iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** XII **Month of publication:** December 2025

DOI: https://doi.org/10.22214/ijraset.2025.76634

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 13 Issue XII Dec 2025- Available at www.ijraset.com

Development of APP Based Automated Inventory Box

Naveen Kumar S N[1], A Kiran[2], M Vishal[3], Santhosh H N[4]
Department of Mechatronics, Acharya Institute of Technology Bengaluru, India

Abstract: ***Running out of supplies slows things down everywhere - factories, shops, hospitals, labs, warehouses. Most places still track what they have by hand. Writing it down takes time. Mistakes happen more than people admit. Updates lag behind real usage. This setup cannot keep up anymore. A small smart storage unit might change that. It runs on a tiny computer brain connected to detectors inside. Light beams notice when something moves in or out. Scales check how much remains at all times. Data travels wirelessly to a phone application automatically. People barely need to touch the process once it works right. Fewer errors show up next morning. When something gets put into or taken out of the storage box, these parts notice right away, updating the count without delay. You can check the logged details on a screen nearby - also pulling them from afar using internet-linked systems. Less need for hand-written logs means fewer errors in stock numbers, giving clearer insight into how supplies are actually used. Because the setup works in pieces, it fits places like offices keeping tabs on supplies, labs managing tools, even compact warehouses sorting goods. What results is a straightforward way to handle inventory more consistently across common settings.***

Keywords: App-Based Inventory Box, Microcontroller Connectivity, Wireless Communication, Real-Time Stock Monitoring, Smart Inventory System, Inventory Automation, Remote Monitoring.

I. INTRODUCTION

Starting off, keeping track of goods matters a lot in places like factories, stores, hospitals, shipping hubs, and science labs. Knowing what you have stops delays, keeps spending predictable, makes better use of supplies. Yet despite that need, plenty of groups - especially smaller businesses and university departments - still rely on old-fashioned ways to manage stock. Often, they check items by hand, write things down on paper, or lean on basic spreadsheets. Spending too long on old methods usually leads to mistakes. These ways can't track changes as they happen. Because of that, inventory numbers might not match reality. Items sometimes go missing without anyone noticing. Restocking either arrives late or doesn't fit actual needs.

Nowadays machines that think can watch supplies all by themselves. Because old ways of counting stuff fall short sometimes. Tiny computers wired into shelves notice when items come or go. Information travels through airwaves straight to dashboards people check later. Accuracy gets better without someone always typing updates. Workers spend less time walking around with clipboards. Anyone with permission views current numbers from faraway places. Still most ready-made tools target giant factories mostly. They cost too much for corner shops or niche labs usually. Setup takes expert help plus extra gear lying around. Simpler spaces just cannot fit those bulky designs easily. A fresh approach tackles small-scale tracking through a smart container built into an app-driven setup. Running on a microcontroller, it uses infrared plus load sensors to catch when items come or go. This mix helps spot each movement with strong precision. Data travels wirelessly to a phone interface, showing current levels and past trends without delay. Even offline, the unit gives feedback thanks to a screen fitted right on the device. Updates flow smoothly whether someone checks remotely or stands next to the box. A built-in structure focuses on separate components, lower expenses, one task at a time setup fitting labs, offices, even compact storage spaces. By shifting tracking tasks to automatic routines, mistakes drop while data accuracy rises. Clearer records lead to smarter choices about when to restock or reorganize supplies. Combining affordable computer chips with internet-linked sensors shows promise, offering growth potential without complexity. Real-time updates emerge naturally from linked devices, answering demand for responsive tools in everyday settings.

II. SCOPE AND OBJECTIVE OF THE STUDY

A. Scope of this Project

This project focuses on building an app-connected smart inventory container capable of tracking stored items automatically via built-in sensors and elementary Internet of Things methods. As objects enter or leave the enclosure, detection mechanisms trigger updates to reflect current stock levels instantly.

©IJRASET: All Rights are Reserved

Processing signals from these detectors, a small computer manages both operational functions and accurate data transmission. Its role includes interpreting inputs, coordinating actions across parts, and ensuring timely reporting through digital channels.

A small device takes shape - fitted with infrared detection and scales, showing current stock on a built-in screen. Communication happens wirelessly, sending updates to a phone app without physical checks. Accuracy climbs when automated tracking replaces handwritten logs. Real-time numbers appear instantly, cutting delays in supply oversight. Effort drops as the tool runs with minimal human steps.

Although built for compact spaces like labs or classrooms, the setup also fits offices and minor retail spots. Where space is tight, performance stays tied to how much the hardware can hold and detect. Instead of sprawling warehouses, it focuses on simpler needs no robots move goods here. Cloud analysis does not shape its function, nor do corporate software links define its role. Its purpose remains narrow, avoiding complex industrial demands.

Looking at how microcontrollers are programmed forms one part of this effort. Sensor connections to systems appear alongside these coding tasks. Communication through internet-of-things methods makes up another layer here. Mobile app links play a small role too. Learning by doing drives the purpose instead of building a market-ready tool. Real-world automation ideas take center stage in practice. Full-scale stock tracking isn't the goal. Classroom learning shapes the approach more than product design.

B. Objectives of this project

The primary objective of this project is to design and develop an app-based automated inventory box that can monitor inventory levels automatically and provide real-time updates with minimal human involvement.

