iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** X **Month of publication:** October 2026

DOI: https://doi.org/10.22214/ijraset.2026.85413

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 X Oct 2026- Available at www.ijraset.com

Smart Shopping Trolley Using RFID, ESP32 and IOT for Real-Time Bill Estimation

Chaithra A C[1], Ramyashree V[2], Rishitha D[3], Ruchitha CU[4], Dr. Hadimani Shivakumar[5]
Department of Electronics and Communication Engineering

Abstract: ***Conventional shopping often requires customers to wait until checkout to learn the full cost of selected products. This paper presents a proposed smart shopping trolley that combines radio-frequency identification (RFID), an ESP32 microcontroller, an I²C liquid-crystal display (LCD), a servo actuator and Wi-Fi connectivity. Each participating product is associated with a registered RFID tag. When a tag is scanned, the controller looks up its product record, updates the basket total and presents the item name and price on the display. The ESP32 can also send basket information to an IoT web interface for remote viewing. The paper describes the architecture, hardware roles, software sequence, expected benefits, limitations and a practical evaluation plan. The design is a prototype proposal; no measured performance is claimed until the system is assembled and tested.***

Keywords: Smart shopping trolley, RFID, ESP32, automatic billing, IoT, I²C LCD, embedded system, servo motor.

I. INTRODUCTION

Retail shopping combines product selection, price checking, basket management and payment. In a conventional process, prices are usually added at the checkout counter, so customers may not know their running expenditure while choosing products. Manual scanning and entry can also contribute to delays, especially when many customers arrive at the same time.

A smart shopping trolley places a small embedded system on the trolley so that item information can be recorded during shopping. In this proposed design, an RFID reader obtains the identifier stored in a product tag. The ESP32 checks that identifier against a product table and calculates the updated total. An LCD gives immediate feedback, while Wi-Fi can publish the basket summary to a web server. This approach combines identification, computation, display and connectivity in a single demonstrator.

The project is intended as an electronics and communication engineering prototype rather than a complete commercial checkout replacement. It focuses on a clear scan-and-display workflow and a basic IoT connection. Accurate operation depends on suitable tag placement, reliable product records, correct power regulation and careful handling of repeated scans.

II. PROBLEM STATEMENT AND OBJECTIVES

In a traditional store, customers often have limited visibility of the bill until all products reach the cashier. Repeated barcode scanning requires line-of-sight alignment and can lengthen checkout during busy periods. Customers may also exceed their planned budget without noticing the total during product selection. A compact trolley-mounted system can address part of this problem by recording product identifiers at the time of selection. The main objective is to design a prototype that identifies registered products, calculates an estimated bill and displays the current total. Supporting objectives are to provide clear feedback for invalid tags, transmit basket details through Wi-Fi, demonstrate an optional servo-controlled mechanism and define tests that can evaluate the design objectively.

III. RELATED WORK

Earlier smart-cart concepts have explored RFID-based identification, Bluetooth communication, wireless sensor networks, barcode scanning and image processing. RFID is attractive for a small prototype because the tag carries an identifier that can be mapped to product information in software. Unlike a printed barcode, the RFID tag does not need to be visually aligned with a camera, although its reading distance and reliability remain limited by the reader, tag, orientation and nearby materials.

Wireless communication has also been used to share cart data with a checkout station. The proposed trolley uses Wi-Fi available on the ESP32 to send a basket summary to a web interface. This reduces the need for a separate wireless module in a basic prototype. However, the design still needs a clear local operating mode when the network is unavailable.

The present proposal combines product identification, a local display and optional remote monitoring. Its contribution is an educational implementation path that connects common embedded-system functions in one prototype. It does not claim to introduce a new RFID technique or to replace the verification and security processes required in a commercial retail environment.

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

IV. PROPOSED SYSTEM OVERVIEW

The system is divided into five functional units: power supply, RFID reader, ESP32 controller, LCD/servo output and IoT web server. A product tag is presented to the reader. The reader transfers the tag identifier to the ESP32, which searches the registered product table. If the identifier matches a record, the controller obtains the product name and price, updates the basket and refreshes the LCD. The controller can then send the updated information over Wi-Fi.

