-in camera. The scanned data is transmitted via the Internet network to Firebase, a cloud-based real-time database that stores the product information and makes it accessible to the mobile application in real time. The ESP32-CAM is powered by a LiPo 2S 7.4V battery, which also supplies energy to the L298N motor driver. The motor driver controls TT gear motors, enabling the trolley to move in various directions based on user input from the mobile application.

The mobile app, called SwiftCart, serves as the interface between the user and the trolley system. It displays scanned product information, shows the total shopping cost, and provides movement controls. Communication between the app and hardware is facilitated through Wi-Fi connectivity. The physical prototype, shown in the bottom part of Figure 3, demonstrates a functional shopping trolley with a mounted basket on a 4WD chassis, complete with electronic wiring and mounted hardware. This prototype confirms that the system can operate effectively in real-world scenarios, supporting mobility, scanning, and real-time data updates. Together, the sketch and prototype validate the functionality of the IoT Smart Trolley, showcasing its potential to enhance the retail shopping experience by combining movement automation, QR scanning, cloud connectivity, and smartphone control in one cohesive system.

IV. Implementation

4.1 Hardware Setup

The hardware setup of the IoT Smart Trolley consists of various interconnected components that enable movement control, product scanning, and communication with a mobile application via the Internet of Things (IoT).

A. Components:

The IoT Smart Trolley system consists of a combination of electronic and mechanical components that work together to enable mobility, scanning, and wireless communication. Each component has a specific function within the system, contributing to the overall performance of the trolley. The setup supports features such as remote control via a mobile app, product QR code scanning, and real-time database interaction.

1. Mini Car Chassis

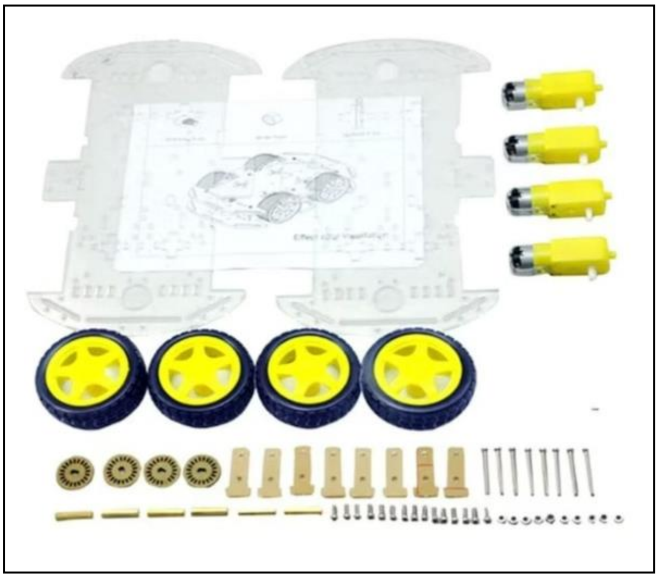

Figure 5. Mini Car Chassis component

The mini mobile chassis acts as the physical frame that holds and supports all the components of the smart trolley. Made from durable and lightweight materials, it serves as the foundation on which the motors, driver board, and controller are mounted. It provides mobility and structure to the system. It also comes with 4 TT Gear Motors and 4 Mini Car Chassis' wheels to provide mobility for the prototype, and the additional brackets and screws to help in building the chassis and keeping the motors from moving loose.

2. ESP32-CAM

Figure 6. ESP32-CAM component

The ESP32-CAM module serves as the main controller for the system. It includes an integrated camera used for scanning product QR codes and built-in Wi-Fi for wireless communication with the mobile application and Firebase. It also handles the control signals for movement and processing scan data.

