SELLA: An IoT-based smart shopping trolley with real-time RFID tracking and automated checkout

Hadj Zerrouki, Salima Azzaz-Rahmani
Department of Telecommunications, Faculty of Electrical Engineering, Djillali Liabes University of Sidi Bel Abbes, Sidi Bel Abbes, Algeria

Article Info

Article history:

Received Aug 11, 2025
Revised Nov 3, 2025
Accepted Dec 8, 2025

Keywords:

Automated checkout
Microcontrollers
Performance evaluation
Radio frequency identification
Retail technology
Smart shopping cart

ABSTRACT

The contemporary retail sector faces a persistent challenge in enhancing in-store customer experience, primarily due to inefficiencies at checkout. This paper presents smart e-cart for lean logistics application (SELLA), a smart shopping trolley system engineered to eliminate this bottleneck. The system's architecture is centered on a Raspberry Pi 4 microcontroller, orchestrating an ultra-high frequency (UHF) radio frequency identification (RFID) subsystem for instantaneous, non-line-of-sight product identification, and a responsive 7-inch touchscreen graphical user interface (GUI) developed in PyQt. The core contribution lies in developing a self-contained shopping solution with integrated payment processing, supported by comprehensive performance validation. We present a detailed methodology, including the system's multi-threaded software architecture and core operational algorithm. Experimental evaluation demonstrates a mean tag detection accuracy of 98.2% under optimal conditions, a robust user interface (UI) latency of under 500 ms, and an average central processing unit (CPU) utilization of 28%, proving system efficiency. Comparative analysis confirms that SELLA's integration of on-trolley automated payment and detailed performance metrics represents a significant advancement over existing prototypes. The system provides a validated, high-performance solution for next-generation smart retail environments.

Corresponding Author:

Hadj Zerrouki
Department of Telecommunications, Faculty of Electrical Engineering
Djillali Liabes University of Sidi Bel Abbes
BP 89, Sidi Bel Abbes, 22000, Algeria
Email: zerrouki.hadj@gmail.com

1. INTRODUCTION

The retail industry seeks to combine the convenience of e-commerce with the tangible experience of physical stores [1]. A major obstacle is the traditional checkout process, which contributes to customer dissatisfaction, queue abandonment, and operational inefficiency [2]. Internet of things (IoT) technologies, and particularly radio frequency identification (RFID), offer the potential to address this issue by enabling automated, real-time product tracking. While prior prototypes have demonstrated RFID-based trolleys, most lack rigorous quantitative validation and full integration of secure, on-board payment [3], [4].

This work introduces smart e-cart for lean logistics application (SELLA), a self-contained smart shopping trolley that addresses these gaps by combining real-time RFID tracking, an intuitive touchscreen interface, and autonomous payment processing. The system emphasizes a scientific validation framework, reporting metrics such as user interface (UI) latency, central processing unit (CPU)/memory usage, and RFID read accuracy under various conditions-factors often omitted in earlier studies. By focusing on performance reproducibility and end-to-end functionality, SELLA aims to bridge the gap between concept and deployable smart retail systems.

A recent survey by Saxena and Dhote [5] highlights the extensive impact of IoT in creating intelligent retail environments, from supply chain management to personalized customer engagement. Among enabling technologies, RFID, particularly in the ultra-high frequency (UHF) band, offers unparalleled advantages over conventional barcodes, including non-line-of-sight reading and simultaneous multi-tag identification [6], [7]. This makes it exceptionally suited for a "scan-as-you-shop" model implemented via a smart trolley.

The evolution of smart shopping systems can be categorized into several key research streams. Early research focused on proving the viability of RFID for automated item detection. Systems proposed by Yusoff *et al.* [8] and Jothi *et al.* [9] successfully demonstrated this using simple microcontrollers like Arduino. While foundational, these systems often offloaded billing to a centralized counter and did not feature sophisticated user interfaces or autonomous payment. More advanced prototypes, such as the one by Pradhan *et al.* [10], utilized a Raspberry Pi microcontroller, hinting at greater processing capabilities but still lacked detailed performance evaluation. The growing enterprise adoption of IoT technologies has been documented in recent studies [11], highlighting both opportunities and challenges. These early systems, including those by Chandrasekar and Sangeetha [12], often relied on centralized billing and lacked integrated payment solutions.

To address potential theft (e.g., placing an item in the cart without its tag), researchers have explored sensor fusion, most commonly combining RFID with weight sensors (load cells). The principle is to cross-validate the scanned items with the total weight in the trolley [13], [14]. Recent work by Subudhi and Ponnalagu [15] and Lakshmi *et al.* [16] further refines this approach, demonstrating improved accuracy in theft detection. However, this method introduces additional hardware complexity and calibration challenges, and can be defeated by items with similar weights.

A parallel stream of research eschews RFID entirely in favor of computer vision (CV). In these systems, cameras mounted on the trolley identify products based on their appearance [17], [18]. Projects like Amazon's Dash Cart are commercial examples. Research by Sarwar *et al.* [19] demonstrates a deep learning-based approach for product recognition. The primary advantage of CV is that it requires no product tagging. However, it faces significant challenges, including high computational demand, sensitivity to item orientation, and failure in cases of occlusion (when one item blocks the view of another) [20], [21].

Architectural considerations are also a key research topic. While many prototypes, including SELLA, use an on-board architecture for processing, others explore cloud-based models. A study by Al-Fuqaha *et al.* [22] discusses the broader implications of cloud-IoT integration for data analytics and scalability. A cloud-based smart cart could offload heavy computation and enable powerful, fleet-wide analytics for retailers [23].

While these foundational works are significant, they often leave critical questions unanswered regarding real-world performance. Key limitations across many studies include a lack of reported UI latency, CPU/memory overhead, or detailed RFID read-rate analysis under varying conditions. The SELLA system, as detailed in this paper, distinguishes itself by focusing on a pure RFID-based approach to minimize hardware complexity, while integrating a fully autonomous payment workflow. Most critically, we address the validation gap by placing a central focus on the scientific validation of our system's performance through repeatable, quantitative experiments. This work builds upon the foundational concepts of systems like [24] and [25] but provides the rigorous performance data necessary for assessing practical viability. Recent advances in smart shopping carts have also explored personalized recommendations to enhance customer experience [26].