The power unit supplies regulated voltages suitable for each component. The ESP32 executes the program and coordinates data flow. The LCD displays the most recent product and estimated total. The servo is an optional actuator and should only be used for a defined gate or release mechanism. The web server can show the basket remotely but should not be treated as a payment gateway unless that feature is separately designed and verified.

The overall design is modular: each unit can be tested independently before the complete trolley is assembled. This helps locate faults in wiring, tag reading, product lookup, display updates or network communication.

V. SYSTEM ARCHITECTURE

Fig. 1 presents the proposed system architecture. The battery feeds a buck converter, which provides regulated supply rails for the ESP32 and peripheral modules. The RFID RC522 reader sends tag data to the controller. The LCD communicates through the I²C interface, while a PWM output may control the servo. Wi-Fi connects the ESP32 to an IoT web server for remote viewing.

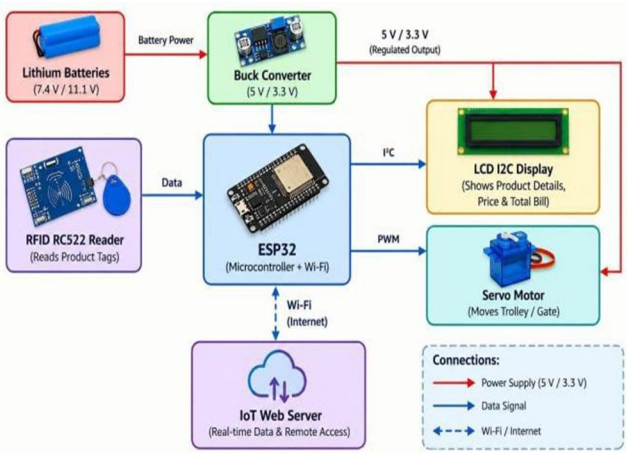


Fig. 1. Proposed block diagram of the smart shopping trolley system.

A. Power Supply Unit

A portable battery makes the trolley prototype convenient to move. A buck converter is used to reduce and regulate the battery voltage to levels required by the controller and peripherals. The output voltage and current capability must be selected from the specifications of the actual modules. A servo may draw short current peaks, so its supply should be sized appropriately. The controller, reader, display and servo signal circuit should share a common ground where required.

The RC522 reader normally uses 3.3 V logic and supply. Its pins should not be connected to 5 V signals unless the specific module is designed to tolerate them. The LCD backpack and servo supply requirements must also be checked before wiring. A stable supply helps prevent resets and corrupted readings.

B. RFID Reader and Tags

RFID identifies a product through a unique tag identifier. In the prototype, each participating product is associated with a registered tag and a record in the product table. When a tag is presented within the reader's range, the reader obtains its ID and passes it to the ESP32. The controller then looks for a matching record. The effective reading distance depends on the tag type, antenna arrangement, orientation and nearby metal or liquid.

An unknown tag should not change the bill. Instead, the display should show an error or an unregistered-item message. This prevents the system from adding a guessed price. For a real retail deployment, product identity and price data would need to be maintained securely and updated when prices change.

©IJRASET: All Rights are Reserved

C. ESP32 Controller

The ESP32 is the central processing unit of the design. It performs hardware initialisation, reads RFID data, searches the product table, manages the basket, calculates the total, refreshes the LCD and handles Wi-Fi communication. Its integrated wireless capability makes it suitable for an IoT demonstration without requiring a separate Wi-Fi module.

Firmware should be organised into small functions such as initialiseDevices(), readTag(), findProduct(), updateBasket(), calculateTotal(), refreshDisplay() and transmitBasket(). This modular approach improves readability and makes each function easier to test. A simple state machine can represent states such as READY, TAG_DETECTED, VALIDATING, UPDATE_BASKET and SYNCING.