3. Arduino Uno

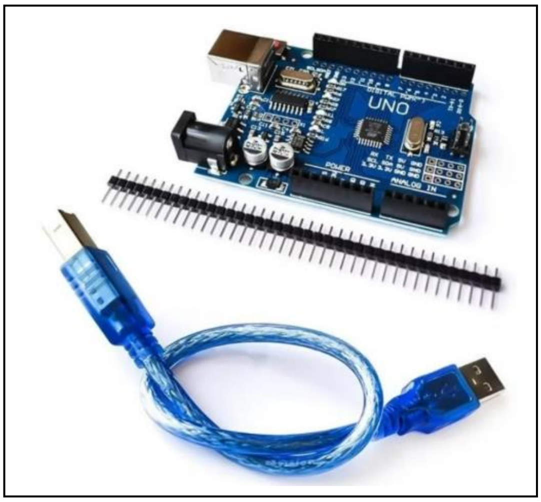

Figure 7. Arduino Uno component

The Arduino Uno may be used as a secondary microcontroller, depending on the design integration. It helps in extending hardware capabilities by managing additional input/output operations or assisting in communication between other components through uploaded codes. However, its use is optional if the ESP32-CAM is sufficient for all tasks.

4. Motor Driver Board (L298N)

Figure 8. Motor Driver Board L298N component

The L298N motor driver controls the speed and direction of the DC motors. It receives control signals from the ESP32-CAM and applies power to the motors accordingly. It supports two-channel motor control, making it ideal for 4WD (four-wheel drive) robotic chassis used in the trolley.

5. LiPo 2S 7.4V Battery

Figure 8. LiPo 2S 7.4V Battery component

The LiPo 2S 7.4V battery serves as the main power source for the smart trolley. It supplies sufficient voltage and current to power both the motor driver and the ESP32-CAM. LiPo batteries are chosen for their lightweight, high energy density, and rechargeability. They also come with deans cable that can only be charged with a compact charger (in this case, we use imaxRC B3 PRO compact charger).

6. Jumper Wires

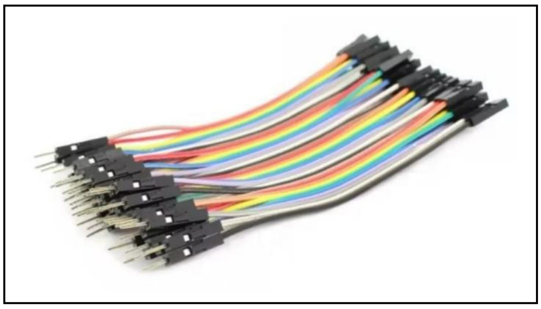

Figure 9. Jumper Wires component

Jumper wires are used to establish electrical connections between various components on the trolley. These flexible wires allow data and power signals to be transferred between the controller, motor driver, and motors. They are essential for assembling and prototyping electronic circuits.

These parts collectively provide the structure, power, sensing, and communication capabilities for the smart trolley system.

B. Installation/Building Prototype

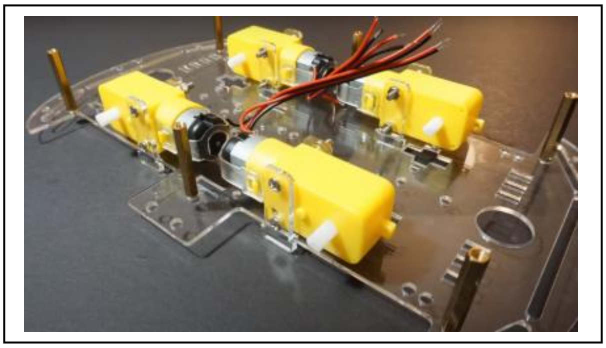

Figure 6. Applying TT gear motors to the car chassis

Figure 6 shows the installation of TT gear motors on the car chassis. The trolley uses four TT motors (two on each side), enabling full directional control—forward, backward, left, and right. The motors are mounted securely on the acrylic base using screws and brackets. Each motor is connected to the motor driver through soldered red and black wire cable for signal and power delivery. The TT motors are DC motors with built-in gearboxes, which reduce speed and increase torque—ideal for carrying the weight of the trolley and its components.