This paper introduces SELLA, a smart shopping trolley system designed to fill these gaps. The primary contributions of this work are threefold:

- A fully autonomous system architecture: the design and implementation of a self-contained trolley that handles item identification, real-time billing, and on-board payment processing without reliance on any fixed checkout infrastructure.
- A rigorous scientific validation framework: a detailed experimental methodology for evaluating the performance of such systems, including defined metrics for accuracy, latency, and resource utilization, which are crucial for assessing practical viability.
- A comprehensive performance analysis: the presentation of quantitative results that validate the system's robustness, efficiency, and accuracy, providing a benchmark for future research in this domain.

The remainder of this paper is organized as follows. Section 2 details the proposed system architecture and methodology, including the hardware and software design, as well as the core operational algorithm. Section 3 describes the experimental setup, performance metrics, and the procedure used for rigorous system validation. In section 4, we present and discuss the quantitative results of our experiments, including a comparative analysis against prior art. Finally, section 5 concludes the paper, summarizing our findings and suggesting directions for future research.

2. SYSTEM ARCHITECTURE AND METHODOLOGY

The SELLA system is engineered as an embedded system that integrates hardware and software into a cohesive, high-performance unit. It utilizes a Raspberry Pi as the central controller to manage peripheral communication and data processing tasks efficiently. This design ensures low-latency performance and reliable operation, which are critical requirements for a dynamic retail environment.

2.1. System overview and operational flow

The system is designed to be intuitive. A customer begins by authenticating on the trolley's touchscreen. As they place items with RFID tags into the trolley, they are automatically detected. The item's details appear on the main screen, and the total cost is updated in real-time. If an item is removed, its cost is subtracted from the total. Upon finishing, the customer initiates the payment process, which generates a quick response (QR) code for mobile payment. The entire architecture is illustrated in the block diagram shown in Figure 1.

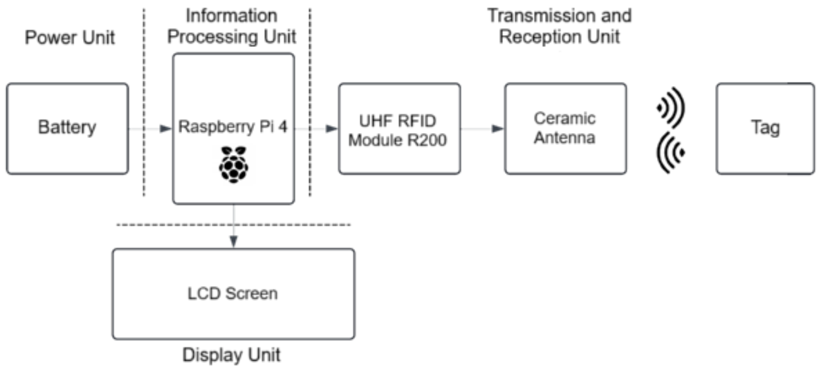


Figure 1. Block diagram of the proposed system

The operational workflow of the SELLA application is illustrated by the flowchart in Figure 2. The application initiates with a login window where the user is required to authenticate. Upon successful authentication, the user gains access to the main application window, which functions as the primary point-of-sale (POS) interface. The system continuously reads the RFID data from products scanned by the reader, identifies the corresponding items in its database, and adds them to the shopping cart. The total price is calculated and displayed in real-time. The user can then proceed to payment, which confirms and records the transaction in the database.

Additionally, the application allows the user to set a spending budget, add items that require weighing, and display supplementary information about the products. Furthermore, the system monitors the presence of RFID tags and automatically removes products from the cart if a tag is withdrawn from the scanning zone.

2.2. Hardware architecture and component justification

The system's hardware components were selected to balance performance, power consumption, and cost. The core components are interconnected as shown in the system schematic (Figure 3).

- Central processing unit (Raspberry Pi 4 Model B): a Raspberry Pi 4 (4 GB RAM) was chosen over less powerful single-board computers or microcontrollers due to its ability to run a full-fledged Linux OS, support the sophisticated PyQt graphical framework smoothly, and handle multi-threaded processing, which is essential for our application.
- UHF RFID subsystem:
  a. Reader module: a module R200 UHF reader operating in the 840-960 MHz band. It supports the ISO 18000-6C/EPC C1G2 protocol and communicates with the Raspberry Pi via a USB-to-serial interface at 115200 bps.

b. Antenna: a passive UHF ceramic antenna with 3 dBi gain and right-hand circular polarization (RHCP). It is optimized for reading up to 50 tags per second at a distance of up to 1 meter, which is necessary to cover the entire volume of a shopping basket, and its robust anti-collision protocol (EPC Gen2), enabling the simultaneous detection of dozens of items.
c. Tags: passive EPC Gen2 UHF tags made from coated paper. Each tag stores a 96-bit EPC and has a data retention life of 10 years.

– User interface (7-inch LCD touchscreen): a high-resolution (800×480) capacitive touchscreen provides an intuitive and modern user interaction medium, essential for customer adoption.