D. LCD and Servo Actuator

The LCD gives immediate feedback and can display the item name, unit price, running total and status messages. A 16×2 LCD is sufficient for a compact demonstration, although longer product names may need scrolling or abbreviated labels. The display should distinguish a valid scan from an invalid tag or a network failure.

A servo motor can operate a small gate or item-release mechanism if the mechanical design includes one. The servo should not be described as driving the whole trolley unless motors, a suitable driver and a motion-control system are actually installed. Servo movement should be limited to safe positions and should not obstruct the shopper.

E. IoT Web Server

After the basket is updated, the ESP32 can transmit structured data to a local or cloud web server over Wi-Fi. The record may contain a trolley identifier, item identifiers, names, quantities, prices, current total and timestamp. The server interface can show the latest basket to a user or operator. For a student prototype, a lightweight HTTP endpoint is adequate for demonstrating data exchange. Network communication should include connection checks and timeouts. If Wi-Fi is unavailable, the local reader and display should continue to operate and the system should report that remote synchronisation has failed. Basket data should not be erased simply because the network connection is interrupted.

VI. HARDWARE AND SOFTWARE REQUIREMENTS

A. Hardware Components

ESP32 development board: Executes the control program, stores or retrieves product records, calculates the total and provides Wi-Fi connectivity.

RFID RC522 reader and compatible tags: Read the identifier assigned to each registered product. Reader and tag frequency must be compatible.

I²C LCD: Displays item information, current total, readiness and error messages. I²C reduces the number of controller pins needed.

Servo motor: Provides a controlled mechanical action for a small gate or release demonstration. It is optional and must be matched to the mechanical load.

Battery and buck converter: Supply portable power and regulate voltage. Current capacity and runtime should be checked under load.

Frame, wires and mounting hardware: Hold the modules securely and protect connections while the trolley is moved.

B. Software Components

The firmware can be developed in Arduino IDE using libraries suitable for the ESP32, SPI, MFRC522, Wire, LCD, servo control and Wi-Fi. A product table stores the tag ID, item name and unit price. A simple server application can accept basket data and present it in a browser. The final report should list the exact board, library versions, pin assignments and server platform used during implementation.

VII. OPERATING METHODOLOGY

At start-up, the ESP32 configures the RFID reader, LCD, servo signal and Wi-Fi interface. The LCD displays a ready message after local peripherals initialise. The controller then waits for a tag. If no tag is detected, it remains in the scan loop.

When a tag is detected, its identifier is read and validated against the product table. If the ID is unknown, an invalid-tag message is shown and the total remains unchanged. If the ID is valid, the product name and price are retrieved and the basket is updated. The system then recalculates the total and refreshes the display. A deliberate add/remove rule is needed so repeated scans do not unintentionally add the same item more than once.

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

The updated basket may be sent to the IoT server. The controller should check the server response and display whether the transfer succeeded. During checkout, the user or operator reviews the basket summary and verifies the items before payment. The prototype demonstrates estimated billing; it does not by itself provide payment settlement, anti-theft security or inventory reconciliation.

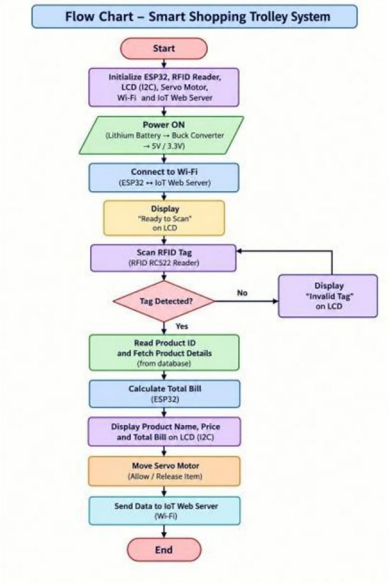


Fig. 2. Flow chart for the proposed smart shopping trolley operation.

