AUTOMATED DISPATCH SYSTEM USING RFID

-Mughuntha V

Abstract-*Warehouses serve as the fundamental component of supply chains, where effective management of dispatch operations is essential for minimizing delays and inaccuracies. This study introduces a cost-effective automated dispatch system that utilizes RFID (Radio Frequency Identification) technology in conjunction with Arduino Uno and Excel for data logging. The system is intended to function as a reliable model, incorporating specific locking mechanisms to ensure security. It facilitates the movement of inventory and oversees dispatch processes with very reduced reliance on human involvement.*

1.INTRODUCTION

In modern warehousing, it is essential to reduce operational expenses while simultaneously improving efficiency. An Automated Dispatch System (ADS) fulfills this requirement by monitoring inventory, automating the order fulfillment process, and accelerating dispatch activities. Conventional systems may be financially burdensome for smaller warehouses. Therefore, this research aims to develop and implement a cost-effective dispatch solution utilizing RFID technology, integrated with Arduino and a Thonny IDE Python application on a PC. Various automatic identification technologies, such as barcodes, magnetic stripes, and Radio Frequency Identification (RFID), are employed in security systems. RFID is a emerging technology and represents one of the fastest-growing sectors within the automatic identification data collection industry. This technology provides enhanced performance compared to other automatic identification systems. Unlike optical technologies such as barcoding, RFID does not necessitate a direct line of sight between the reader and the RFID-tagged item. The system outlined in this study employs an RFID tag equipped with an integrated circuit designed for storing and processing unique information, as well as modulating and demodulating the transmitted radio frequency signal.

2.METHODOLOGY

Hardware components include an Arduino Uno or a compatible variant, an MFRC522 RFID Reader Module, Piezoelectric Buzzer, passive RFID tags, and a A-B cable for serial communication. The software requirements consist of Thonny IDE utilizing the Python, along with libraries such as pyserial and openpyxl, and PlatformIO for Arduino firmware development.

Hardware Overview RFID Technology - RFID stands for Radio Frequency Identification and it's a non-contact technology that's broadly used in many industries for tasks such as personnel tracking, access control, supply chain management, books tracking in libraries, tollgate systems and so on [5]. Our RFID system consists of two main components, a tag which is located on the RFID card in which one want to be identified, and a transceiver or a reader which is installed at the secured entrance.

Our system RFID reader consists of a radio frequency module, a control unit and an antenna coil which generates high frequency electromagnetic field as shown in Fig.1. On the other hand, the tag used in this work is a passive component, which consists of just an antenna and an electronic microchip, so when it gets near the electromagnetic field of the transceiver installed at the secured entrance (2 to 5 inches), due to induction, a voltage is generated in the tags antenna coil and this voltage serves as power for the microchip of our system tag. Now as the tag is powered, it can extract the transmitted message from the reader, and for sending message (UID) back to the reader, it uses a technique called load manipulation. Switching on and off a load at the antenna of our tag will affect the power consumption of the reader's antenna which can be measured as voltage drop. These changes in the voltage will be captured as ones and zeros and that's the way the data is transferred from the tag to the reader.

How the reader reads the tag?:
We have two RFID tags with UID 737DA2D9 and 46F23302 and for RFID reader to get such information from the tag it needs to be converted from hexadecimal value to binary as shown in **Table 1.**

TABLE.1 HEX TO BINARY CONVERSION

Hex	Binary
7	0111
3	0011
7	0111
D	1101
A	1010
2	0010
D	1101
9	1001

32 bit worth of data is transferred from the tag to the reader in binary form (0111 0011 0111 1101 1010 0010 1101 1001). This data is transferred using high

frequency (HF) 13.56MHz, which is the frequency that our RFID system operates on [1].

