iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** XI **Month of publication:** November 2026

DOI: https://doi.org/10.22214/ijraset.2026.85555

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue XI Nov 2026- Available at www.ijraset.com

AI-Based Smart Self-Checkout Autonomous Shopping Trolley using IoT

Prof. P. A. Kale[1], Shubhangi Balate[2], Priya Gangawane[3], Nikita Shirsath[4]
Department of Electronics and Telecommunications Engineering, Shri Chhatrapati Shivaji Maharaj College of Engineering, Nepti, Ahilyanagar, Maharashtra, India

Abstract: ***Conventional supermarket checkout systems generally require products to be identified and billed at a designated point-of-sale counter. During busy shopping periods, customers may need to wait for checkout and may not have a convenient way to monitor the total cost while selecting products. A smart shopping trolley can move some product-recording and bill-calculation tasks closer to the shopper. Earlier smart-cart research has explored automatic billing, product information and anti-theft functions using RFID and wireless communication [1].***
Keywords: Smart shopping trolley, QR code, automatic billing, load cell, weight verification, exit-gate security.

I. INTRODUCTION

Retail checkout is an important part of the shopping experience because it affects the time required to complete a purchase and the accuracy of the final bill. In a conventional process, products are generally scanned after the customer has finished shopping. A trolley that records products during selection could provide a running bill and a visible list of scanned items. QR codes are a practical candidate for product identification because they encode machine-readable information that can be decoded by a compatible scanner or camera [2]. In this proposal, the code carries a product identifier, while the product name, price and reference weight are maintained in a product database.

This paper proposes a smart shopping trolley that integrates QR-code-based product identification, automatic bill calculation, load-cell-based weight verification and an exit-gate checking procedure. A microcontroller coordinates the scanner, display, weighing circuit and verification logic. The weight check compares the expected weight of recorded items with the measured trolley load, using a tolerance determined through calibration. The check is intended to identify a discrepancy for review; it cannot independently establish why a mismatch occurred. The system is a proposed educational prototype, and its performance must be established through implementation and controlled testing. The scope is limited to product identification, bill calculation, trolley-load measurement and a prototype-level exit check. The design does not claim to replace a retailer's complete point-of-sale or payment infrastructure. Product prices and reference weights must be maintained accurately, and the final paper must identify which features are physically implemented and which are simulated.

II. PROBLEM STATEMENT

The main problem addressed is the absence of a single trolley-level process that maintains a running bill and compares recorded products with a physical-load measurement. A QR-based system can associate each scan with a product record, while a load cell can provide an independent measurement of the trolley load. However, measured and expected weights may differ because of packaging variation, products being removed, inaccurate database entries, trolley movement, mechanical loading or sensor drift. Weight verification should therefore be treated as a secondary discrepancy indicator, and any alert should lead to manual inspection rather than an automatic accusation.

III. OBJECTIVES

- Identify products by scanning QR codes linked to a product database.
- Calculate and display item-wise prices and the running total automatically.
- Measure the trolley load using a load cell and HX711 interface.
- Compare the expected item-weight sum with the measured load using a configurable tolerance.
- Provide exit-gate verification and raise an alert when records do not match.
- Keep the design modular so that it can be tested within a student-project budget.

©IJRASET: All Rights are Reserved

The objectives can be addressed through a staged implementation: establish a unique product-code format; create a product table containing identifier, name, price and reference weight; implement a running bill and item ledger; calibrate the load-cell subsystem; define discrepancy and exception rules; and evaluate the integrated workflow with repeatable test cases. This staged approach helps distinguish software errors from sensor and mechanical errors.

IV. RELATED WORK

Smart-cart research has investigated RFID-based identification, wireless communication, automatic billing, product information and anti-theft checking. Karpakavalli et al. described a smart cart combining RFID and ZigBee with automatic billing and an exit anti-theft reader [1]. RFID and QR codes use different identification approaches: RFID requires compatible tags and readers, whereas QR-based identification can use printed codes and a compatible optical scanner. QR codes are a possible low-cost option for an educational prototype, although scanning depends on code visibility, print quality, lighting and reader compatibility [2]. The present proposal combines QR scanning with load-cell-based weight comparison. This is a design-level distinction only; claims about superior accuracy, speed or cost require a controlled comparison.