A. Billing Calculation

The total is calculated from the quantity and unit price of each item in the current basket. It can be expressed as:

Total = □ (quantity of each product × unit price)

For example, if the basket contains two items priced at ■30 each and one item priced at ■50, the estimated total is (2 × ■30) + (1 × ■50) = ■110. The example illustrates the calculation only; it is not a measured result from the prototype.

When an item is removed, its quantity should be reduced and the total recalculated. The program must prevent negative quantities and must handle the case where an item is scanned but its product record is missing. If several identical products are required, the interface should make quantity changes clear to the user.

B. Duplicate-Scan Handling

An RFID tag can remain within range for more than one polling cycle. Without a scan lockout or confirmation rule, the same product may be counted repeatedly. The program can temporarily ignore the same ID, wait until the tag leaves the reading zone, or ask the user to confirm an add/remove action. The selected rule should be documented and tested with repeated scans.

C. Error Handling

Errors may include an unknown tag, failed reader initialisation, low battery, LCD communication fault or Wi-Fi disconnection. Error messages should be simple and should not silently alter the total. The firmware should preserve the local basket during a network failure and retry transmission only according to a controlled policy.

VIII. IMPLEMENTATION CONSIDERATIONS

The RFID RC522 generally communicates with the ESP32 over SPI, while the LCD uses I²C. The selected GPIO pins should be documented in a wiring table and checked against the particular ESP32 board. Pin assignments can vary between development boards, so a generic diagram should not be treated as a final wiring specification.

Before powering the circuit, verify polarity, ground connections, supply voltage and logic-level compatibility. The servo should normally use a supply capable of meeting its current demand rather than relying on the ESP32 3.3 V output. If the servo supply is separate, its ground should be connected to the controller ground for a shared control reference. Secure the wiring to avoid intermittent connections during trolley movement.

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

The product table may be stored in firmware for a small demonstration or retrieved from a local database. A hard-coded table is easy to test but requires firmware updates when the catalogue changes. A database-based design is more flexible but adds communication and data-consistency requirements. In either case, prices should be validated and the software should handle missing or malformed records.

A. *Firmware Structure*

A clean implementation separates reading, lookup, billing, display and networking. The main loop should not wait indefinitely for a server response because that could prevent further scans. Network operations should use timeouts, and errors should be logged for later diagnosis. The display should always show whether the amount is local and estimated or has been synchronised with the server.

B. *Mechanical and User-Interface Design*

The reader should be positioned so users can present tags consistently without needing to reach awkwardly into the basket. A clear scan area, stable mounting and short visual or audible confirmation can make the prototype easier to use. The LCD should not be obscured by products, and the servo mechanism should be guarded to avoid pinching or trapping fingers.

IX. TESTING AND EVALUATION PLAN

No measured performance results are claimed in this paper because the hardware has not been confirmed as tested. Evaluation should be performed after assembly using a known set of products and tags. Each test should be repeated and the observed result recorded rather than assuming that expected behaviour has been achieved.

The first test should scan each registered tag and confirm that the correct name and price appear. The second should scan an unknown tag and verify that the total does not change. The third should compare the displayed total with a manual calculation for a basket containing several products. The fourth should test repeated presentation of the same tag and check that duplicate handling follows the chosen rule..

Further tests should verify item removal, Wi-Fi loss and recovery, LCD messages, servo movement and operation under battery load. A useful test record contains test ID, input or procedure, expected result, observed result and pass/fail status. The test should be performed under the same conditions when comparing results.

A. *Suggested Evaluation Metrics*

Useful metrics include successful tag reads divided by the total number of attempts, average time from tag presentation to display update, correctness of the computed bill, rate of unintended duplicate entries and successful server transmissions. These values should be measured with a stated number of trials and test conditions. A claim such as “billing time is reduced by 50%” should not be made unless the study compares measured timings against a defined baseline.

The evaluation can also record behaviour at different tag orientations and distances. Tests should include normal use and likely error conditions. The final paper should report the test setup, sample size, results and any failures, allowing another student to reproduce the evaluation.