Arduino Uno Board - The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM (pulse width modulation) outputs, 6 Analog inputs, a 16 MHz ceramic resonator, a USB (universal serial box) connection, a power jack, an ICSP (in-circuit serial programming) header, and a reset button. It contains everything needed to support the microcontroller, simply connect it to a computer with a A-B cable or power it with an AC-to-DC adapter(7/9/12v) or battery to get started [2].

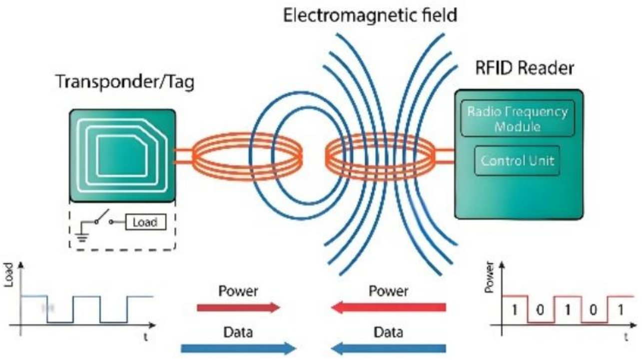

Fig.1. Working of RFID (image Source: Google)

TABLE.2. Available Methods and Cost Comparison.

Material/Sensor	**Cost**
RFID (MFRC522 + Tags)	65 rs
Arduino Uno	450 rs
Raspberry Pi + camera	6000 rs
Barcode System	2831 rs
NFC	254 rs
Bluetooth/Wifi Module	227 rs

Based on the prices presented in Table 2, it can be inferred that RFID technology is the most cost-effective option compared to others. While barcode scanners may have a comparable cost, it is important to note that the expense associated with the scanners themselves is significant. However, barcodes can be printed on paper or stickers and affixed to products, making them suitable for use in supermarkets. In contrast, their application in warehouses is limited due to the time-consuming nature of the process. Consequently, RFID is preferred for large-scale implementations, as RFID tags can be easily attached to products, similar to magnets, and magnetic RFID options are available in the market, thereby enhancing efficiency.

From the comparative analysis from Table.3, it is evident that although barcode scanning and NFC are cost-effective and widely used, they introduce significant operational limitations such as the need for manual handling and line-of-sight requirements. These constraints increase human dependency and reduce the system's efficiency in fast-paced warehouse environments. Bluetooth and Wifi based solutions, while wireless and autonomous, demand continuous power and are relatively expensive, making them unsuitable for low-budget deployments. The Raspberry Pi and camera-based systems using ML, though accurate and versatile, are unnecessarily complex and costly for a simple dispatch verification task. Given these evaluations, RFID emerged as the most balanced solution, offering contactless, automated, and cost-effective identification with minimal maintenance. It satisfies both budget and functional criteria essential for efficient warehouse dispatch operations.

TABLE.3. Comparative Analysis & Rejected Alternatives

Method	**Advantage**	**Disadvantage**	**Reason for Rejection**
Barcode scanning	Low cost tag, widespread	Manual, error prone, line of sight needed	Increases labour dependency
NFC	Secure, fast	Very short range, line of sight needed	Not scalable for warehouse
Bluetooth/ WIFI tags	Wireless, no wires	Expensive, needs power	Increases cost & maintenance
Raspberry Pi with camera	Powerful, high accuracy	Overkill for simple RFID tasks	Higher cost & complexity