V. PROPOSED SYSTEM ARCHITECTURE

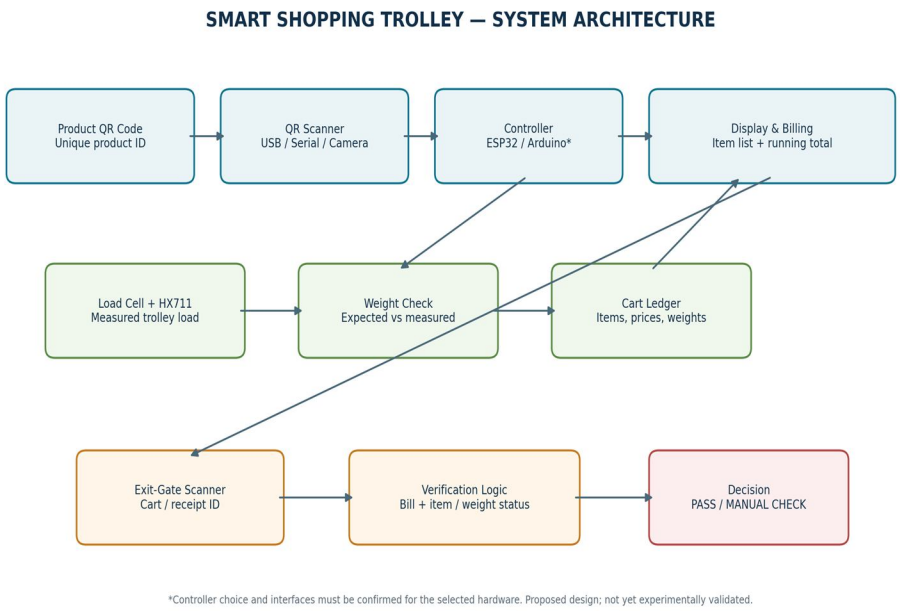


Fig. 1. Proposed architecture of the smart shopping trolley.

The architecture is organized into product identification, control and billing, weight measurement, user display, product data storage and exit verification. When a code is scanned, the scanner sends the decoded identifier to the controller. The controller validates the identifier and retrieves the product name, price and reference weight from the product database. It updates the cart ledger and calculates the running total. The load cell converts mechanical load into an electrical signal, while the HX711 interface conditions and digitizes the sensor signal for the controller [3]. Once a stable reading is available, the controller compares the measured load with the expected total weight and checks whether the difference is within the calibrated tolerance. At the exit, a trolley or receipt identifier can be checked against the bill and verification state. A mismatch should generate a manual-check alert rather than an automatic accusation.

A key design requirement is that each module communicate through a defined interface. The scanner should provide a decoded product identifier, while the database remains the source for price and reference weight. The controller should maintain the cart state and reject invalid records. The display should distinguish a normal scan from an error or pending manual check. This separation makes individual modules easier to debug and replace.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

VI. HARDWARE AND SOFTWARE REQUIREMENTS

Component	Purpose	Selection notes
Microcontroller (ESP32 or Arduino)	Coordinates scanning, display, weighing and verification	Choose after checking scanner interface, available ports and power requirements.
QR-code scanner / compatible camera	Reads product identifier	A USB/serial scanner is simplest if its interface matches the controller; camera-based decoding needs suitable software.
Load cell + HX711 module	Measures trolley load	Requires mechanical mounting, calibration and tare handling.
LCD/OLED display	Shows item name, price and running total	Select interface compatible with controller.
Buzzer / indicator LED	Signals mismatch or scan errors	Use a clear alert state and manual override procedure.
Exit-gate scanner or reader	Reads trolley/receipt identifier	Prototype may use a second QR scan; real gate integration requires a defined ID format.
Power supply, wiring and frame	Powers and mounts the prototype	Use regulated power and protect wiring from load and movement.
Product database / firmware	Stores SKU, price and reference weight; computes bill	Can begin with a local table; validate invalid and duplicate scans.

Table 1. Proposed component groups. Exact part numbers and costs must be confirmed against local supplier quotations before the final bill of materials is reported.

A. Illustrative Component Images

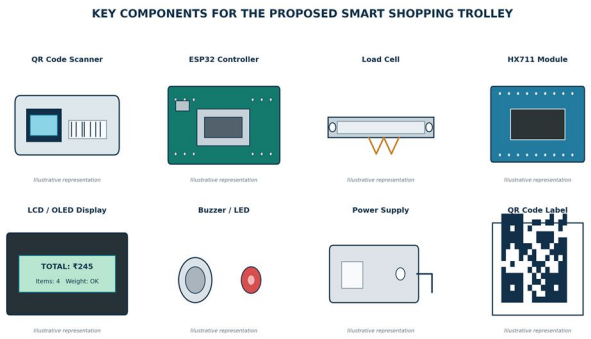


Fig. 2. Illustrative representations of the key hardware elements.