The specific objectives of the project are listed below:

1) To design a compact and efficient inventory box suitable for storing and monitoring items.
2) To develop a microcontroller-based control unit for processing sensor inputs and managing inventory updates.
3) To implement wireless communication to support remote inventory monitoring through a mobile application.
4) To reduce manual effort and minimize errors associated with conventional inventory management practices.
5) To enhance the accuracy and reliability of inventory records through automation.

III. EXISTING DESIGN AND METHOD

A. Literature Study

Nowadays, handling inventory well matters more because products come in many types, storing them gets harder, yet seeing current stock levels remains essential. Instead of automated systems, older ways rely heavily on hand-written logs along with occasional checks done in person. Such processes take up much time while errors creep in regularly. Research shows depending only on these techniques may lead to wrong stock counts, late restocking, and slower workflows - particularly where goods move in and out fast [1], [2].

Emerging IoT tools have led many scholars to design smart tracking setups using sensor networks, compact computing units, one kind of radio signal transmission. These networked approaches make it possible to watch supply amounts without breaks, delivering fresh details whenever needed. According to findings from Jayanth and team [1], linking devices through IoT sharpens count precision while cutting down time spent on record changes. Work by Ugbebor with colleagues [2] showed such systems also lower errors in records, strengthening how choices are made across operations.

Some studies focus on using sensors to automate inventory tracking. Instead of manual checks, devices like infrared detectors, weight-sensitive cells, or closeness sensors help spot when items shift. One team led by Bose et al [3] found these tools could signal changes instantly upon detecting motion. Elsewhere, work by Rezwan et al [9] showed how combining small controllers with pressure-measuring sensors improves accuracy in spotting withdrawals especially useful where space is limited or machines dispense goods.

Though often explored, radio frequency identification remains a practical option for automating stock control. Without needing physical touch, these systems detect objects while handling many tags simultaneously. Studies led by Aluguri and team [6], along with work from Anusha's group [7], confirm better oversight of goods and fewer mistakes than hand-scanned methods. Still, findings reported by Alwadi and colleagues [10] highlight barriers - expensive setup, tangled networks, and constant upkeep - that slow adoption despite benefits.

Modern inventory systems rely heavily on wireless links and connections to online storage spaces. Using devices like NodeMCU or ESP8267, researchers have sent product counts over Wi-Fi into digital dashboards. Remote viewing of current supply levels became possible through a network setup built by Choi and team [12]. On another note, Mansor's group [13] showed how weight sensors joined with internet-linked chips support steady tracking - especially once tied to phone apps or websites.

Mobile apps now play a key role in tracking inventory, thanks largely to how common smartphones have become. With these tools, people can check current stock from distant locations, get notifications when supplies run low, yet also review how materials are used over time. Studies show such systems make information easier to reach, speed up reactions to changes, while making daily operations smoother especially in smaller storage areas or workplaces [14], [17]. Beyond real-time updates, platforms linked to the cloud allow deeper looks into past records; this helps anticipate future needs and shape restocking strategies [21].

One key focus in current research involves smart warehouses equipped with automated storage. Using IoT technology, Tejesh and Neeraja [14] developed a system that boosts tracking precision while cutting manual tasks. When processes run automatically, growth becomes easier to manage - especially where goods move quickly through facilities [16], [24].

Work on small-scale storage solutions has looked into portable designs for labs, workspaces, and tight rooms. In research by Patel et al. [19], alongside findings from Roy et al. [27], infrared sensors paired with load sensing offered consistent tracking where space is restricted. Such methods tie directly to smart containers managed through apps, aiming mostly at ease of use, precision, and little need for manual input.

Though earlier studies show IoT-powered inventory tools outperform manual tracking in precision and speed, several current options target only big facilities or depend heavily on RFID setups - adding expense and setup difficulty for modest operations. What stands out is a gap: a small, sensor-operated storage unit with app connectivity, delivering live updates without high costs or complicated installation, suitable for labs, offices, or mini warehouses [22],[30]. Instead of copying past models, this work builds an integrated approach using sensing devices, control units, network modules, and smartphone access all streamlined into one responsive framework.

B. Gaps

Despite clear benefits in precision and operational speed, IoT-driven inventory solutions still face barriers to real-world use. Earlier analyses point to lingering issues, particularly within limited-space settings. Some difficulties remain unaddressed even though technology advances. Smaller operations often struggle with adaptation due to structural constraints. Gaps persist in how these systems perform outside controlled conditions. Practical deployment lags behind theoretical promise. Research has not fully resolved reliability under space or resource limits.

Most studies so far have focused on big warehouses or company-wide stock setups using RFID tech along with heavy software support [6], [7], [10]. While such approaches work well, their steep setup expenses, complicated architecture, and ongoing upkeep needs limit suitability - smaller environments like labs, schools, tiny offices, or mini storage spaces often can’t manage them.

Another point is that many suggested setups depend almost entirely on just one type of tech - RFID alone, for instance, or only barcodes. Such methods tend to run into issues like needing clear sightlines, signal clashes between tags, high setup demands, and lower precision when used in tight areas. Still, solutions mixing sensors - like pairing infrared detection with weight checks - remain less studied, despite fitting well within small, budget-friendly storage units.

Notably, a number of inventory setups rely on cloud-linked IoT networks, yet most only record or display information without strong live performance. While some include instant stock tracking, notifications at set levels, or user-friendly phone access, these elements frequently remain underdeveloped. When quick insight into supplies matters, such gaps tend to weaken overall system usefulness.