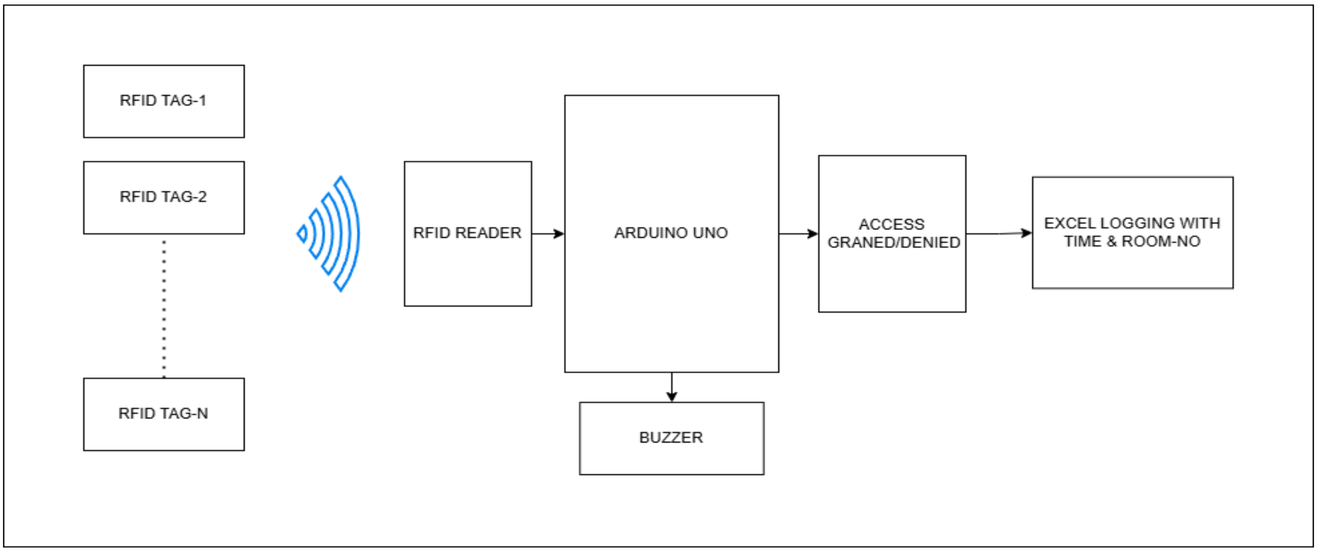

Fig.2. Block Diagram of Working of the prototype

3.CONCEPT

The system as mentioned in Fig.2 employs passive RFID tags that are allocated to inventory items or pallets. An RFID reader is utilized to scan each tag as it enters or exits a dispatch area [3],[4]. The Arduino identifies the tags and sends the data to the PC, where it is processed by Thonny IDE, which listens to the serial data and logs it into an Excel CSV file with timestamps using Python. A cooldown period of three minutes is implemented to prevent the recording of duplicate scans. This system enables warehouse operators to monitor the real-time movement of goods between specified zones (for instance, from Room A to Room B). If the same card is scanned after entering Room A, it is recorded in the 'entered Room B' column, with a waiting time of three minutes after entering Room A. On a larger scale, this can be applied in dispatch systems for e-commerce companies for prototyping purposes, and for small-scale implementations, our prototype utilized the terms Room A and Room B for better understanding, although they do not hold any significance in real-world scenarios.

The flowchart Fig.3 outlines the logical sequence of operations in the RFID-based automated dispatch system. It begins with scanning an RFID tag and validating whether it is pre-registered in the system. If the tag is not recognized, access is denied, and no further action is taken—ensuring unauthorized entries are blocked. For valid tags, access is granted, a buzzer beep confirms the scan, and the system records the event along with a real-time timestamp into a CSV log file. This design ensures secure, contactless dispatch logging with minimal human involvement. The looped structure allows continuous scanning, making it ideal for high throughput warehouse environments where speed and reliability are critical [7].