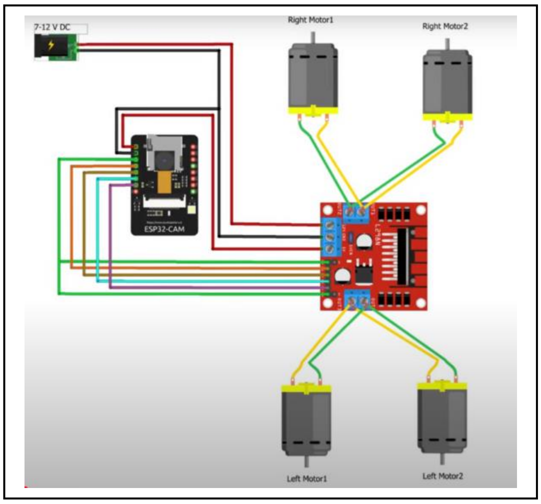

Figure 7. Connections of Smart Trolley's structure

Figure 7 shows the electrical connections among the components. The central controller (ESP32-CAM) is connected to the L298N motor driver, which acts as the interface between the controller and the motors. The L298N board can control two pairs of motors (left and right) through its dual H-bridge configuration, allowing independent control of motor direction and speed.

Each of the four TT motors is wired into the motor driver—Left Motor 1 and 2, and Right Motor 1 and 2. The power supply from the LiPo battery is connected to the driver and controller to energize the entire system. The motor driver receives command signals from the ESP32-CAM and activates the motors accordingly.

4.2 Mobile Application

The SwiftCart mobile application acts as the user interface for the IoT Smart Trolley system. Developed using React Native, the app provides a real-time, cross-platform experience for users to control the trolley, scan product QR codes, monitor shopping carts, and perform digital checkouts. The following sections describe the application features in detail, as illustrated in Figures 8 to 10.

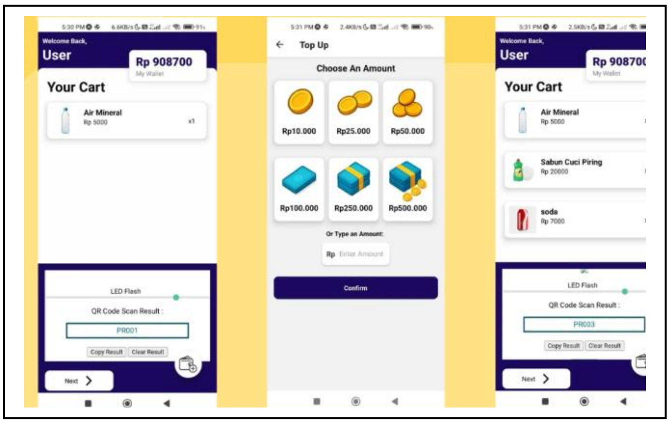

Figure 8. SwiftCart interface for payment feature

a. Payment and Product Scanning Interface

Figure 8 shows the SwiftCart interface for scanning products and managing the shopping cart. The user interface displays:

- A list of scanned products including item name and price,
- The wallet balance (in this case, Rp 908700),
- A QR code scan result box, where scanned QR values like "PR001" are displayed,
- A button to copy or clear the scan result, and
- A "Next" button to proceed to the checkout phase.
- Users can also top up their account through a Top Up screen, where they can choose predefined denominations (e.g., Rp10.000, Rp25.000, etc).

  Once scanned, products are added to the cart and displayed for user review in real-time.

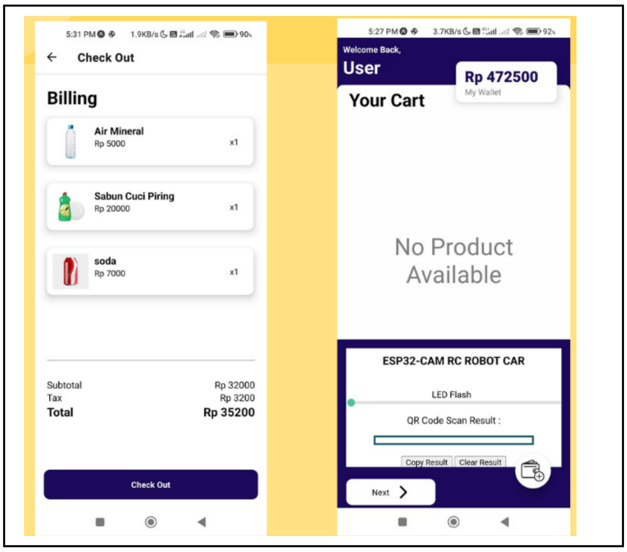

Figure 9. SwiftCart interface for checkout feature

b. Checkout Interface

Figure 9 presents the checkout screen in SwiftCart. After all desired products have been scanned and added to the cart, users are directed to a summary page. This section shows:
- Each scanned item with quantity and price,
- A billing summary, including subtotal, tax, and final total (e.g., Rp 35200),
- A Check Out button to finalize the transaction.

