International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0356
e ISSN: 2584-2854
Volume: 04 Issue: 07
July 2026
Page No: 2483 - 2489

Smart Inventory CRUD Web Application with NLP

Prathmesh G. Sose[1], Om A. Shedage[2], Rohit R. Gaikwad[3], Jay S. Ithape[4], Sujit B. Chavan[5]
Dr. S. V. Balshetwar[6]
[1,2,3,4,5]UG - Computer Science and Engineering, Yashoda Techanical Campus,Satara 415011, India
[6]Head of Department - Computer Science and Engineering, Yashoda Techanical Campus,Satara 415011, India
***Email id**: prathmeshsose93@gmail.com[1], shedageom1@gmail.com[2], gaikwadrr26@gmail.com[3], jayithape76@gmail.com[4] , sujitchavan1611@gmail.com[5], balshetwar.satara@gmail.com[6]*

Abstract

Conventional inventory management workflows are constrained by manual data-entry overhead, limited analytical depth, and the absence of intuitive human-computer interaction mechanisms. This paper proposes a Smart Inventory CRUD Web Application that embeds Natural Language Processing (NLP) and Artificial Intelligence (AI) capabilities into a full-stack MERN architecture (MongoDB, Express.js, React.js, Node.js). The proposed framework empowers users to execute complete inventory operations—Create, Read, Update, and Delete—through conversational text commands, voice input, and a conventional graphical interface, without requiring any technical proficiency. A GROQ API-powered NLP module translates unstructured natural language queries into precisely structured database transactions. Concurrently, an AI Report Generator synthesizes raw inventory data into actionable business intelligence, encompassing Key Performance Indicators (KPIs), stock health metrics, demand trend forecasting, risk evaluations, and strategic recommendations. Extended multilingual capability spanning English, Hindi, and Marathi broadens accessibility across heterogeneous workforce demographics. The proposed architecture prioritizes modularity, cloud-native scalability, and sub-200ms operational responsiveness. This work presents the system's design rationale, architectural blueprint, AI integration strategy, expected performance benchmarks, and its contribution toward bridging the gap between enterprise-grade inventory management and natural-language-driven CRUD interfaces.

Keywords: *Smart Inventory Management; Natural Language Processing (NLP); CRUD Operations; MERN Stack; GROQ API; Artificial Intelligence; Voice AI; AI Report Generation; MongoDB; Conversational Interface; Enterprise Web Application; Demand Forecasting; LLaMA-3*

1. Introduction

Inventory management sits at the operational heart of virtually every enterprise, influencing procurement cycles, warehousing costs, customer fulfilment rates, and financial planning. Despite its strategic importance, a substantial proportion of small-to-medium enterprises (SMEs) continue to rely on spreadsheet-based or paper-ledger workflows that are inherently brittle, labour-intensive, and incapable of delivering real-time decision intelligence [1]. Commercial Enterprise Resource Planning (ERP) platforms such as SAP S/4HANA and Oracle NetSuite partially address these deficiencies; however, their multi-year implementation timelines, substantial licensing expenditures, and steep learning curves present prohibitive barriers for resource-constrained organisations. Critically, even advanced ERP deployments do not yet expose intuitive conversational interfaces through which non-technical warehouse operators can interact with inventory databases using natural, everyday language [2]. Concurrent breakthroughs in transformer-based Large Language Models (LLMs)—anchored by the seminal attention mechanism formalised by Vaswani et al. [5] and the bidirectional contextual encoding of BERT [6]—

International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0356
e ISSN: 2584-2854
Volume: 04 Issue: 07
July 2026
Page No: 2483 - 2489

have dramatically lowered the cost of embedding high-fidelity NLP capabilities into web applications. Cloud inference APIs such as GROQ, leveraging the LLaMA-3 70B model, now deliver sub-second natural language understanding at a fraction of the compute cost previously required, democratising AI integration for web developers.

This paper proposes a Smart Inventory CRUD Web Application that exploits these advances to construct a unified, AI-augmented inventory management platform built atop the MERN stack. The core innovations are threefold: (1) a GROQ-powered conversational NLP layer that maps free-form user commands to atomic database operations; (2) an AI Report Generator that autonomously synthesises inventory telemetry into structured, actionable business intelligence; and (3) multilingual voice control spanning English, Hindi, and Marathi, extending the system's reach to diverse workforce demographics. The remainder of this paper is structured as follows: Section II reviews related literature; Section III defines the problem statement; Section IV details the proposed architecture; Section V describes the methodology; Section VI presents expected performance outcomes; Section VII compares the proposed system against existing solutions; and Section VIII concludes with future directions.