Another point: research on mobile app integration exists, yet the focus tends to fall elsewhere, leaving this aspect underdeveloped within overall system design. When real-world stock levels shift, linked apps frequently fail to reflect these adjustments instantly - updates lag, creating friction during daily use.

Still, questions about how simple it is to set up, adjust components, or manage expenses tend to remain unresolved in most available options. Some present methods depend on unique setups or intricate arrangements, making them harder to fit into varied situations while also creating barriers for people without technical expertise.

Based on the literature review, it is clear that there is a need for a compact, low-cost, sensor-driven, app-based automated inventory box that meets the following requirements:

1) Specifically designed for small-scale and storage environments.
2) Integrates multiple sensing mechanisms to enhance the measurement accuracy.
3) Provides real-time inventory updates through a mobile application.
4) Allows for simple deployment, scalability, and ease of use for users.

IV. METHODOLOGY AND WORKFLOW

A. *Methodology*

At its core sits an Arduino Mega microcontroller, handling primary computation and coordination tasks. Selection of the Mega came down to its higher count of available input and output pins - this allows many devices to link at once. Devices like RFID readers, display screens, relays, sensing units, and data pathways can attach without conflict. Power reaches each part through a steady source designed to maintain constant voltage levels. Stability here helps keep performance predictable across all elements tied into the setup.

When needed, the system's moving parts come to life through DC motors guided by relay modules. Powering up happens solely after correct signals arrive, keeping motion accurate. Electric separation within the relays shields the Arduino Mega during heavy load tasks. Security takes shape via an RFID unit built into the setup. Only when confirmation clears does any motor respond. A single RFID tag goes to every person using the system. Once scanning happens, the Arduino Mega checks the data against known IDs already saved inside. On match, entry gets approved and the needed task runs right after. No match means no entry, plus a warning signal activates immediately afterward.

Real-time updates appear on a 16 by 2-character LCD linked to the device, showing login results, operation notes, and guidance cues. Sound alerts come from a buzzer while two small lights flash - this combination delivers both sound and sight warnings when access works, fails, or malfunctions occur.

A link forms wirelessly when the Arduino Mega connects to a phone via an IoT module. Through this channel, signals travel both ways without physical contact. Operation happens on a small app called Blynk, found on handheld devices. From there, people watch current readings as they change moment by moment. Commands go out from the screen straight into the hardware. Alerts pop up whenever conditions shift beyond set points. Interaction stays constant, happening minute by minute. Using just a mobile, adjustments occur instantly across distance. Access opens wide, even when far from the actual machine. Efficiency climbs because delays drop sharply.

A solid approach ties strong login security to automatic physical operations, live updates, one after another, through radio signals - building a setup that runs smoothly, holds up over time, yet stays simple to operate. This version fits best where tracking stored items matters, also where entry needs checking.

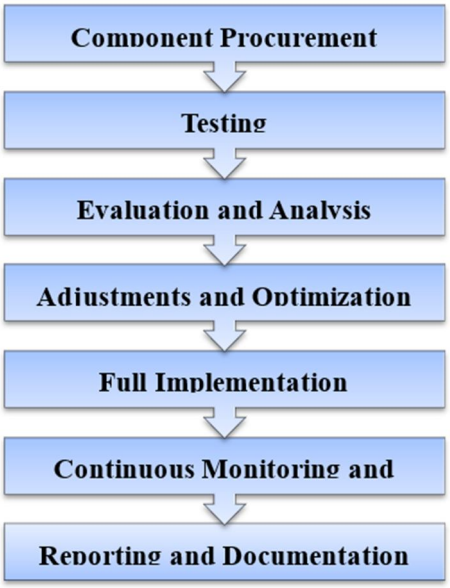


Fig 1: Methodology of the system

B. Workflow of the system

Starting up, the NodeMCU sets up communication channels for attached hardware components. Right after, it configures digital ports linked to six infrared detectors. A network link activates through the built-in wireless module. The process follows a predefined sequence coded into the device. Shown in Figure 2, the diagram outlines how inventory tracking functions via this setup. Detection of items occurs systematically. Data updates happen instantly across connected systems. Alerts trigger automatically when stock runs low. All operations rely on internet-enabled coordination. Following authentication, the system connects to the Blynk IoT cloud through its designated template credentials. With that done, every vending spring loads exactly five items by default; once stock dips below a fixed point, monitoring kicks in automatically.

Now operating, infrared units watch every coil's release system without pause. As something drops, it breaks the light path - this sends a brief LOW pulse to the matching input. Code keeps scanning these inputs, shifting status only when changes stick. Instead of reacting instantly, delays paired with change alerts filter out electrical jitters that could trick the count. A single drop becomes official only when the signal shifts from high to low then returns to normal. After confirmation, the system subtracts one from that spring's stored quantity. Every spring has its own counter, running separately without overlap. Data flows instantly to the Blynk platform using virtual pins as channels. With each spring linked to a unique stream, the app reflects current stock without delay.

Once stocks get updated, comparison happens between existing levels and set limits. When a spring's count meets or drops under that limit, an alert activates. This process runs on events, firing just one time per breach to prevent repeat messages. Notifications travel via Blynk, giving fast updates straight to the operator. Cloud-linked email warnings also stand ready if turned on.