The controller should provide suitable interfaces for the selected scanner, display, HX711 module and alert device. Selection between ESP32, Arduino or another board should follow checks of voltage levels, serial/USB support, memory and power requirements. A camera-based QR decoder may require more processing and software than a scanner that outputs decoded text. The final paper should identify the exact hardware actually used.

VII. METHODOLOGY

The methodology follows a modular build-and-test process. First, the hardware interfaces and product data format are selected. Next, the QR scanner and controller are integrated, and product records are created with unique identifiers, prices and reference weights. The load cell is mounted securely and calibrated using known reference loads. Billing and weight-check functions are implemented separately before integration with the exit-verification workflow. Finally, the prototype is evaluated using predefined test cases. Test records should include the date, hardware and firmware versions, product or load used, test conditions, expected behaviour, observed behaviour and corrective action. The following steps describe the proposed operating sequence; they do not imply that the prototype has already been built or validated.

A. Proposed Operating Workflow

1) Initialization: switch on the controller, initialize product records, allow the weighing circuit to stabilize, tare the empty trolley and check scanner and display communication. If the sensor reading is unstable, request another reading instead of starting verification.

2) Product scanning: scan the product QR code and validate the decoded identifier. If the code is unknown, malformed or unreadable, do not add an item to the bill; display an error and allow another scan or staff review.

3) Billing and cart ledger: retrieve the product name, unit price and reference weight. Add the item to the cart ledger and update the running total. The software should define how duplicate scans are handled and how an item is removed from the ledger if the shopper changes their selection.

4) Weight verification: obtain a stable load-cell reading and compare the change in measured load with the expected weight of the scanned item or the expected sum for the cart. Apply a tolerance established through calibration and repeated trials. Do not assume one tolerance is appropriate for every product or trolley condition.

5) Exit verification: scan the trolley or receipt identifier and compare it with the current cart record. Where payment integration is not implemented, clearly identify exit verification as a simulated or manual step; do not imply that payment has been confirmed.

6) Exception handling: unknown codes, missing reference weights, unstable measurements, duplicate records and weight mismatches should produce understandable messages. Staff review or a manual override should be available for legitimate exceptions.

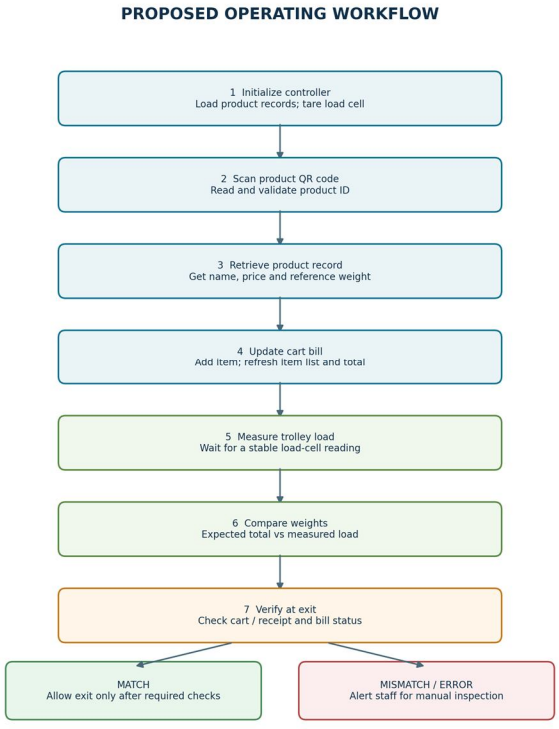


Fig. 3. Proposed operating workflow.

The software can be divided into scanner input handling, product lookup, cart-ledger management, total calculation, sensor acquisition, discrepancy checking, display updates and exit-state management. A simple state model may use READY, ITEM_SCANNED, WEIGHT_CHECK, REVIEW_REQUIRED and EXIT_CHECK. State transitions should occur only when the required input is valid, reducing the chance that a failed scan changes the bill.

VIII. VERIFICATION LOGIC

For a cart containing n recorded products, the expected weight can be estimated as $W_expected = \Sigma w_i$, where w_i is the stored reference weight of item i. The measured trolley load after tare is W_measured. The absolute discrepancy is $\Delta W = |W_measured - W_expected|$. A review alert may be raised when $\Delta W > \tau$, where τ is a tolerance selected from calibration and test data. The software must define whether it checks individual weight changes, total cart weight, or both. Sensor drift, tare error, package variation, products being added or removed, movement and mechanical mounting can affect the result. Accordingly, τ should be determined experimentally and documented; this equation is a proposed screening rule, not evidence of detection accuracy.