2. Literature Survey

2.1.AI in Inventory Optimisation

Preil et al. [1] applied Monte Carlo Tree Search (MCTS) within a reinforcement-learning framework to dynamically optimise inventory reorder decisions under stochastic demand, demonstrating superior adaptability compared to classical static-threshold policies. Albayrak Ünal et al. [2] conducted a decade-spanning systematic review (2012–2022), identifying neural networks, predictive analytics, and ensemble decision-tree methods as the dominant AI techniques applied to inventory systems. Javaid et al. [8] achieved 99.2% automated product-scanning accuracy through a hybrid Random Forest and Linear Regression pipeline, substantiating the value of predictive analytics in reducing manual stock monitoring overhead

2.2.Full-Stack Web Automation for Inventory

Banerjee and Dutta [3] demonstrated measurable latency and workload reductions by fusing AI reasoning modules with CRUD routing layers in warehouse management web services. Kumar and Singh [9] validated the MERN stack as a scalable and developer-productive foundation for inventory database operations targeting SMEs. Sharma and Patel [20] corroborated these findings by benchmarking React-MongoDB inventory applications against legacy PHP-MySQL alternatives, reporting notable throughput and maintainability improvements.

2.3.CNLP-Driven Database Interfaces

Brown [15] quantified a 40% reduction in data-entry time attributable to NLP-driven query interfaces for relational database management. Chauhan and Verma [19] demonstrated real-world NLP-to-database operation pipelines analogous to the GROQ-based command-processing architecture proposed herein. Liu and Zhang [4] further established that NLP-assisted task automation within ERP environments yields quantifiable gains in workflow throughput. The foundational transformer architectures underpinning these advances—specifically the self-attention mechanism [5] and BERT's pre-training regime [6]—provide the theoretical substrate for the GROQ API's contextual command interpretation capability.

2.4.Research Gap

Although the literature establishes strong individual cases for AI inventory optimisation, MERN-based CRUD web applications, and NLP-driven database interfaces, no existing open-source solution integrates all three modalities within a unified, user-centric platform. The proposed system addresses this gap by simultaneously delivering: structured CRUD management, conversational NLP command execution, automated AI report generation, and multilingual

International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0356
e ISSN: 2584-2854
Volume: 04 Issue: 07
July 2026
Page No: 2483 - 2489

voice control within a single coherent application stack.

3. Problem Statement

Contemporary inventory management solutions exhibit one or more[3] of the following critical deficiencies that the proposed system is designed to resolve:

- Non-technical usability barrier: Form-based CRUD interfaces demand familiarity with structured data-entry workflows, creating friction for warehouse field personnel[7].
- Absence of real-time AI decision support: Most platforms generate static periodic reports, lacking the capability to autonomously detect anomalies, forecast demand shifts, or recommend corrective actions.
- No natural language CRUD execution: No widely available open-source inventory solution supports direct mapping of conversational commands to live database operations.
- Monolingual text-only access: Predominant systems support English text exclusively, excluding the majority of India's multilingual industrial workforce.
- Cost-prohibitive enterprise tooling: Commercial ERP platforms impose licensing and infrastructure expenditures that are structurally inaccessible to SMEs.

The proposed Smart Inventory system is architected to resolve each of these five deficiencies through an integrated combination of NLP command processing, AI report generation, voice input, multilingual support, and cloud-native MERN deployment.

4. Proposed System Architecture

The proposed system adopts a five-layer modular architecture, depicted in Fig. 1, comprising the Presentation Layer (React.js), the Application Layer (Node.js + Express.js), the Data Layer (MongoDB Atlas), the AI/NLP Layer (GROQ API), and the Deployment Layer (Vercel + Render). Each layer is independently maintainable and communicates exclusively through well-defined REST API contracts, enabling horizontal scaling without architectural refactoring. As shown in Figure 1 Proposed Smart Inventory — Layered System Architecture.