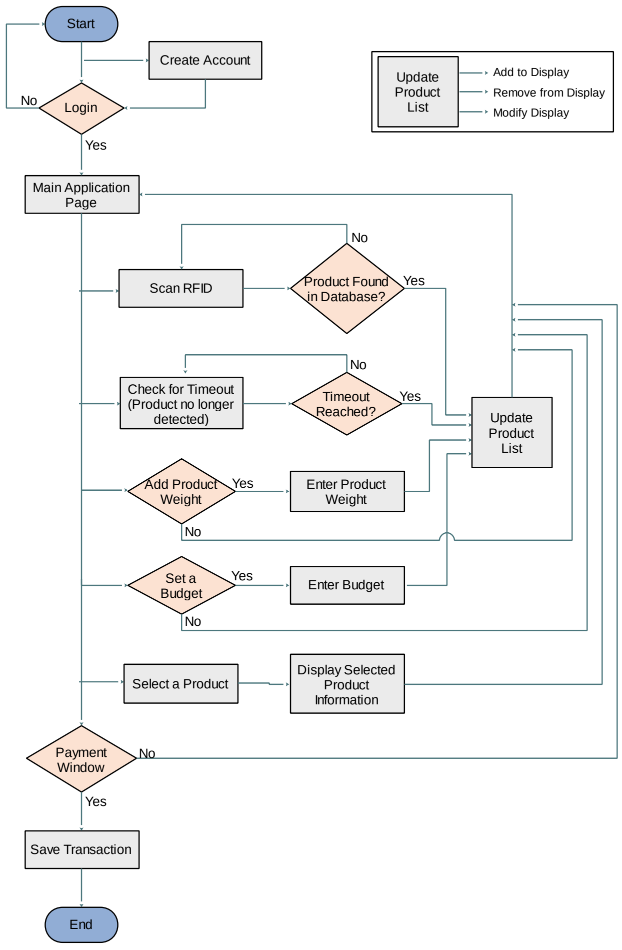


Figure 2. Flowchart of the proposed system

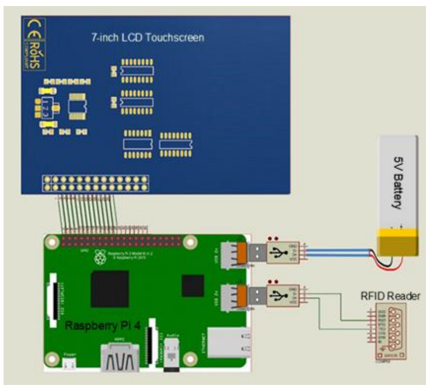


Figure 3. Circuit schematic of the proposed system

2.3. Software architecture

The system's software, developed in Python 3, employs a multi-threaded architecture to ensure a responsive user interface and concurrent background processing.

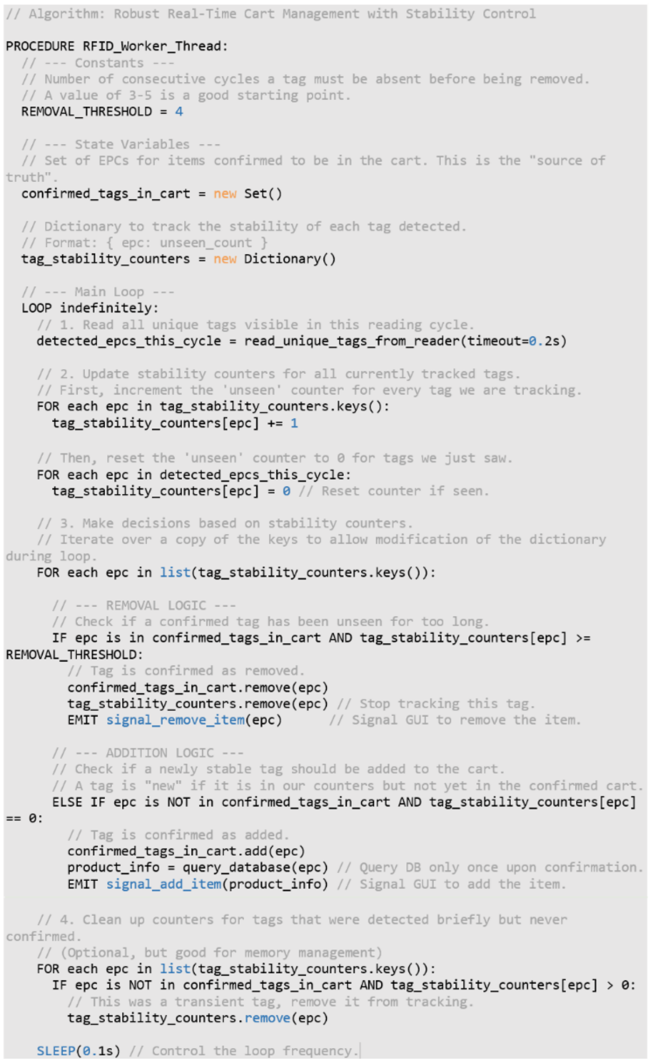
– Main thread (graphical user interface (GUI) event loop): this thread is exclusively responsible for running the PyQt application and handling all user interactions (touch events, button clicks). This separation prevents the UI from freezing during input/output (I/O)-intensive operations.
– Worker thread (RFID polling): a separate background thread is spawned upon application start. Its sole purpose is to continuously poll the USB serial port connected to the RFID reader.

This architecture ensures that the RFID reading process runs independently and communicates with the UI asynchronously via Qt's robust signal-and-slot mechanism, guaranteeing system responsiveness.

2.4. Core system algorithm

The core logic for managing the shopping cart contents is implemented in the RFID worker thread. A simplified pseudocode of the algorithm is presented in Figure 4. The main issue is its sensitivity to intermittent tag reads. In any real RFID environment, a tag that is physically present in the cart might not be detected in every single reading cycle due to:

– Its orientation changing
– Temporary obstruction by other items (especially liquids or metals)
– Radio frequency (RF) signal fluctuations



```
// Algorithm: Robust Real-Time Cart Management with Stability Control

PROCEDURE RFID_Worker_Thread:
  // --- Constants ---
  // Number of consecutive cycles a tag must be absent before being removed.
  // A value of 3-5 is a good starting point.
  REMOVAL_THRESHOLD = 4

  // --- State Variables ---
  // Set of EPCs for items confirmed to be in the cart. This is the "source of
truth".
  confirmed_tags_in_cart = new Set()

  // Dictionary to track the stability of each tag detected.
  // Format: { epc: unseen_count }
  tag_stability_counters = new Dictionary()

  // --- Main Loop ---
  LOOP indefinitely:
    // 1. Read all unique tags visible in this reading cycle.
    detected_epcs_this_cycle = read_unique_tags_from_reader(timeout=0.2s)

    // 2. Update stability counters for all currently tracked tags.
    // First, increment the 'unseen' counter for every tag we are tracking.
    FOR each epc in tag_stability_counters.keys():
      tag_stability_counters[epc] += 1

    // Then, reset the 'unseen' counter to 0 for tags we just saw.
    FOR each epc in detected_epcs_this_cycle:
      tag_stability_counters[epc] = 0 // Reset counter if seen.

    // 3. Make decisions based on stability counters.
    // Iterate over a copy of the keys to allow modification of the dictionary
during loop.
    FOR each epc in list(tag_stability_counters.keys()):

      // --- REMOVAL LOGIC ---
      // Check if a confirmed tag has been unseen for too long.
      IF epc is in confirmed_tags_in_cart AND tag_stability_counters[epc] >=
REMOVAL_THRESHOLD:
        // Tag is confirmed as removed.
        confirmed_tags_in_cart.remove(epc)
        tag_stability_counters.remove(epc) // Stop tracking this tag.
        EMIT signal_remove_item(epc)      // Signal GUI to remove the item.

      // --- ADDITION LOGIC ---
      // Check if a newly stable tag should be added to the cart.
      // A tag is "new" if it is in our counters but not yet in the confirmed cart.
      ELSE IF epc is NOT in confirmed_tags_in_cart AND tag_stability_counters[epc]
== 0:
        // Tag is confirmed as added.
        confirmed_tags_in_cart.add(epc)
        product_info = query_database(epc) // Query DB only once upon confirmation.
        EMIT signal_add_item(product_info) // Signal GUI to add the item.

    // 4. Clean up counters for tags that were detected briefly but never
confirmed.
    // (Optional, but good for memory management)
    FOR each epc in list(tag_stability_counters.keys()):
      IF epc is NOT in confirmed_tags_in_cart AND tag_stability_counters[epc] > 0:
        // This was a transient tag, remove it from tracking.
        tag_stability_counters.remove(epc)

    SLEEP(0.1s) // Control the loop frequency.
```

Figure 4. Pseudocode block for RFID tag handling and cart update logic

To avoid that the algorithm interprets a single missed read as a "removal," deleting the item from the user's screen. In the very next cycle, when the tag is read again, it would be re-added. This would cause items to constantly flicker in and out of the cart on the GUI, which is unacceptable. To solve this, we need to introduce a "debouncing" or "stability" mechanism. An item should only be considered "removed" after it has been consistently absent for several consecutive reading cycles. This ensures that temporary read failures do not affect the cart's state.

2.5. System configuration and prototype assembly

The Raspberry Pi operating system (OS) was installed using the Raspberry Pi imager tool. System configuration, including network settings and peripheral enablement, was performed using the raspi-config command-line utility. Software packages were kept current via sudo apt-get update and dist-upgrade.

As illustrated in Figure 5, the physical assembly involved mounting the liquid crystal display (LCD) screen on the trolley handlebar. A custom, 3D-printed enclosure was designed to house the Raspberry Pi and a 5 V battery pack, which was affixed to the rear of the screen mount. The RFID reader and its antenna were strategically positioned at the base of the trolley basket, angled upwards to maximize coverage within the basket while minimizing stray reads from outside.

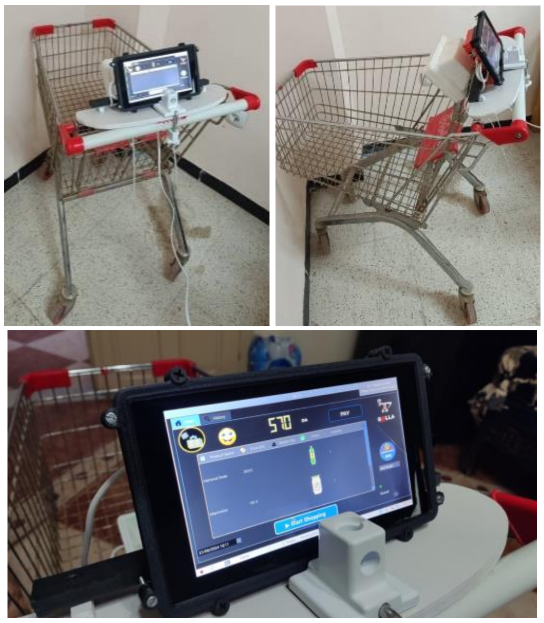


Figure 5. Photos of the fully assembled prototype on the shopping cart

2.6. Software implementation and user interface

The SELLA application was developed in Python, leveraging its rich ecosystem of libraries for rapid and robust development. Python was specifically selected due to its seamless integration with Raspberry Pi hardware and its extensive support for data processing tasks. Furthermore, this choice facilitates future scalability and ensures the code base remains easy to maintain and update.

2.6.1. GUI development with PyQt5

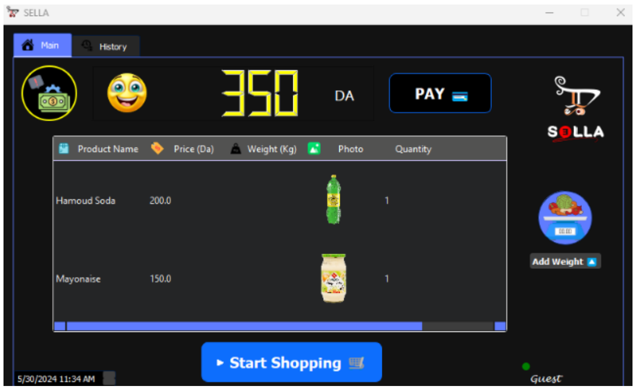
The graphical user interface was built using the PyQt5 framework. This choice enabled the creation of a modern, responsive, and event-driven application. Core application windows and their functionalities were implemented as distinct classes. For example, user interactions are handled by connecting widget signals (like button.clicked) to custom Python methods (slots), such as self.login_button.clicked.connect (self.handle _login).