For repeatability, conduct tests on a stable surface and keep the trolley stationary during weight readings unless movement is itself being evaluated. Tare the empty trolley before each test series and check the reference loads. Record failures as well as successes. If software changes during testing, record the firmware version and affected test cases so that results from different versions are not mixed.

IX. PLANNED TESTING AND EVALUATION

Testing is planned for after the hardware and software have been assembled. Each test should be repeated sufficiently to assess consistency, and raw observations should be retained. The result cells below are intentionally left blank for actual measurements. Before testing, define the expected behaviour and acceptance criteria for each test; after testing, record both successful and failed trials. Do not enter estimated values or treat a simulated test as a physical prototype result.

Test	Procedure	Metric to record	Result
QR recognition	Scan valid, invalid and damaged/poorly printed codes	Successful scans / total attempts	[Fill after test]
Bill calculation	Scan a known list of products	Calculated total vs manually calculated total	[Fill after test]
Weight measurement	Measure known reference loads repeatedly	Mean error and repeatability	[Fill after test]
Mismatch detection	Introduce a controlled difference between scanned items and load	True alerts, missed alerts, false alerts	[Fill after test]
Exit verification	Try matched and mismatched cart/receipt records	Correct pass/alert decisions	[Fill after test]
Endurance	Run repeated scan-and-weigh cycles	Errors, resets, battery/power issues	[Fill after test]

Table 2. Test plan and fields to complete with observed data.

X. LIMITATIONS

The design has technical and operational limitations. QR scanning may fail when a code is damaged, dirty, obscured, poorly printed or poorly illuminated. Product records may be incorrect or outdated, and duplicate scans or unrecorded removals can affect the bill. A load cell requires rigid mounting, stable power, tare handling and calibration; readings can drift or vary with item position and trolley movement. A weight mismatch can arise for legitimate reasons and cannot by itself prove theft. Exit verification depends on a consistent trolley or receipt identifier and integration with bill or payment status. Cost, usability and performance remain unknown until parts are selected and practical testing is completed.

XI. EXPECTED BENEFITS

The proposed system is expected to offer several practical benefits: shoppers can view the running total, the cart ledger can maintain an item-wise record, and the weight check can provide an additional reason to review a discrepancy before exit. Product scanning may also make billing more structured and reduce manual price addition. These are anticipated benefits based on the proposed functions, not measured outcomes. Their value will depend on scan reliability, ease of use, response time, weight calibration, database quality and exit-check implementation. Practical trials are required before making claims about time savings, billing accuracy or loss prevention.

XII. FUTURE SCOPE

Future work may include a more robust QR-scanning interface, product-database synchronization, a clearer item-removal workflow and automatic recovery from scanner or sensor faults. Multiple load cells or improved mechanical mounting may be investigated if initial measurements are not repeatable. A wireless dashboard could be added provided that communication is secured and unnecessary shopper data is not exposed. Integration with retailer inventory, payment and exit-gate systems would require defined interfaces, access controls and dedicated testing. Future evaluation should compare the prototype with conventional checkout using measured scan success, billing error, weight error, completion time and false-alert rate.