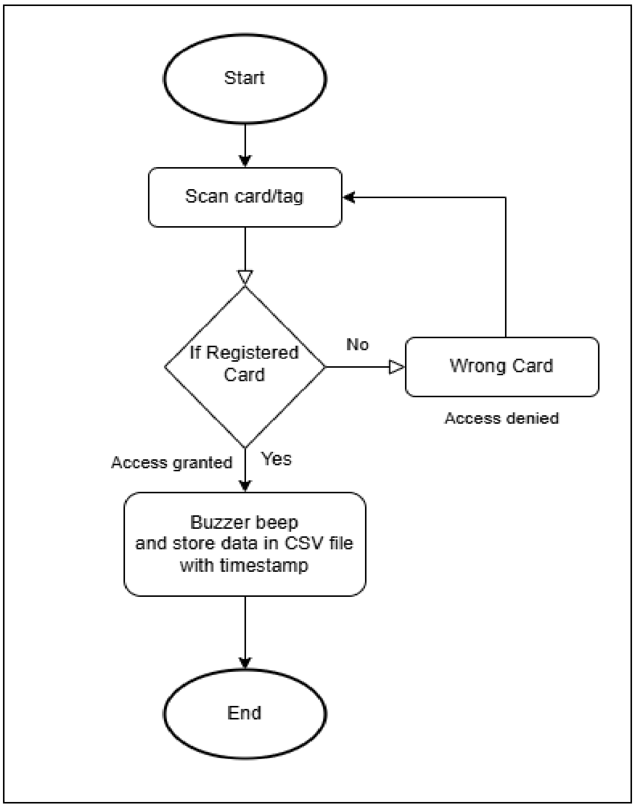

Fig.3. Operational framework diagram of the prototype.

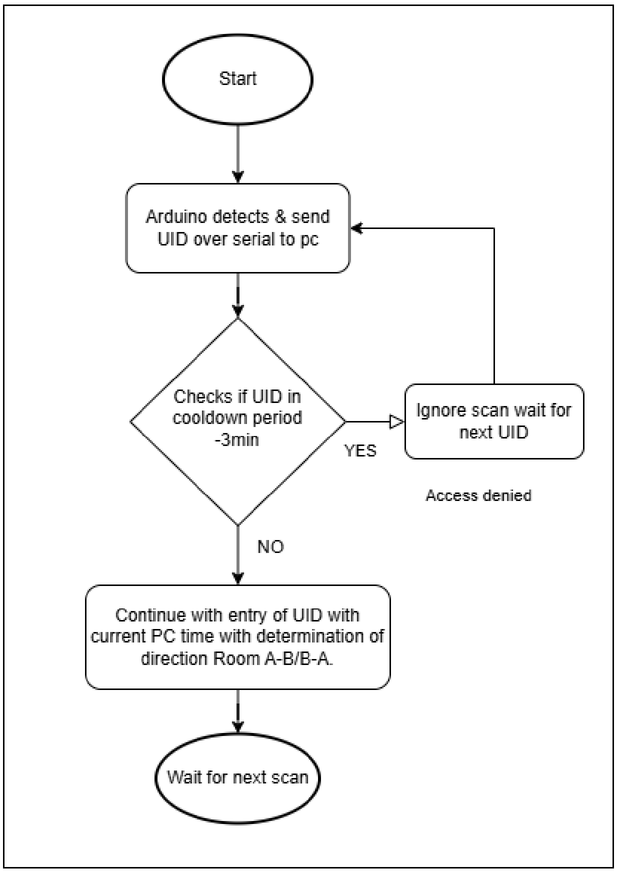

Fig.4 Thonny IDE Serial communication and logging Operational Framework

Python-Based Dispatch Logging Flowchart: The flowchart depicted in Fig.4 illustrates the Python- side logic for the RFID-based automated dispatch system. Once an RFID UID is sent from the Arduino to the PC through serial communication, the PySerial library listens to the connected port and checks for incoming data. After receiving the UID, the script checks whether it is currently within a cooldown period of three minutes. This mechanism is essential for eliminating duplicate entries from repeated scans within a short time frame. If the UID is still within the cooldown window, the scan is ignored and no entry is made. Otherwise, the system proceeds to log the UID into the dispatch record along with the current PC timestamp and the determined direction of movement (either Room A to B or B to A). The loop then continues to monitor for the next scan. By default, the first scan is considered as an entry from Room A and is not logged in Room B. This flow ensures efficient and accurate data logging while minimizing redundancy and system load, making it ideal for high-frequency dispatch zones in warehouses.[8]