2.6.2. User interface workflow and features

The user interface workflow and its main features can be broken down as:

– Login and authentication: the application starts with a login window, allowing users to authenticate or proceed as a guest. Authentication logic queries a local SQLite database to verify credentials.

- Main shopping interface: this is the primary user dashboard. A QTableWidget displays scanned products with columns for name, price, photo, and quantity. Item data is populated dynamically using tableWidget.setItem(). A large display shows the real-time total. The screenshot of the main window during shopping is shown in Figure 6.
- Automated payment process: the payment window is triggered by the user. It displays the final amount and generates a unique QR code by embedding a payment URL (e.g., https://.../sella?total_price=370.0) into a QR image. A "save transaction" button, connected via self.save_button.clicked.connect (self.saveTransaction), commits the purchase details to the database.



Product Name	Price (Da)	Weight (Kg)	Photo	Quantity
Hamoud Soda	200.0			1
Mayonaise	150.0			1

Figure 6. Screenshot of the main window during shopping

A lightweight SQLite database is used to store product information (product ID, name, price, and image URL) and user transaction history (receipt no, timestamp, customer, and total price). The history table is initialized with defined columns and populated upon successful transactions, accessible to logged-in users.

3. EXPERIMENTAL SETUP AND PERFORMANCE EVALUATION

To scientifically validate the SELLA system, a series of controlled experiments were designed and executed with detailed methodology to ensure full reproducibility. These experiments focused on monitoring critical performance metrics under varying environmental conditions and usage scenarios. The primary objective was to stress-test the hardware and software integration to ensure reliability in a real-world retail setting.

3.1. Testbed and environment

Our experimental setup is characterized by the following key aspects:

- Hardware: the fully assembled prototype as described in section 2, with all components precisely documented including model numbers, firmware versions, and configuration settings.
- Test items: a set of 20 distinct retail items were used, including plastic containers, cardboard boxes, and clothing items. Each was affixed with an EPC Gen2 UHF tag. A known RF-interfering item (an aluminum foil-wrapped box) was also used. All items were systematically cataloged with dimensions, materials, and tag placement locations.
- Environment: tests were conducted in a standard indoor lab environment (22 °C, 45% humidity), simulating a retail aisle. Environmental RF noise was measured at -85 dBm using a spectrum analyzer.

3.2. Performance metrics

To provide a quantitative assessment, the following key performance indicators (KPIs) were defined:

- Tag detection accuracy (%): the percentage of correctly identified tags out of the total number of tags placed in the trolley. Accuracy = (correct detections / total tags) × 100.
- Read range (cm): the maximum distance from the antenna at which a tag is reliably detected (>99% read rate).

- UI update latency (ms): the time elapsed between a tag being physically placed in the cart and the corresponding visual update on the GUI. This was measured using high-speed video capture (120 fps) and frame analysis.
- System resource utilization (%): the average CPU and RAM usage of the Raspberry Pi during active shopping simulation, measured using the htop command-line utility with sampling every 5 sec.

3.3. Experimental procedure

For each test scenario, 50 trials were conducted to ensure statistical significance, included:

- Scenario A (baseline): adding 10 non-interfering items one by one (each 30-second).
- Scenario B (multi-tag): adding all 20 items simultaneously in a randomized arrangement.
- Scenario C (interference): adding 10 items along with the aluminum-wrapped box placed directly over the antenna.
- Scenario D (dynamic movement): simulating realistic shopping conditions by continuously moving items in and out of the cart at varying speeds.
- Scenario E (mixed materials): testing with items containing liquids (water bottles), metals (canned goods), and electronics to evaluate performance with challenging materials.

Each trial followed a standardized protocol: 1) system initialization and calibration, 2) baseline noise measurement, 3) item placement according to scenario specifications, 4) data collection for 60 seconds, and 5) system reset and validation between trials. Raw data including timestamps and tag IDs were logged.

3.4. Pilot usability evaluation

To validate the user-friendly design, a pilot usability study was conducted with five representative users (ages 25-50). This heuristic evaluation required participants to perform predefined tasks (log in, add/remove 5 units, show total, and process mock payment), while using the “think aloud” protocol. Qualitative feedback was recorded, and the session concluded with a quantitative system usability scale (SUS) questionnaire.

4. RESULTS AND DISCUSSION

4.1. RFID subsystem performance

The RFID reader’s performance is critical. Our tests yielded the results summarized in Table 1. The system demonstrated excellent accuracy (>98%) under normal conditions. As hypothesized, the presence of metallic material (Scenario C) and mixed materials including liquids (Scenario E) significantly degraded performance. Dynamic movement (Scenario D) also introduced a moderate decrease in accuracy. These results confirm that while the system is robust, optimal performance requires either user guidance on item placement or future hardware enhancements like multi-antenna arrays.

Table 1. RFID performance metrics under different scenarios

Metric/scenario	Scenario A (baseline)	Scenario B (multi-tag)	Scenario C (interference)	Scenario D (dynamic movement)	Scenario E (mixed materials)
Detection accuracy	99.6%	98.2%	78.4%	94.7%	89.3%
Mean read time	150 ms	210 ms	350 ms	245 ms	280 ms
Max read range	55 cm	52 cm	30 cm	48 cm	42 cm
Std. deviation	±2.1%	±3.4%	±8.7%	±5.2%	±6.8%

Common RFID pitfalls observed during testing included tag shadowing (where one tag blocks another), signal absorption by liquid-containing items, and interference from metal surfaces. These effects are consistent with findings from recent research by Dobkin [27] and Griffin *et al.* [28], who have extensively documented RF interference challenges in retail environments.

4.2. System-level performance

- System-level metrics are crucial for assessing the real-world user experience and long-term stability.
- UI update latency: the mean latency was measured to be 485 ms, with a standard deviation of 35 ms (minimum: 420 ms, maximum: 580 ms). This sub-second response time is perceived by users as instantaneous, providing a fluid experience.
- Resource utilization: during active use (Scenario B), the system’s average CPU utilization was 28% (peak: 45%), and RAM usage was 310 MB (peak: 380 MB). These low figures demonstrate the efficiency of the software architecture.