X. EXPECTED BENEFITS AND LIMITATIONS

A. *Expected Benefits*

The running total is expected to help customers monitor expenditure during product selection. RFID-based identification may reduce the need for manual entry at a checkout terminal in a controlled prototype. The project also demonstrates embedded C/C++ programming, serial peripheral interfaces, display control, product-data handling and IoT communication. Its modular design allows additional functions to be added after the basic workflow is stable.

A web interface can provide remote visibility of a basket and can serve as a foundation for future inventory or analytics features. Local display feedback can keep the basic interaction understandable even for users who are not familiar with the underlying electronics.

B. *Limitations*

RFID reading is affected by tag orientation, distance, interference and surrounding materials. A tag identifies a record but does not prove that the physical product has been correctly placed in or removed from the trolley. A mismatch between the product label and database entry can lead to an incorrect estimated bill. Repeated scans and nearby tags require careful handling.

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

Wi-Fi depends on network availability and the server must be protected against unauthorised access. Battery runtime depends on component current and operating conditions. The proposed prototype also lacks a verified payment system and commercial-grade loss prevention. These limitations should be acknowledged rather than presenting the prototype as a complete retail solution.

XI. EXPERIMENTAL IMPLEMENTATION AND OBSERVED RESULTS

The smart shopping trolley prototype was assembled using an ESP32 development board, an RFID reader, an LCD display, a battery supply and connecting wires mounted on a basket. The ESP32 program was developed and monitored through the Arduino IDE. The photographs document the physical setup and the serial output observed during the demonstration.

A. Hardware Observation

The LCD displayed a running total and a prompt to scan a product. This indicates that the display and controller were powered and that the user interface had reached its scanning state. The RFID reader was positioned near the basket lid so that a tag could be presented to the reader during product selection.

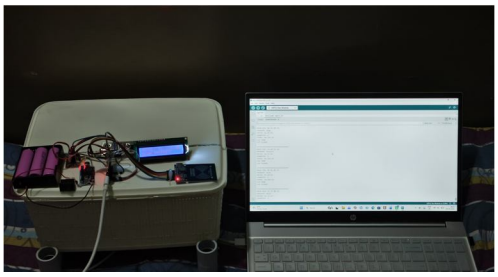


Fig. 3. Prototype hardware and LCD showing the total.

B. Serial Monitor Output

The Arduino IDE Serial Monitor showed records for scanned RFID identifiers, product names, unit prices, quantity and updated total. The displayed examples include Milk at Rs. 30.00, Oil at Rs. 120.00 and Biscuit at Rs. 20.00. The output also included “LID OPEN” and “LID CLOSED” status messages, indicating that lid-state information was reported by the program.

The captured output is evidence that the software produced product-related information after tag detection. It also shows that the prototype can present an accumulated total in the serial console while the LCD gives a compact on-device summary. The photographs are implementation observations, not a controlled accuracy study.

C. Example of Bill Calculation

For the three product records visible in the serial monitor, the listed unit prices are Rs. 30.00, Rs. 120.00 and Rs. 20.00. If one unit of each item is included, the arithmetic sum is Rs. 170.00. The serial display also shows a total of Rs. 350.00 after the Milk entry, Rs. 470.00 after the Oil entry and Rs. 490.00 after the Biscuit entry. These displayed totals suggest that earlier items or stored basket values were already included; the exact basket history cannot be confirmed from the photographs alone.

D. Interpretation

The prototype demonstrates the main interaction cycle: detect a tag, retrieve the corresponding product information, update the running amount and show status information. A visible total helps the shopper track estimated expenditure without waiting until checkout. The lid status messages may support a simple indication of whether the basket lid is open or closed, depending on the sensor and logic used in the implementation.