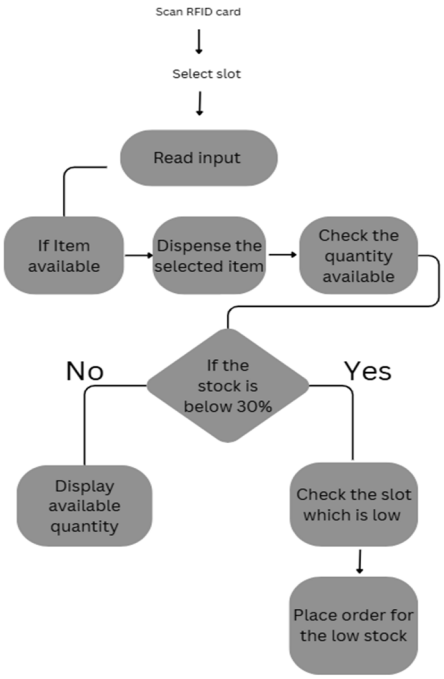


Fig 2: Workflow of the system.

The core process runs on its own, handling sensor checks, sending data to the cloud, while also judging alerts as they arise. Messages appear through serial output, aiding in verifying function and finding issues quickly. From start to finish, the sequence allows correct identification of products, steady updates to inventory levels, along with alerts that respond to actual conditions. With sensors feeding live inputs and connectivity supporting remote oversight, performance gains emerge naturally - fewer errors occur, less hands-on checking is needed, daily operations run smoother across self-service units.

V. DESIGN AND IMPLEMENTATION

A. Overview of Setup

Monitoring what is available happens inside the software part of the new smart inventory setup. This core section takes incoming signals from sensors, updates how much remain on shelves, then sends warnings using Blynk's internet-of-things network.

Six infrared detectors support the process - each placed beneath its own spring-loaded holder, where every slot fits maximum five units. Once something gets taken out, the spring shifts slightly. That motion triggers the nearby sensor to signal change. Stock numbers drop by one right after detection. Decisions and operations run through a NodeMCU chip, known as ESP8266. Wireless links plus cloud interactions come from the same board handling internal logic. Information flows without physical buttons or manual checks. A sequence begins as infrared sensors gather data nonstop, feeding it into processing routines right away. Connected individually to digital pins on the NodeMCU, every sensor responds when an object crosses its path. Interruption of the infrared beam leads to a drop in voltage, producing a LOW signal at that moment. Rather than rely solely on one approach, monitoring happens through periodic checks or triggered responses based on system needs. Because electrical disturbances might distort readings, small delays filter out unstable pulses before registration. When validation confirms removal, the system reduces the inventory number by one for that spring. Suppose a spring starts with five - taking away one leaves four behind.

Tracking inventory separately happens through unique runtime variables - spring1_count up to spring6_count - on the NodeMCU. During active operation, these figures live in temporary memory. If needed, they transfer into long-term storage like EEPROM or SPIFFS before power loss or restart interrupts occur. Whenever changes take place, updated counts move toward the Blynk platform using virtual pins. That flow keeps the mobile app display aligned with current stock states. Real-time updates emerge without delay after every adjustment.

From a distance, the inventory system links up with the Blynk IoT platform for live tracking. Stock amounts for every spring appear through six separate channels - labeled V1 through V6 - in the Blynk setup. Running on NodeMCU, the code adjusts those values on the fly by sending virtual writes. With updates stored in the cloud, anyone online can check current stock, no matter where they are.

A single alert system operates within the code, watching item counts closely. Five units mark the starting amount for every spring compartment. Should levels drop to two or lower, a signal triggers without delay. This detection relies on constant scanning by the running process. Notification happens through Blynk's built-in logging tool. Once activated, the message travels straight to the operator's phone. Each warning points directly to the exact storage position needing refill.

When needed, emails go out through the Blynk system alongside phone alerts. Receiving updates continues uninterrupted, should a mobile alert fail to reach the user. Alerts fire just one time upon hitting the stock limit, preventing repetition.

Every time it starts, the system sets up sensor pins to receive data, connects to Wi-Fi, then activates Blynk with stored login details. Functioning on a NodeMCU, its design relies heavily on setup and recurring loop processes. Cloud links stay active through constant checks inside the primary cycle. Sensors get watched without pause, their signals filtered by debounce routines before any update occurs. Inventory figures change only after valid readings pass validation steps. Alerts trigger when thresholds cross predefined limits during routine scans.

A shift in state gets detected through an algorithm designed to count every removed item just one time. Even though infrared readings may waver when something moves, the code waits for a full cycle - HIGH to LOW then back to HIGH - before accepting it as real. To help check performance and catch errors, data flows out over serial connections. Messages pop up on the monitor showing when sensors trigger, stock levels change, or warnings begin.

The design brings together sensor inputs, stock tracking, data transfer to remote servers, and alerts triggered by preset levels. Operation runs independently once active, delivering up-to-date item counts reliably. Decisions about replenishment happen faster due to consistent updates. Efficiency improves alongside dependability of the entire setup no human efforts needed for stock maintenance.

B. Software Implementation

The software Running on a NodeMCU (ESP8266) board, the software part of the suggested smart inventory setup takes charge of decisions and operations. Because it has native Wi-Fi support, live connection to cloud services happens without delay. Instead of relying on manual inputs, this component gathers readings from sensors, adjusts stock numbers, then triggers warnings when needed. While handling constant updates, its role stays central in tracking items precisely across time.