This design allows the user to simulate a self-checkout process, helping to reduce cashier wait times and streamline in-store transactions. Once completed, the user's wallet balance will be deducted after a short delay of processing the confirmed payment.

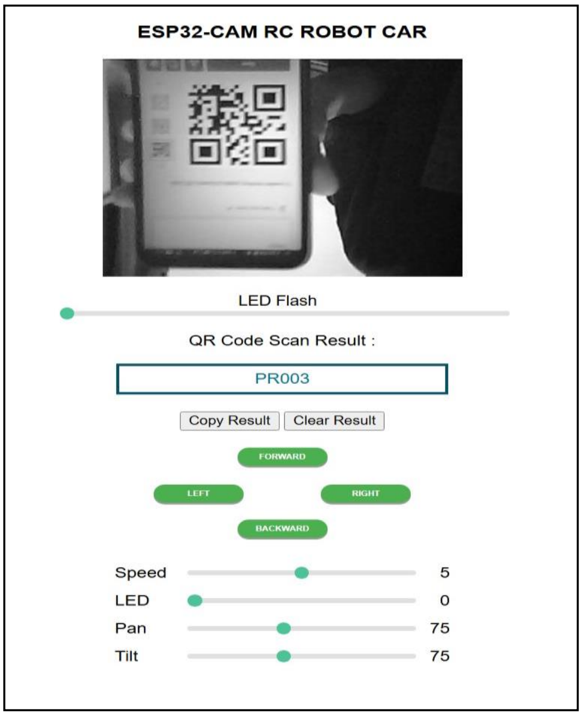

Figure 10. SwiftCart screen display for move/scan feature

c. Cart Movement and Scan Control Panel

Figure 10 illustrates the screen responsible for controlling trolley movement and scanning. This interface is integrated with the ESP32-CAM and provides real-time interaction features such as:

- Live camera feed showing the QR code being scanned,
- Scan result display area (e.g., "PR003"),
- Navigation controls (FORWARD, BACKWARD, LEFT, RIGHT) to move the trolley,
- Adjustable parameters for Speed of the trolley, LED flash intensity, and Camera pan and tilt using sliders.

This panel offers direct, responsive control over the trolley's physical operations and scanning, allowing the user to navigate and interact with the environment effectively.

V. EVALUATION AND TESTING

This section presents the testing and evaluation of the smart trolley system, focusing on the performance of the QR code scanning functionality. Two aspects were evaluated: (1) the success rate of product QR code scans, and (2) the time taken by the ESP32-CAM to read a QR code.

5.1 Functional Testing: QR Code Scan Success Rate

The QR code scanning function was tested over 10 trials using different product QR codes under normal lighting and camera conditions. Out of the 10 attempts, 8 were successful, resulting in a success rate of 80%. Failures occurred mainly due to poor lighting and improper camera distance, which caused difficulty in detecting the QR code. The summary of results is shown in Table 1.

Table 1. QR Code Scan Success Rate

Trial	Product	Scan Result	Notes
1	Product1	Success	Quick and accurate
2	Product2	Success	Slight angle, still readable
3	Product3	Fail	Low lighting
4	Product4	Success	Normal
5	Product5	Fail	Too far from camera
6	Product6	Success	Normal
7	Product7	Success	2 second delay
8	Product8	Success	Adequate lighting
9	Product9	Success	Slightly blurry, still worked
10	Product10	Success	Fast detection

Overall, the system demonstrates a high degree of reliability in reading QR codes, although minor environmental factors can affect performance.

5.2 Functional Testing: QR Code Scan Success Rate

The time required by the ESP32-CAM to read a QR code was measured across 10 trials. The results ranged from 2 to 4 seconds, with an average scan time of 2.8 seconds. This is considered acceptable for real-time trolley use, though further optimization may reduce this delay. Table 2 presents detailed timing results.