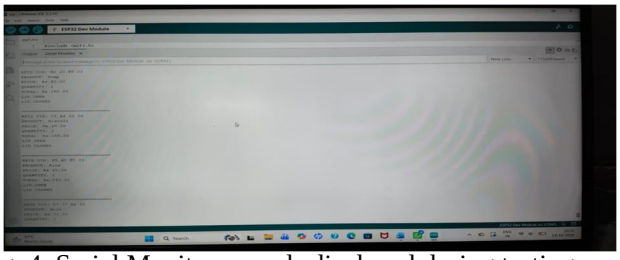


Fig. 4. Serial Monitor records displayed during testing

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

E. Limitations and Further Testing

The photographs do not establish RFID read accuracy, response time, duplicate-scan prevention, Wi-Fi delivery, or total-bill correctness across a full test set. Further evaluation should scan each registered tag repeatedly, test unknown tags, compare the displayed bill with a manual calculation, verify item removal and duplicate handling, and record Wi-Fi disconnection and recovery. The final report should include measured results from these tests rather than assumed performance.

F. Result Summary

Based on the provided photographs, the hardware was powered, the LCD showed a total and scan prompt, and the serial monitor displayed product identifiers, names, prices, totals and lid status. These observations support a basic functional demonstration of RFID-based product reporting and bill estimation. They do not, by themselves, prove commercial readiness or a specific percentage improvement in billing time.

XII. FUTURE ENHANCEMENTS

Several improvements can be considered after the basic prototype is validated. First, a database-backed product catalogue can simplify price updates and reduce the need to reprogram the controller. Second, a more informative display or a mobile-friendly web page can show item quantities, subtotal and error status. Third, a robust local storage method can preserve the basket during temporary network outages and synchronise it when connectivity returns.

The design could also include a battery-level monitor, low-battery warning and power-management mode. An improved reader placement or antenna arrangement may help make tag presentation more consistent, although the actual improvement must be verified through testing. A barcode or QR option may be added as a fallback for products without compatible RFID tags.

For a larger deployment, user authentication, encrypted communication, server-side price validation and audit logs would be important. Integration with an existing point-of-sale system would require a documented data format and secure interfaces. Any automatic gate or trolley movement feature would need mechanical safety checks and an emergency stop. Machine-learning recommendations are possible in a future version, but they would require appropriate data and privacy safeguards.

XIII. CONCLUSION

This paper presents a proposed smart shopping trolley based on RFID, an ESP32 controller, an LCD, an optional servo and Wi-Fi communication. The reader identifies a tagged product, the controller retrieves the associated record, calculates an estimated basket total and presents the result locally. A web server can provide remote viewing of the basket summary. The architecture brings together product identification, embedded processing and IoT communication in a modular educational prototype.

The usefulness of the design depends on reliable tag reads, correct product records, duplicate-scan control, stable power and appropriate error handling. The prototype should be evaluated using repeatable tests for identification, billing, display response and network recovery. Actual measurements must be reported before making claims about accuracy, time savings or commercial readiness.