Item detection is carried out using six infrared sensors, each assigned to a single storage spring. Each spring can hold up to five items. When an item is removed, the corresponding spring mechanism releases the object, temporarily interrupting the infrared beam. This interruption generates a LOW logic signal at the NodeMCU digital input pin. The software continuously monitors these signals using polling or interrupt-based methods. To ensure accurate operation, a debouncing routine combined with state-transition logic is implemented to filter noise and prevent multiple counts for a single item movement. Once an event is validated, the inventory count for the corresponding spring is reduced by one unit.

The system maintains six independent inventory variables (spring1_count to spring6_count), each representing the current stock level of an individual spring. These values are stored in volatile memory during runtime and can optionally be saved to non-volatile storage, such as EEPROM or SPIFFS, to preserve inventory data during power interruptions or system restarts. After each update, the revised stock levels are transmitted to the Blynk Cloud using virtual pin communication, ensuring that the mobile application displays accurate and up-to-date inventory information.

One standout part of the software lies in how it signals when supplies run low. Starting fresh, every spring holds exactly five units - with two set as the minimum safe count. Tracking happens nonstop, checking whether levels dip near that limit. If a spring hits two or fewer, an automatic flag goes up. This signal rides through the Blynk system, sending instant alerts straight to the operator's phone. Each message clearly points to the exact slot needing refill.

When inventory drops below a set level, an alert activates just one time. The setup includes email warnings via Blynk, adding another way messages get through. Even if phone alerts go unnoticed, updates still arrive by mail. After the first signal, further removals won't prompt more alerts - keeping things quiet until levels reset.

Every time it starts, the system sets up infrared sensor ports to receive signals. Right after, connection to Wi-Fi begins alongside activation of the Blynk service with saved login details. Running nonstop, the core cycle checks sensors, applies delay-based filtering to avoid false triggers. Inventory data changes happen right after readings stabilize. Syncing with online servers occurs shortly thereafter, ensuring remote access stays current. Alerts get reviewed each round, based on updated values just collected.

Errors leads to data shared through serial output. On-screen, sensor signals arrive before current inventory numbers clear views help when reviewing. Warnings pop up without delay, making problems easier to catch early. Information moves in a way that simplifies tests, as each update appears right away. When timing and order make sense, finding problems takes less time. Over days, patterns in how things run show up just by observing log growth.

One feature stands out: automatic updates triggered by live sensor inputs. Data flows smoothly between components, avoiding delays. Where older methods require constant checks, this approach uses connected devices to track changes instantly. Alerts activate under specific conditions, not on fixed schedules. Inventory levels influence decisions directly, reducing guesswork. What happens next depends on actual usage patterns, not preset rules. System stability improves because responses align closely with current needs. Reliability grows when actions follow real-world events.

C. Hardware Implementation

At its core sits an Arduino Mega 2560 - handling coordination across sensing, movement, visual output, and network connectivity. Built around this board are several attached components: infrared detectors keep watch, while a motor driven by relays adjusts physical access. A small screen shows real-time status using character-based feedback. Wireless data transfer happens through a compact Wi-Fi chip linked directly to the circuit. Meanwhile, identification of stored items relies on signals captured by the RFID sensor. Together, these elements allow hands-off tracking and management of stock levels.

Power reaches the Arduino Mega 2560 via a steady 5 V source or through a 7–12 V DC connection at the Vin pin or barrel jack. To ensure consistent voltage levels, every part shares the same ground point. Mechanical tasks rely on a high-power DC motor switched safely using a relay linked to the board. This relay draws its power from a 5 V line, yet gets activation cues from one of the digital pins.

A power source of 12 volts drives the DC motor, while relay contacts separate it electrically from the microcontroller. Protection for the switching components comes from a diode placed across the motor's leads, which also reduces voltage spikes caused by back EMF.

When something passes through, the infrared beam breaks. Each sensor runs on 5 volts. Digital pulses form instantly during interruption. These signals go to separate pins on the Arduino Mega. Real-time tracking happens through pulse recognition. Processing occurs directly on the board. Inventory numbers change as items move. Accuracy improves with consistent signal timing.

A small screen with sixteen columns and two rows connects using half the usual signal lines to save port space. Instead of full wiring, only four data terminals - labelled D4 through D7 - link to output points on the microcontroller. Control signals like RS and E also attach directly to digital ports for precise timing. To fine-tune visibility, a variable resistor shapes the darkness level seen on the panel. Power flows to the backlit layer via a protective resistor that prevents excess electrical flow.

Cloud-based alerts and monitoring come from the ESP8267's wireless link. Running on 3.3 V, the chip gets steady power through a separate regulator. To prevent signal damage, data exchange between the 5 V Arduino Mega and the lower-voltage module passes through a level shifter. Communication travels via Serial1, one of the Mega's built-in UART channels.

Connected to the Arduino through SPI, an RFID module handles identification tasks. Running on 3.3 volts, it uses the shared SPI pin setup on the Arduino Mega - yet keeps a separate slave select wire for precise operation. This arrangement supports stable data reading, controlled energy flow, smooth signal exchange, and room to grow. Such traits fit naturally into live tracking systems for stock levels.