4.3. User-centric evaluation

The pilot usability study yielded promising qualitative results. The five participants all performed the tasks set without assistance. The feedback invariably emphasized the “intuitive” feel of the central shopping page and the “clear and simple” payment process. The average SUS score was 85.5, in the usability “excellent” grade. There was one very small area for improvement identified: two participants suggested that the confirmation of the “remove item” can be made more explicit when an item is removed from the cart. The feedback provides an empirical basis for the claim of having made the design “user-friendly” and identifies a clear future direction for refinement of the user interface.

4.4. Contribution and comparison with prior art

The novelty of the SELLA system is best illustrated by comparing it against prior work on key technical and validation dimensions, as shown in Table 2. This comparison clearly highlights SELLA’s contribution. It is important to note that a direct quantitative performance comparison with commercial systems like the Amazon Dash Cart [29] is challenging, as their detailed performance metrics and operational data are proprietary.

Table 2. Comparative analysis of smart trolley systems

Feature	Yusoff *et al.* [8]	Pradhan *et al.* [10]	Amazon dash cart [29]	SELLA (this work)
Processor	Arduino	Raspberry Pi	Proprietary	Raspberry Pi 4
Checkout process	Centralized	Not specified	Autonomous	Autonomous (QR code)
GUI	Basic LCD	Not specified	Advanced	Advanced (PyQt)
Performance validation	Conceptual	Basic	Limited	Rigorous and quantitative
UI latency reported	No	No	No	Yes (<500 ms)
Resource usage reported	No	No	No	Yes (CPU, RAM)
Payment integration	No	No	Yes	Yes
Anti-theft mechanism	No	No	Weight + vision	RFID debouncing
Cost estimation	Low	Medium	High	Medium

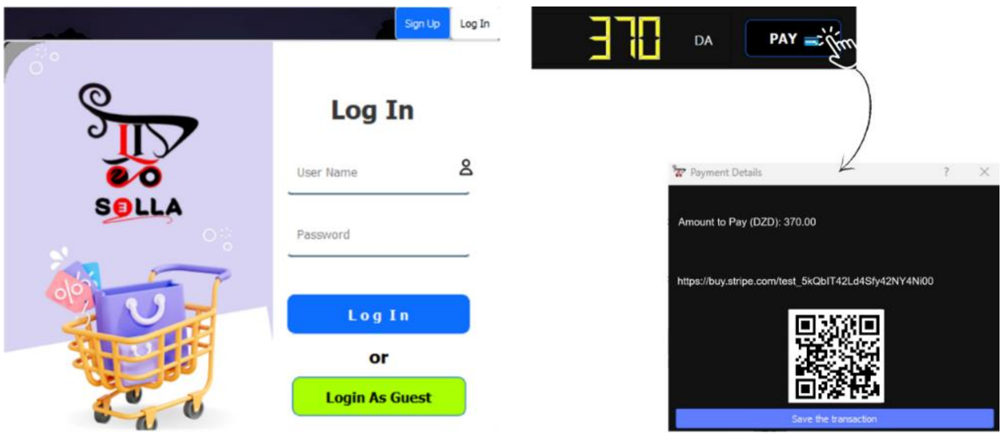
However, SELLA provides a more transparent and cost-effective approach using open-source technologies. Recent industry advances, as highlighted in systematic reviews of IoT technologies for retail [30], emphasize the growing importance of seamless checkout experiences. SELLA’s approach aligns with these trends while offering a more accessible implementation.

4.5. Security and privacy considerations

In order to provide a reliable user experience, the SELLA solution deploys a multi-layered security and privacy architecture. The most predominant measures intend to safeguard user information, make payment transactions secure, and overcome privacy issues inherent in RFID technology.

– User authentication: the system secures access through a mandatory authentication process. A robust login interface (Figure 7(a)) requires a username and password to verify user identity before granting access to the main application and personal data, such as transaction history. This serves as the primary gateway for preventing unauthorized account access.
– Secure payment processing: financial transactions are secured with a dynamic, one-time-only QR code per payment (Figure 7(b)). This is a naturally secure mechanism that does not involve storing or transmitting vulnerable credit card information on the device itself, thereby reducing the risk of breaches or physical card skimming. The transaction is processed by a secure, external payment processor, in line with current-day e-commerce security practices.
– Data protection: sensitive user information, in particular, transaction history, is all encrypted before being written to the local SQLite database. The software automatically deletes any temporary payment information following a successful transaction so that no sensitive information is maintained longer than necessary.
– Tag privacy: to safeguard shopper privacy, the system is configured to read non-personal electronic product codes (EPCs) of products only. No personally identifiable or personal information resides or is transmitted by the RFID tags. This design feature is consistent with privacy-preserving tendencies in IoT systems and complies with the concerns surrounding RFID technology [31].

These together form an end-to-end security architecture that not only secures user information, but payment transactions too, addressing the most salient security concerns evident in prevailing IoT retail implementations [32].



Amount to Pay (DZD): 370.00

https://buy.stripe.com/test_5kQbIT42Ld4Sfy42NY4Ni00

(a) (b)

Figure 7. SELLA security interface components: (a) user authentication and (b) secure QR code payment

5. CONCLUSION

This paper presented SELLA, a fully autonomous IoT-based smart trolley integrating UHF RFID tracking, a responsive touchscreen interface, and automated on-board payment. Comprehensive testing demonstrated high detection accuracy (98.2%), low UI latency (<500 ms), and efficient resource usage, addressing the key shortcomings of prior work. By coupling robust hardware design with a stability-focused software architecture, SELLA delivers a validated, user-friendly solution for queue-free retail. Future work will explore multi-antenna configurations to mitigate interference, integration with retailer inventory systems, and large-scale deployment trials to assess scalability in operational environments.