Table 2. QR Code Reading Time

Trial	Reading Time (seconds)	Notes
1	2	Fast detection
2	2	Fast detection
3	3	Slight delay
4	4	Low lighting
5	3	Normal
6	2	Fast
7	3	Slight angle
8	3	Normal
9	4	Far from Camera
10	2	Very quick

Despite occasional delays due to lighting or angle, the average time remains within an acceptable range for practical use.

5.3 Discussion

The testing shows that the QR code scanning feature is generally robust, with an 80% success rate and an average scan time of under 3 seconds. Failures were mostly attributed to non-ideal conditions, such as insufficient lighting or improper distance from the camera.

It is important to note that the ESP32-CAM module used in this system is equipped with a controllable onboard LED (flash). This LED can be programmatically turned on to provide better illumination when scanning QR codes, especially in low-light environments. During testing, the LED was not always enabled, which likely contributed to some failed scans.

To further improve the system, it is recommended to implement automatic lighting control, where the LED is turned on during scanning operations in dim conditions. This would likely increase the success rate and reduce the average scan time, leading to a more efficient and reliable user experience.

Overall, the performance is adequate for the intended application in a smart trolley environment, and with minor enhancements such as dynamic lighting, it can be further optimized.

VI. Conclusion

The Smart Trolley project was successfully developed as a prototype of an Internet of Things (IoT)-based smart trolley that can be controlled via a mobile application and is equipped with a product QR code scanning feature. Test results show that the trolley can move forward, backward, and turn according to commands from the application interface. The QR scanning feature is also functional, with a recorded success rate of 80% and an average scan time of approximately 2.8 seconds.

The system evaluation indicates that the core functionalities perform adequately, though environmental factors such as lighting and camera angle can influence scan accuracy and speed. As the ESP32-CAM module includes an onboard LED light, incorporating automatic lighting control during scanning is recommended to improve performance.

With features such as trolley navigation, scanned product display, price calculation, and self-checkout, this project contributes meaningfully to the development of automated shopping solutions in supermarket environments.

However, some technical challenges still need to be fixed, such as the stability of the connection between the ESP32-CAM and the mobile application, as well as improving the accuracy and speed in reading the qr code. As a future development, the system can be equipped with:

- A security system in the form of a trolley cover that will only open when the customer has scanned the qr code and when the user has finished making the payment
- An automatic navigation system using indoor sensors or GPS.

Acknowledgment

The author would like to express his deepest gratitude to the Internet of Things (IoT) course supervisor for his guidance and support on this research IoT project. Thanks also to the authors and researchers indexed in Scopus and Google Scholar who have provided insights and references to this research.

References

[1] Smart Shopping Trolley System using IoT Journal of Physics: Conference Series, 1937 (2021) 012042. DOI: 10.1088/1742-6596/1937/1/012042

[2] Dhavale, S. D., Dhokane, T. J., & Shinde, P. S. (2016). IOT Based Intelligent Trolley for Shopping Mall. International Journal of Engineering Development and Research (IJEDR), 4(2), 1283–1285. ISSN: 2321-9939.

[3] Bedi, H. S., Goyal, N., Kumar, S., & Gupta, A. (2017). Smart Trolley using Smart Phone and Arduino. Journal of Electrical & Electronic Systems, 6(2):223. DOI: 10.4172/2332-0796.1000223

[4] Patil, M. S., Deore, S. N., & Bisht, M. H. (2022). Smart Trolley System. International Journal for Research in Applied Science & Engineering Technology (IJRASET), 10(11).

[5] Sedana Yoga, I. G. A. M., Piarsa, I. N., & Buana, P. W. (2025). Kontrol Multi Motor Servo dengan Joystick Virtual Berbasis Android. JUTIK : Jurnal Teknologi Informasi dan Komputer, 11(1), 1–13.

[6] Anwar, R. (2024). Mengoptimalkan Bisnis dengan IoT: Solusi Cerdas untuk Masa Depan. The 5th International Conference on Information Technology and Security. Universitas Teknokrat Indonesia.

[7] Saputra, A., & Prabowo, W. W. (2023). Menuju Era Retail Cerdas Berbasis Internet of Things (IoT). KRAKATAU: Jurnal Teknologi, Informasi, dan Komputer, 6(4).