VI. MODEL AND SIMULATION

A. System Model

The tests took place using actual working settings to check how well the software handled precision, consistency, and speed. As seen in Figure 3, the physical model put together for trials included a full setup. Testing began after placing exactly five objects into every storage coil of the machine. Right away, the smartphone app named Blynk showed accurate starting quantities. This outcome indicated that both startup procedures worked fine while proving stable communication existed between the microcontroller and online service.

A custom-built circuit runs on an Arduino Mega 2560, linked to several relay units alongside a stable power source. One part uses a four-relay board; another adds a second four-relay set; a third brings in a dual-relay configuration. Signals from the microcontroller's digital outputs activate these switches, managing separate electric circuits like small motors or signal lights. Connections for alternating or direct current devices lock into place through built-in screw terminals on each module. A steady 5 V comes from converting 12 V DC through a controlled power source, running both the relays and their control parts. Operation is visible via an LED light built into the unit. Connection options include a barrel socket along with terminal screws, distributing energy without interruption.

Midway through tests, whenever a product dropped from a spring channel, its paired infrared sensor reliably registered the motion. Signal shifts triggered immediate analysis by the system, using logic checks to confirm real drops versus noise. Each release got logged exactly one time, despite quick on-off pulses in the data stream. Across repeated trials, every drop showed up correctly - no extra tallies, no omissions. Performance stayed steady, pointing to well-tuned code behind the scenes.

A system kept track of items while running nonstop. After someone took an object, the new amount showed up fast on the phone app using specific digital channels. Each of the six springs had its current level visible at once on the screen. There were no lags, which suggested solid wireless connection and smooth data transfer to and from the internet.



Fig 3: System Model

Midway through testing, a signal fired off the moment stock dipped to two units. Should levels hit that mark, Blynk sent a message without delay. One particular coil spring prompted the warning, clearly named in the update arriving on the phone. Instead of constant pings, the process woke just once per drop below the limit. Each trigger stayed tied to a fresh dip, never repeating unless the condition reset first. From start to finish, findings show the new inventory setup works well in everyday environments. Detection of items remains precise, thanks to consistent updates across devices as changes occur. Alerts appear quickly when needed, showing the method fits practical uses in automatic stock tracking. Performance holds up even outside controlled lab settings.

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

B. Simulation Result

The simulation of the proposed inventory system was conducted to verify the correctness of system operation and alert functionality. The results confirmed the successful execution of all software modules. When the inventory level of a spring reached the predefined threshold, the system generated a low-stock event as expected. The notification was successfully received on the user's mobile device via the Blynk platform, validating proper cloud communication and alert triggering. Figure 4 shows the notification received during the simulation, confirming that the alert mechanism operates as intended. These results demonstrate the system's reliability under simulated operating conditions.

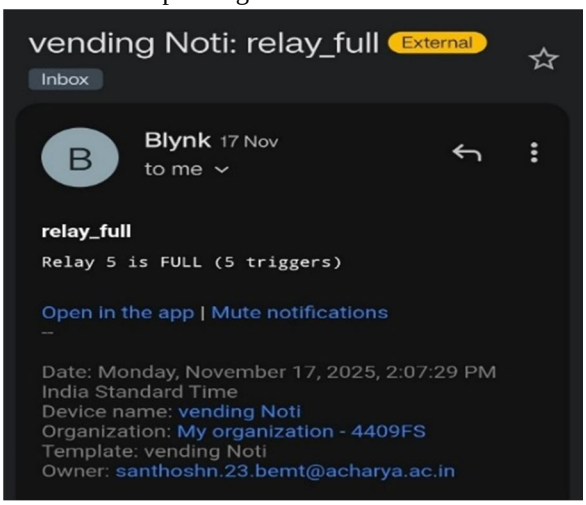


Fig 4: Notification received

VII. CONCLUSION

A small automated storage unit runs on a smart controller, making inventory tasks easier without constant supervision. Infrared detectors track when items come or go, signalling changes instantly. Movement triggers a belt system powered by a DC motor, moving goods smoothly within the box. Safety during electrical switching comes from relay components built into the circuitry. Information appears clearly on a compact screen that shows updates as they happen. A Wi-Fi chip based on ESP8266 technology links everything wirelessly, sending data outward. Users see current levels through a phone app designed for continuous tracking.

Tracking items nonstop while refreshing stock numbers on its own cuts down on people needing to step in, also lowering mistakes made by hand-written logs. Because updates happen right away, plus warnings pop up instantly, choices get sharper and refills arrive when needed. Warehouses lean into this setup, just like robotic shelves, snack boxes that sell themselves, or any spot running tight on supply oversight. Another benefit lies in its use of affordable parts alongside freely available software, allowing adjustments whether for minor setups or extensive production environments. Built-in automation features combined with networked sensors enable the App-Based Automated Inventory Box to reduce mistakes, simplify daily tasks, while offering clearer tracking - turning it into a working answer for today's connected stock control needs.

REFERENCES

[1] S. Jayanth, M. B. Poorvi, and M. P. Sunil, "Inventory management system using IoT," in Proc. Int. Conf. Computational Intelligence and Informatics, Springer, 2017, pp. 203–212, doi: 10.1007/978-981-10-2471-9_20.