The system represents a significant advancement in smart retail technology, offering a scientifically validated solution that addresses both customer experience and operational efficiency. As retailers continue to seek innovations that blend physical and digital shopping experiences, systems like SELLA provide a practical foundation for the next generation of retail environments.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Hadj Zerrouki	✓	✓	✓	✓	✓	✓		✓		✓	✓	✓	✓	
Salima Azzaz-Rahmani		✓			✓	✓	✓	✓	✓	✓	✓			✓

C : **C**onceptualization	I : **I**nvestigation	Vi : **Vi**sualization
M : **M**ethodology	R : **R**esources	Su : **Su**pervision
So : **So**ftware	D : **D**ata Curation	P : **P**roject administration
Va : **Va**lidation	O : Writing - **O**riginal Draft	Fu : **Fu**nding acquisition
Fo : **Fo**rmal analysis	E : Writing - Review & **E**diting	

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.

REFERENCES

[1] M. R. Jacob, N. M, and D. R, “Smart Cart-Assisted In-Store Shopping,” *SSRN Electronic Journal*, 2024, doi: 10.2139/ssrn.4927047.

[2] N. Wagh, T. Waghe, R. Sahare, P. Diyawar, A. Ninawe, and K. Bogawar, “Smart line following shopping cart with billing system using RFID,” *SSRN*, Elsevier BV, 2025, doi: 10.2139/ssrn.5197671.

[3] R. Singh, K. N. Rao, R. Naik, Geetha, K. Anjali, and P. Vineeth, “Smart Trolley Using Automated Billing Interface,” *2022 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC)*, 2022, doi: 10.1109/assic55218.2022.10088393.

[4] Swetha K. B., Abhishek G., Ruthvik T, Meghana B. N., and N. R. G. Naresh, “Design and Implementation of IoT-Based Smart Shopping Dash Cart,” *International Journal of Engineering Technology and Management Sciences*, pp. 5–14, 2021, doi: 10.46647/ijetms.2021.v05i04.002.

[5] S. Saxena and T. Dhote, “Leveraging IoT Technologies in Retail Industry to improve Customer Experience: Current Applications and Future Potential,” *2023 Somaiya International Conference on Technology and Information Management (SICTIM)*, pp. 50–54, 2023, doi: 10.1109/sictim56495.2023.10104882.

[6] R. Li, T. Song, N. Capurso, J. Yu, J. Couture, and X. Cheng, “IoT Applications on Secure Smart Shopping System,” *Internet of Things Journal*, vol. 4, no. 6, pp. 1945–1954, 2017, doi: 10.1109/jiot.2017.2706698.

[7] S. Mekruksavanich, “Supermarket Shopping System using RFID as the IoT Application,” *2020 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT &amp; NCON)*, pp. 83–86, 2020, doi: 10.1109/ectidamtncon48261.2020.9090714.

[8] Z. M. Yusoff, A. M. Markom, N. D. K. Ashar, and Z. Muhammad, “Development of a Smart Scanner Trolley System for Intelligent Billing Solutions using RFID and IoT System,” *International Journal of Emerging Trends in Engineering Research*, vol. 8, no. 9, pp. 6220–6225, Sep. 2020, doi: 10.30534/ijeter/2020/212892020.

[9] E. Jothi, R. Thenmozhi, and S. Vidhyalaxmi, “IoT-Enabled Smart Shopping Trolley with Automated Billing and Real Time Cost Tracking,” *International Conference on Modern Trends in Engineering and Management (ICMTEM-25)*, 2025, doi: 10.59544/vnqo7016/icmtem25p11.

[10] D. D. Pradhan, S. Mali, A. Ubale, M. M. Sardeshmukh, S. Pattnaik, and P. B. Sindhe, “Smart Shopping Trolley Using Raspberry Pi,” *2021 International Conference in Advances in Power, Signal, and Information Technology (APSIT)*, pp. 1–4, 2021, doi: 10.1109/apsit52773.2021.9641206.

[11] M. N. Khan,Tanvirahmedshuvo, M. R. H. Ontor, N. Khan, and A. Rahman, “The Internet of Things (IoT): Applications, Investments, and Challenges for Enterprises,” *International Journal For Multidisciplinary Research*, vol. 6, no. 1, 2024, doi: 10.36948/ijfmr.2024.v06i01.22699.

[12] P. Chandrasekar and T. Sangeetha, “Smart shopping cart with automatic billing system through RFID and ZigBee,” *International Conference on Information Communication and Embedded Systems (ICICES2014)*, pp. 1–4, 2014, doi: 10.1109/icices.2014.7033996.

[13] M. Shahroz, M. F. Mushtaq, M. Ahmad, S. Ullah, A. Mehmood, and G. S. Choi, “IoT-Based Smart Shopping Cart Using Radio Frequency Identification,” *IEEE Access*, vol. 8, pp. 68426–68438, 2020, doi: 10.1109/access.2020.2986681.

[14] M. Chiranjivi, A. D. Rao, B. C, Rao, R. N. Raj, and S. Hukesh, “A Novel Technology for Smart shopping trolley system,” *E3S Web of Conferences*, vol. 547, p. 2018, 2024, doi: 10.1051/e3sconf/202454702018.

[15] S. R. Subudhi and R. N. Ponnalagu, “An Intelligent Shopping Cart with Automatic Product Detection and Secure Payment System,” *2019 IEEE 16th India Council International Conference (INDICON)*, pp. 1–4, 2019, doi: 10.1109/indicon47234.2019.9030331.

[16] V. L. S. Devi, S. S. Ahmed, S. A. Khadri, and N. Baig, “IoT-based smart cart with automatic billing and anti-theft,” *Ciência & Engenharia – Science & Engineering Journal*, vol. 11, no. 1, pp. 1658–1666, Feb. 2023, doi: 10.52783/cienceng.v11i1.317.