[8] Pradana, A., & Nurfiana. (2019, November 7). Rancang Bangun Monitor dan Kontrol Suhu Ruang Server Menggunakan Perangkat Mobile Berbasis Internet of Things (IoT). Prosiding SNRT (Seminar Nasional Riset Terapan), Politeknik Negeri Banjarmasin. ISSN 2341-5662 (Cetak), ISSN 2341-5670 (Online).

[9] Nasron, N., Rose, M. M., & Irma, A. (2020). Implementasi Aplikasi Berbasis Teknologi IoT pada Perangkat Tracking dan Kendali Kendaraan Bermotor. Jurnal Computer Science and Information Technology (CoSciTech), 1(2), 57–64. https://doi.org/10.37859/coscitech.v1i2.2191

[10] Setyadi, J., & Wahidin, A. J. (2024). Sistem Monitoring dan Kontrol Berbasis IoT pada Pintu dan Penerangan Rumah Menggunakan Aplikasi Smartphone. Jurnal Komisi (Komputer dan Sistem Informasi), 1(2), 42–48.

[11] Samsudin, S., Abdullah, A., Abdul Muni, A., & Niansyah, N. (2025). Perancangan Alat Pengendali Relay 5V 4-Channel Berbasis IoT dengan WiFi dan Smartphone. Sistemasi: Jurnal Sistem Informasi, 14(3), 1233–1245.

[12] Yusoff, Z. M., Markom, A. M., Khirul Ashar, N. D., & Muhammad, Z. (2020). Development of a Smart Scanner Trolley System for Intelligent Billing Solutions using RFID and IoT System. International Journal of Emerging Trends in Engineering Research, 8(9), 6220–6225. https://doi.org/10.30534/ijeter/2020/212892020

[13] Bhumika, R., Harshitha G. Bhat, Chandana K., Meghashree R. T., & Sangappa, S. B. (2019). Automated shopping trolley for billing system. International Journal of Advance Research, Ideas and Innovations in Technology, 5(2), 1858–1863. ISSN: 2454-132X.

[14] Joseph, Y., Dasari Lakshmi Ganesh, Kandula Ramana Reddy, Baddigam Ratnakar Reddy, & Avulamanda Koteswararao. (2024). IoT Based Smart Shopping Trolley System. International Journal for Modern Trends in Science and Technology, 10(03), 129–133. https://doi.org/10.46501/IJMTST1003020

[15] Kumar, V. S., Tiwari, A., Kancharana, P., & Jalajakshi, Y. (2022). Smart shopping trolley solutions for malls: Leveraging IoT for efficiency. Material Science and Technology, 21(6), 837–841.

[16] Haritha, K. Sivaraman, Priyanka, C., Shwetha, P., Vidyashree Hiremath, & Shilparani. (2018). Automated Smart Trolley for Supermarkets. International Journal of Engineering Research & Technology (IJERT), NCESC 2018 Conference Proceedings, 6(13), Special Issue. ISSN: 2278-0181.

[17] Harlan Kurnia AR. (2025). Implementasi IoT pada Sistem Monitoring Suhu dan Kelembaban Menggunakan ESP32, Firebase dan Kodular. JATI (Jurnal Mahasiswa Teknik Informatika), 9(1), 1781–1787. Universitas Putra Indonesia "YPTK" Padang.

[18] Banoth, R., Bhanu, G., Shishia, N., Vaishnavi, M., Saeed, H. Y., Asha, V., Nijhawan, G., & Khan, I. (2024). Prediction of Stock with On-Go Billing Cart using IoT and Advanced Interdisciplinary Approaches. E3S Web of Conferences, 507, 01013. https://doi.org/10.1051/e3sconf/202450701013

[19] Laksana, D. D., & Zaky, U. (2024). Sistem Monitoring dan Pemberian Pakan Otomatis pada Peternakan Burung dengan Konfigurasi Wi-Fi Dinamis ESP32. Jurnal Informatika Teknologi dan Sains (JINTEKS), 6(4), 894–901. https://doi.org/10.xxxx/jinteks.v6i4.894-901

[20] Al–Janan, D. H., & Muhammad, G. F. (2023). Perancangan Robot Beroda dengan Sistem Kendali Berbasis IoT dan GUI Menggunakan Smartphone Android. Rekayasa Mesin, 14(2), 701–711. https://doi.org/10.21776/jrm.v14i2.1450