[2] F. U. Ugbebor, M. Adeteye, and J. Ugbebor, "Automated inventory management systems with IoT integration," J. Artificial Intelligence General Science, vol. 6, no. 1, pp. 1–12, 2024, doi: 10.60087/jaigs.v6i1.257.

[3] R. Bose, H. Mondal, I. Sarkar, and S. Roy, "Design of smart inventory management system based on IoT," e-Prime – Advances in Electrical Engineering, Electronics and Energy, 2022, doi: 10.1016/j.prime.2022.100051.

[4] O. A. Madamidola, O. Daramola, and A. A. Akinwonmi, "Framework for an IoT-enabled intelligent inventory management system," Int. J. Advances in Engineering and Management, vol. 6, no. 10, pp. 234–243, 2024, doi: 10.35629/5252-0610-234243.

[5] A. Salsabila and L. Anifah, "IoT-based RFID system for automated inventory management," Indonesian J. Electrical and Electronics Engineering, vol. 8, no. 2, pp. 76–83, 2025, doi: 10.26740/inajeee.v8n2.p76-83.

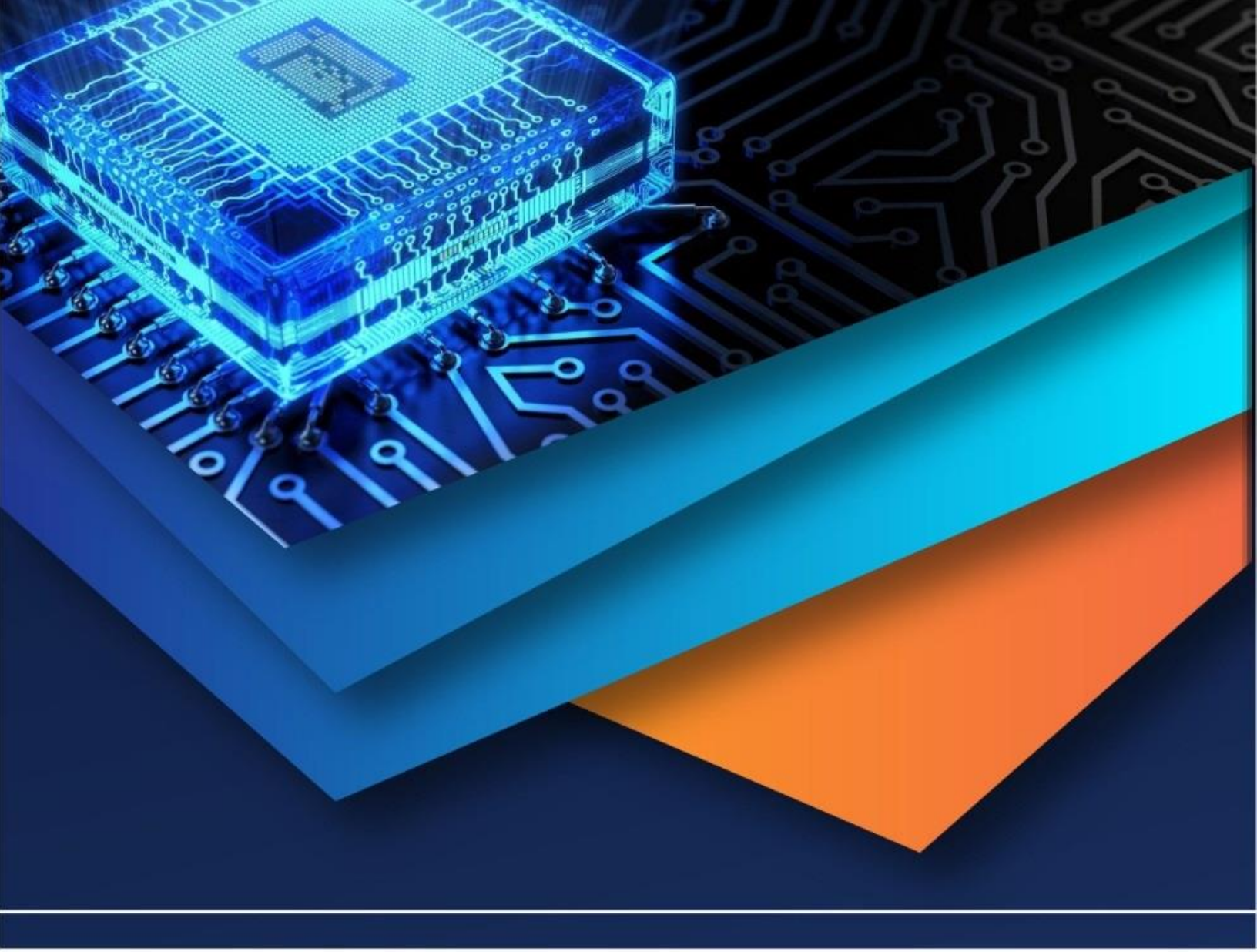
[6] S. V. Aluguri et al., "RFID based inventory management system," Int. J. Research in Applied Science and Engineering Technology, vol. 11, no. 4, pp. 512–517, 2023, doi: 10.22214/ijraset.2023.56210.

[7] C. H. Anusha et al., "RFID based inventory management system," Int. J. Engineering Applied Sciences and Technology, vol. 8, no. 5, pp. 98–102, 2023, doi: 10.33564/IJEAST.2023.v08i05.018.