[17] A. Y. W. Setyanto, F. Ahmad, J. B. Wicaksono, and K. Mutijarsa, “Smart Trolley with Position Localization Method Based on QR Code Mapping using Computer Vision and Internet of Things,” *2022 International Conference on Information Technology Systems and Innovation (ICITSI)*, pp. 346–352, 2022, doi: 10.1109/icitsi56531.2022.9970792.

[18] H.-C. Chi, M. A. Sarwar, Y.-A. Daraghmi, K.-W. Lin, T.-U. Ik, and Y.-L. Li, “Smart Self-Checkout Carts Based on Deep Learning for Shopping Activity Recognition,” *2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pp. 185–190, 2020, doi: 10.23919/apnoms50412.2020.9237053.

[19] M. A. Sarwar, Y.-A. Daraghmi, K.-W. Liu, H.-C. Chi, T.-U. Ik, and Y.-L. Li, “Smart Shopping Carts Based on Mobile Computing and Deep Learning Cloud Services,” *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, 2020, doi: 10.1109/wcnc45663.2020.9120574.

[20] S. R. Patel, “Multi-Modal product recognition in retail environments: Enhancing accuracy through integrated vision and OCR approaches,” *World Journal of Advanced Research and Reviews*, vol. 25, no. 1, pp. 1837–1844, 2025, doi: 10.30574/wjarr.2025.25.1.0122.

[21] B. Santra and D. P. Mukherjee, “A comprehensive survey on computer vision based approaches for automatic identification of products in retail store,” *Image and Vision Computing*, vol. 86, pp. 45–63, 2019, doi: 10.1016/j.imavis.2019.03.005.

[22] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, “Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications,” *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015, doi: 10.1109/comst.2015.2444095.

[23] A. Botta, W. de Donato, V. Persico, and A. Pescapé, “Integration of Cloud computing and Internet of Things: A survey,” *Future Generation Computer Systems*, vol. 56, pp. 684–700, 2016, doi: 10.1016/j.future.2015.09.021.

[24] R. R. Vallabhuni, S. Lakshmanachari, G. Avanthi, and V. Vijay, “Smart Cart Shopping System with an RFID Interface for Human Assistance,” *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*, pp. 165–169, 2020, doi: 10.1109/iciss49785.2020.9316102.

[25] T. M, R. Sudarmani, D. N, and S. Uammaheswari, “IoT based Smart Trolley with Integrated RFID reader and Advanced Billing System,” *2025 3rd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA)*, pp. 1–5, 2025, doi: 10.1109/icaeca63854.2025.11012255.

[26] S. Pokhrel, G. Pradhan, M. Khati, A. Rai, and S. R. Luitel, “A Smart Shopping Cart for Automated Billing and Personalized Recommendations,” *International Journal on Engineering Technology*, vol. 2, no. 2, pp. 104–113, 2025, doi: 10.3126/injet.v2i2.78598.

[27] D. M. Dobkin, “UHF RFID Protocols,” *The RF in RFID*. Elsevier, pp. 361–451, 2013, doi: 10.1016/b978-0-12-394583-9.00008-9.

[28] J. D. Griffin, G. D. Durgin, A. Haldi, and B. Kippelen, “RF Tag Antenna Performance on Various Materials Using Radio Link Budgets,” *IEEE Antennas and Wireless Propagation Letters*, vol. 5, pp. 247–250, 2006, doi: 10.1109/lawp.2006.874072.
[29] D. Kumar, “The next evolution of the Dash Cart: New features and expansion to first Whole Foods Market Store,” *Https://Www.Aboutamazon.Com/News/Retail/Amazon-Dash-Cart-New-Features-Whole-Foods*, 2022 (accessed: Jun, 10, 2025).
[30] A. T. Rosário and R. J. Raimundo, “The Integration of AI and IoT in Marketing: A Systematic Literature Review,” *Electronics*, vol. 14, no. 9, p. 1854, 2025, doi: 10.3390/electronics14091854.
[31] S. L. Garfinkel, A. Juels, and R. Pappu, “RFID Privacy: An Overview of Problems and Proposed Solutions,” *IEEE Security and Privacy Magazine*, vol. 3, no. 3, pp. 34–43, 2005, doi: 10.1109/msp.2005.78.
[32] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, “A Survey on Homomorphic Encryption Schemes,” *ACM Computing Surveys*, vol. 51, no. 4, pp. 1–35, 2018, doi: 10.1145/3214303.

BIOGRAPHIES OF AUTHORS

Hadj Zerrouki is a Lecturer of Telecommunication Systems and Networks at Department of Telecommunications, Faculty of Electrical Engineering, Djilali Liabes University of Sidi Bel Abbes, Algeria. He Holds a Ph.D. degree in Telecommunication with specialization in Systems and Networks. He received the Engineer’s degree in Telecommunication and the M.Sc. degree in Systemes and Networks of Telecommunication from Abou Bakr Belkaid University of Tlemcen, Algeria. His research areas are signal processing, MIMO systems, wireless and mobile communications, networking and antennas in wireless telemetry applications. He has been an integral part of the academic community at the University of Tlemcen since 2010, where he balances his responsibilities between teaching and cutting-edge research as an active member of the STIC Laboratory. He can be contacted at email: hadj.zerrouki@univ-sba.dz.

Salima Azzaz-Rahmani is an Associate Professor, Department of Telecommunications, Faculty of Electrical Engineering, University of Djillali Liabes, Sidi Bel Abbès, Algeria. With a complete academic path accomplished at the University of Tlemcen-Algeria, she earned her State Engineer degree in Telecommunications (2003), a Magister (2006), and a Ph.D. degree in Telecommunications (2013). She further solidified her senior academic standing by obtaining her University Habilitation in 2021. Her research is centered on the cutting-edge field of antenna design and microwave engineering. Her work specializes in the conception, modeling, and optimization of microstrip antennas for a wide array of modern applications. She has developed significant expertise in designing miniature and compact antennas for ultra-wideband (UWB) systems, WLAN, and wireless medical telemetry applications. She can be contacted at email: salima.azzazrahmani@univ-sba.dz.