S.N	RFID	Scan Time	Room-A	Room-B
1	737DA2D9	2025-05-21 13:09:59	Entered	
2	46F23302	2025-05-21 13:10:04	Entered	
3	737DA2D9	2025-05-21 13:13:17		Entered
4	46F23302	2025-05-21 13:13:19		Entered

Fig.5 Excel logged

Using the datetime library, the system retrieves the current date and time from the PC and stores it alongside the RFID entry, as illustrated in Fig. 5. The openpyxl library is used to save this data into an Excel file named rfid_log.xlsx, which is created and updated in real-time. In the above figure the same RFID tag pair is entered again after 3 minutes because if tried before it blocks and doesn't log. One key advantage of using Excel for logging is that it does not require an internet connection, making it suitable for offline warehouse environments.

4.CONCLUSION

The proposed low-cost automated dispatch system offers an effective, dependable, and economical solution for real-time inventory management. It utilizes commonly available components and open-source software, making it suitable for small enterprises. With slight modifications, it can adapt to more intricate logistics frameworks. The terms 'room A' and 'room B' are arbitrary and do not hold practical significance; once the product moves from its designated position, it will not return, thus requiring only a single RFID entry. As a potential enhancement, Google Sheets could replace Excel for improved accessibility, and the ESP8266 module could facilitate data transmission via Wi-Fi to a direct web server, potentially utilizing Amazon AWS for this purpose. At an industrial level, a UHF RFID reader can be employed for extended reading ranges of up to 12 mtrs or a 125 kHz RFID reader can be used with a reading distance of up to 7 cm.

REFERENCES

1. Shaik Mahammad Rasool, J., Varalakshmi, J., Singh, D., Akhil, P., & Mounika, G. (2018). RFID Based Security Access Control System with GSM Technology. *International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET)*, 4(7), 83–89. https://res.ijsrset.com/IJSRSET1844194

2. Pane, S. F., Awangga, R. M., & Azhari, B. (2018). Qualitative Evaluation of RFID Implementation on Warehouse Management System. *TELKOMNIKA (Telecommunication, Computing, Electronics and Control)*, 16(3), 1303–1308. https://doi.org/10.12928/TELKOMNIKA.v16i3.8400

3. Adiono, T., Ega, H., Kasan, H., & Harimurti, C. S. (2017). Fast Warehouse Management System (WMS) using RFID based goods locator system. In *2017 IEEE 6th Global Conference on Consumer Electronics (GCCE)* (pp. 1–2). IEEE. https://doi.org/10.1109/GCCE.2017.8229410

4. Mridha, M. (2022). IoT-Based Warehouse Management System: Enhancing Warehouse Efficiency and Worker Health Monitoring System. *ResearchGate*. https://doi.org/10.13140/RG.2.2.16212.82563

5. Orji, E., Oleka, C. V., & Nduanya, U. (2018). Automatic Access Control System using Arduino and RFID. *The Journal of Scientific and Engineering Research*, 5, 333–340.

6. Li, M., Gu, S., Chen, G., & Zhu, Z. (2011). A RFID-based Intelligent Warehouse Management System Design and Implementation. In *2011 IEEE 8th International Conference on e-Business Engineering* (pp. 178–184). IEEE. https://doi.org/10.1109/ICEBE.2011.28

7. Soni, S., Soni, R., & Waoo, A. (2021). RFID-Based Digital Door Locking System. *Indian Journal of Microprocessors and Microcontroller*, 1, 17–21. https://doi.org/10.35940/ijmm.B1707.091221

8. Chen, J. C., Cheng, C. H., Huang, P. B., et al. (2013). Warehouse management with lean and RFID application: a case study. *The International Journal of Advanced Manufacturing Technology*, 69, 531–542. https://doi.org/10.1007/s00170-013-5016-8