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

[8] W. C. Tan, "RFID and IoT integration in supply chain management: A review," Computers & Industrial Engineering, vol. 167, 2022, doi: 10.1016/j.cie.2021.108075.
[9] S. Rezwan et al., "IoT-based smart inventory system using load sensors and NodeMCU," in Proc. IEEE ICACCAF, 2018, pp. 1–6, doi: 10.1109/ICACCAF.2018.8776761.
[10] A. Alwadi et al., "Smart solutions for RFID-based inventory management systems: A survey," Scalable Computing: Practice and Experience, vol. 18, no. 4, pp. 339–349, 2017, doi: 10.12694/scpe.v18i4.1333.
[11] J. J. A. Basa et al., "Smart inventory system using wireless sensor network," Int. J. Emerging Technology and Engineering Research, vol. 7, no. 10, pp. 45–51, 2019, doi: 10.30534/ijeter/2019/057102019.
[12] E. S. Choi et al., "Implementation of IoT-based automatic inventory management system," Int. J. Advanced Culture Technology, vol. 5, no. 1, pp. 70–76, 2017, doi: 10.17703/IJACT.2017.5.1.70.
[13] M. N. Mansor et al., "Arduino-based inventory monitoring system using load cell and NodeMCU," J. Advanced Research in Applied Sciences and Engineering Technology, vol. 32, no. 3, pp. 12–25, 2023, doi: 10.37934/araset.32.3.1225.
[14] B. S. S. Tejesh and S. Neeraja, "Warehouse inventory management system using IoT," Engineering Applications of Artificial Intelligence, vol. 76, pp. 13–23, 2018, doi: 10.1016/j.engappai.2018.06.009.
[15] M. Chen, Y. Cheng, and C. Siang, "Inventory management using RFID and sensor technologies," Sensors and Materials, vol. 34, no. 4, pp. 1231–1242, 2022, doi: 10.18494/SAM.2022.3632.
[16] Singh and R. Verma, "Smart warehouse management using IoT sensors," Int. J. Intelligent Systems and Applications in Engineering, vol. 7, no. 4, pp. 145–151, 2019, doi: 10.18201/ijisae.2019452783.
[17] S. Raut et al., "IoT-based inventory monitoring system with mobile application," Int. J. Engineering Research and Technology, vol. 9, no. 6, pp. 224–229, 2020, doi: 10.17577/IJERTV9IS060168.
[18] A. Kumar and P. Singh, "Real-time inventory tracking using IoT and cloud computing," Int. J. Scientific and Technology Research, vol. 9, no. 3, pp. 5698–5703, 2020.
[19] R. Patel et al., "Smart inventory system using infrared and weight sensors," Int. J. Innovative Technology and Exploring Engineering, vol. 8, no. 11, pp. 1785–1789, 2019, doi: 10.35940/ijitee.K2269.0981119.
[20] N. Kumar et al., "IoT-based smart inventory management using ESP8266," Int. J. Electrical and Computer Engineering, vol. 10, no. 4, pp. 3982–3990, 2020, doi: 10.11591/ijece.v10i4.pp3982-3990.
[21] A. Verma and S. Gupta, "Cloud-connected inventory monitoring system," Procedia Computer Science, vol. 171, pp. 1456–1465, 2020, doi: 10.1016/j.procs.2020.04.156
[22] P. R. Nair et al., "IoT-based smart box for inventory control," Int. J. Engineering Trends and Technology, vol. 68, no. 6, pp. 87–92, 2020, doi: 10.14445/22315381/IJETT-V68I6P215.
[23] S. M. Girish et al., "Automated inventory system using sensors and mobile app," Int. J. Computer Applications, vol. 176, no. 32, pp. 10–15, 2020, doi: 10.5120/ijca2020920633.
[24] A. K. Sharma and R. Kumar, "IoT-enabled inventory automation for small warehouses," Int. J. Recent Technology and Engineering, vol. 8, no. 6, pp. 321–326, 2020, doi: 10.35940/ijrte.F7563.038620.
[25] M. A. Rahman et al., "Smart inventory monitoring using IoT," Int. J. Advanced Computer Science and Applications, vol. 11, no. 7, pp. 315–321, 2020, doi: 10.14569/IJACSA.2020.0110742.
[26] H. Patel and D. Shah, "IoT based inventory control system," Int. J. Engineering Research & Technology, vol. 9, no. 5, pp. 402–406, 2020, doi: 10.17577/IJERTV9IS050230.
[27] A. Roy et al., "Smart inventory box using microcontroller and sensors," Int. J. Electronics Engineering Research, vol. 12, no. 3, pp. 431–438, 2020, doi: 10.37622/IJEER/12.3.2020.431-438.
[28] S. K. Panda et al., "Inventory automation using IoT and mobile application," Int. J. Emerging Technologies in Engineering Research, vol. 8, no. 8, pp. 44–49, 2020, doi: 10.30534/ijeter/2020/4482020.
[29] R. Mehta and N. Shah, "IoT-based smart inventory management system," Int. J. Innovative Research in Science, Engineering and Technology, vol. 9, no. 6, pp. 5123–5129, 2020, doi: 10.15680/IJIRSET.2020.0906021.
[30] V. Reddy et al., "Automated inventory monitoring system using IoT," Int. J. Engineering and Advanced Technology, vol. 9, no. 3, pp. 2246–2251, 2020, doi: 10.35940/ijeat.C6474.029320.

©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)