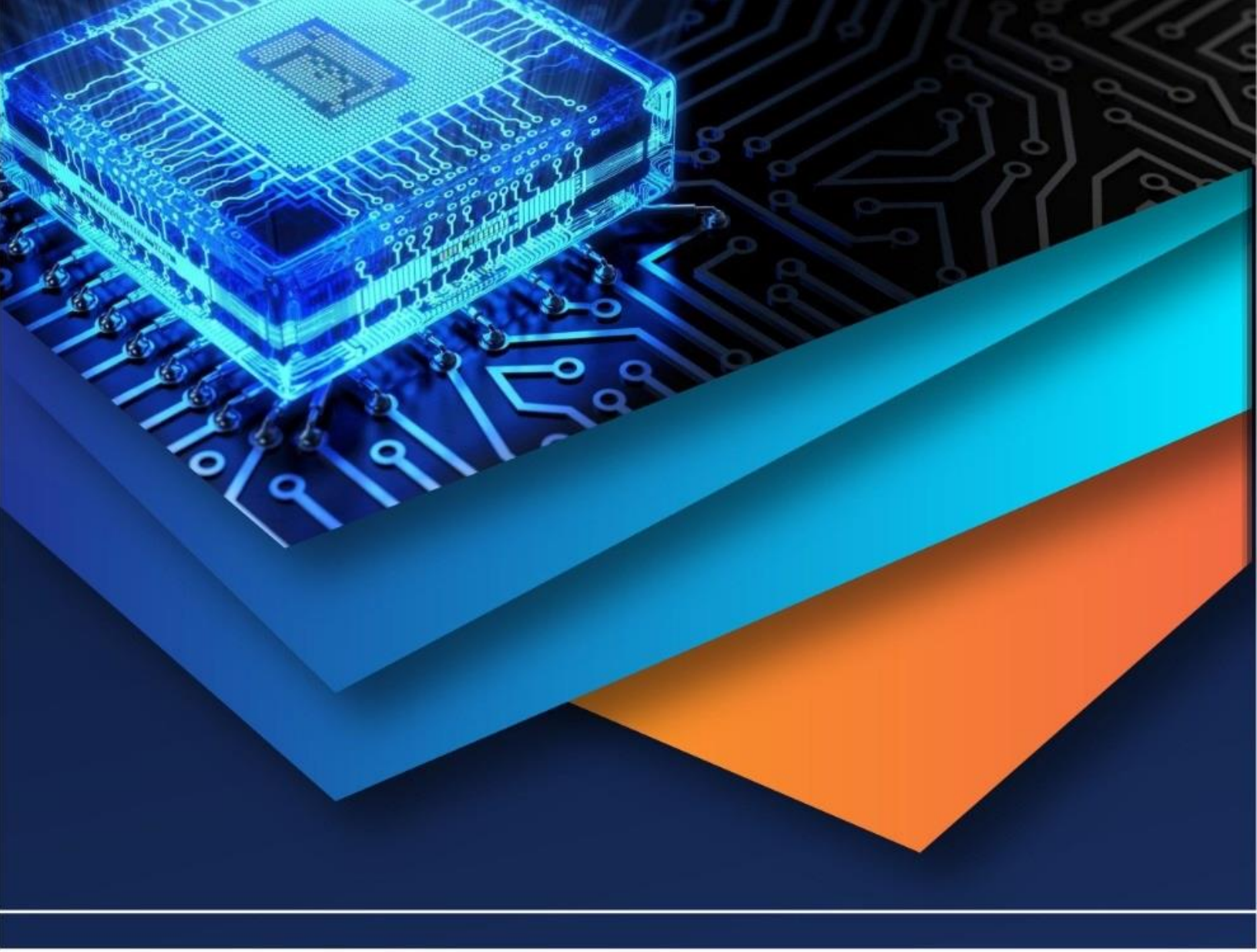
XIV. ACKNOWLEDGMENT

The authors would like to acknowledge the guidance and laboratory support provided by their department during the planning and development of this student project. Author names, institution details and acknowledgments should be updated to reflect the actual project team and support received.

REFERENCES

[1] K. Finkenzeller, RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication, 3rd ed. Wiley, 2010.
[2] Espressif Systems, “ESP32 Series Datasheet.” Available from the official Espressif documentation website. Consult the version corresponding to the development board used.
[3] NXP Semiconductors, “MFRC522 Standard Performance MIFARE and NTAG Frontend,” product data sheet. Consult the manufacturer documentation for electrical and interface limits.
[4] Arduino, “Arduino IDE Documentation.” Official documentation for developing and uploading embedded applications.
[5] IEEE, “IEEE Editorial Style Manual for Authors,” guidance for technical-paper structure, references and figure captions.
[6] M. R. Rieback, B. Crispo, and A. S. Tanenbaum, “The Evolution of RFID Security,” IEEE Pervasive Computing, vol. 5, no. 1, pp. 62–69, 2006.
[7] A. Juels, “RFID Security and Privacy: A Research Survey,” IEEE Journal on Selected Areas in Communications, vol. 24, no. 2, pp. 381–394, Feb. 2006.
[8] Espressif Systems, “ESP-IDF Programming Guide,” official documentation covering Wi-Fi configuration and application development.

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



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)