XIII. SECURITY, PRIVACY AND SAFETY CONSIDERATIONS

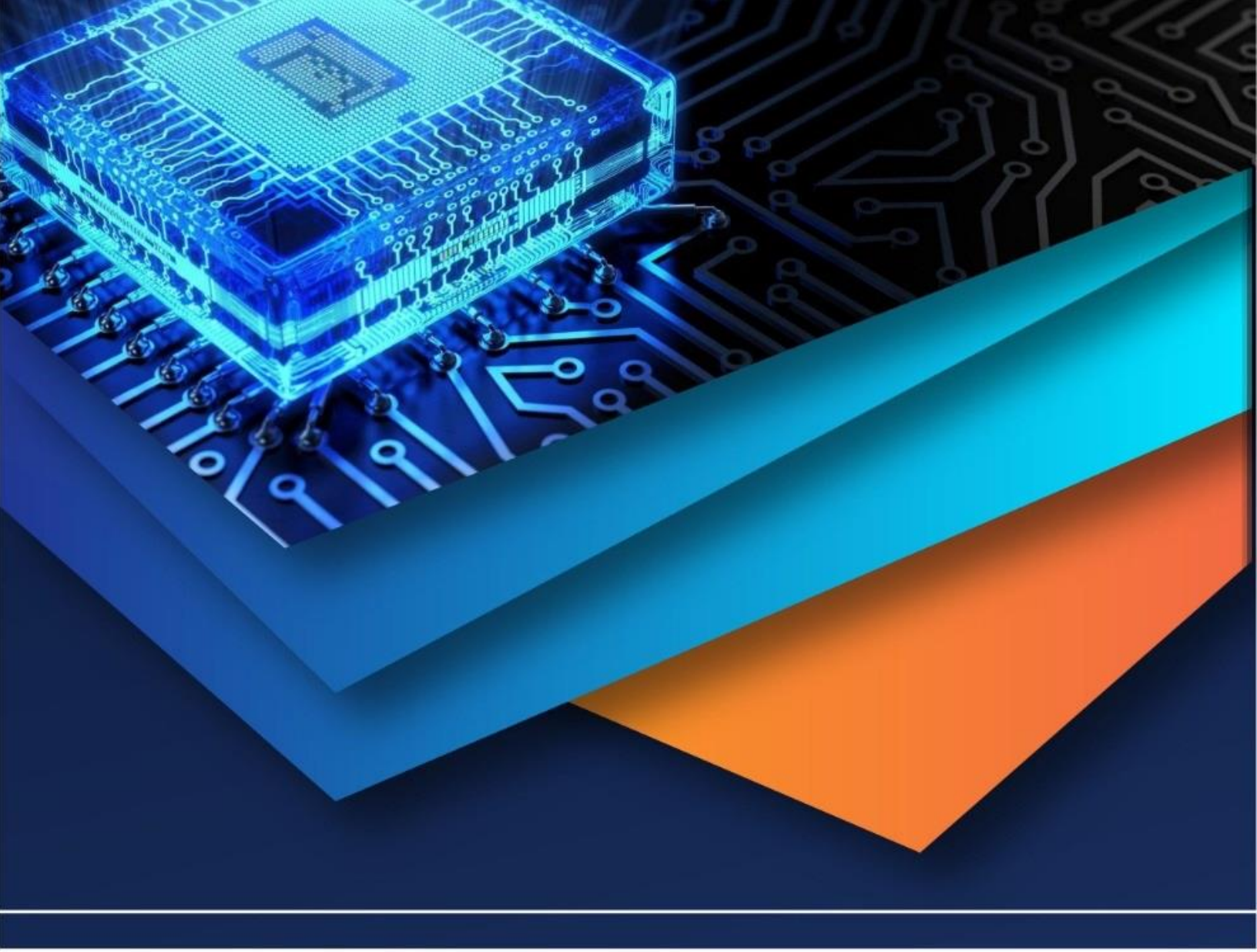
Security, privacy and physical safety should be considered throughout the design. Product identifiers should be validated before database lookup, and invalid input should not alter the bill. The cart ledger should be cleared or securely reset between shopping sessions. The prototype should avoid collecting unnecessary personal information. If real payment processing is later introduced, payment credentials should not be stored directly in the trolley; a suitable payment provider and secure integration would be required. Alerts should request staff review rather than automatically accuse or penalize a customer. Wiring, the power supply and load-cell mounting must be secured against movement, accidental disconnection and excessive mechanical load.

XIV. CONCLUSION

This paper presents a proposed smart shopping trolley integrating QR-code-based product identification, automatic bill calculation, weight verification and exit-gate checking. The modular architecture allows scanning, billing, weighing and exception handling to be implemented and tested independently before integration. Weight comparison provides a possible additional discrepancy check, but reliability depends on calibration, product data and operating conditions. Since the prototype is still under development, no measured accuracy, cost savings, reduced checkout time or security effectiveness is reported. These outcomes must be established using recorded experimental data. The next stage is to assemble the hardware, implement the software, calibrate the load cell and execute the planned test cases, including successful and failed trials.

REFERENCES

[1] S. Karpakavalli et al., “Smart Cart with Automatic Billing, Product Information, Product Recommendation Using RFID & Zigbee with Anti-Theft,” Procedia Computer Science, vol. 79, pp. 793–800, 2016. doi: 10.1016/j.procs.2016.03.107.
[2] DENSO WAVE, “What is a QR Code?” QR Code.com. Available: https://www.qrcode.com/en/about/
[3] Avia Semiconductor, “HX711: 24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales,” datasheet. Verify the exact module and datasheet revision used in the prototype before final submission.
[4] International Journal of Innovative Science and Research Technology (IJISRT), “For Authors /Author Guidelines.” Available: https://www.ijisrt.com/for-authors (accessed 18 October 2026).
[5] International Journal of Innovative Science and Research Technology (IJISRT), “Downloads: Manuscript Template and Copyright Agreement Form.” Available:https://www.ijisrt.com/download (accessed 18 October 2026).



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)