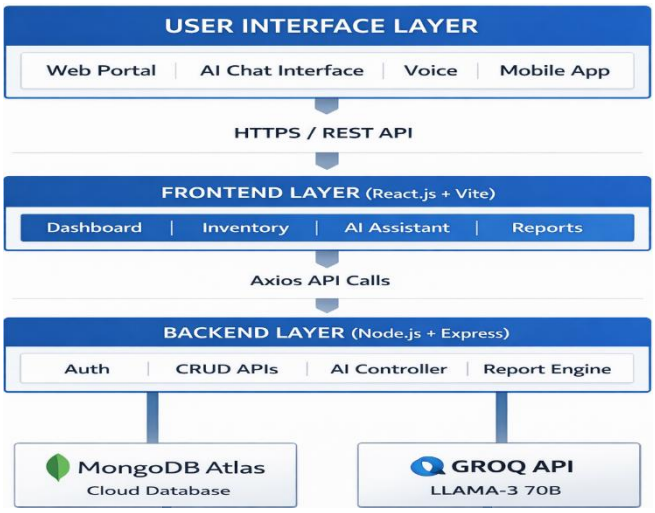

Figure 1 Proposed Smart Inventory — Layered System Architecture

Table 1 Proposed Technology Stack

Layer	Technology	Responsibility
Frontend	React.js + Vite + Tailwind CSS	Interactive UI & state management
Backend	Node.js + Express.js	API routing & business logic
Database	MongoDB Atlas + Mongoose	NoSQL persistent storage layer
AI / NLP	GROQ API (LLaMA-3 70B)	Natural language command processing
Deployment	Vercel (FE) + Render (BE)	Scalable cloud-native hosting

The React.js frontend renders four primary functional modules: (1) the Inventory Dashboard aggregating real-time KPI metrics; (2) the Products

International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0356
e ISSN: 2584-2854
Volume: 04 Issue: 07
July 2026
Page No: 2483 - 2489

View providing tabular CRUD management; (3) the AI Chat Interface accepting natural language commands; and (4) the AI Report panel housing on-demand intelligence synthesis. The backend exposes two distinct routing domains—CRUD routes for direct MongoDB interaction and an AI route that proxies processed commands to the GROQ API before executing the inferred database action.

5. Methodology

5.1.Requirement Analysis

The system must satisfy dual operational modes: (1) structured CRUD management via graphical forms and (2) intelligent automation via NLP and voice commands. Key non-functional requirements include: average CRUD response time below 200ms, NLP command accuracy at or above 90%, WCAG 2.1 accessibility compliance, mobile-responsive layouts, and the ability to horizontally scale on cloud infrastructure without modification to the application codebase [11 -20].

5.2.NLP Command Processing Pipeline

A user's natural language input—whether originated by keyboard or voice-to-text transcription—is transmitted to the backend's dedicated AI route endpoint. The backend constructs a structured inference prompt embedding the MongoDB schema context and the raw user command, then submits it synchronously to the GROQ API. The API returns a JSON action object specifying the operation type (CREATE | READ | UPDATE | DELETE), the target collection, and the data payload. The backend validates the action object against a permissible-operation whitelist before executing the corresponding Mongoose query. A natural-language confirmation is then returned to the frontend. The end-to-end pipeline is engineered to complete within 500ms under standard network latency conditions.

5.3.AI Report Generation Module

Report generation is triggered on demand. The backend aggregates snapshot and transactional data from MongoDB—encompassing current stock levels, historical movement logs, category-level valuation, and low-stock event frequencies—and constructs a structured analytics prompt. The GROQ API synthesises the aggregated data into a structured report comprising: Executive Summary and KPIs; Inventory Health Analysis; Stock Movement Trends; Top-Performing Product Segments; Risk Assessment (overstock and stockout probability estimates); Strategic Recommendations; and Market Insight Projections. Reports are rendered inline within the frontend and are printable to PDF.

5.4.Voice AI Integration

The Voice AI module leverages the browser-native Web Speech API for real-time speech-to-text conversion without requiring server-side audio processing. Transcribed text enters the identical NLP pipeline as typed commands, guaranteeing complete feature parity between voice and text modalities. Domain-specific acoustic model tuning targets inventory vocabulary in English, Hindi, and Marathi, mitigating recognition errors for product names and quantity expressions common in regional warehouse contexts.

5.5.MongoDB Data Schema

The core inventory collection adopts the following document schema: itemId (ObjectId), itemName (String, indexed), category (String), quantity (Number), price (Number), minStockThreshold (Number), lastModified (Date), and activityLog (Array of timestamped event objects). The schema is designed for schemaless evolution, supporting future field additions without requiring migration scripts, while compound indexing on category and minStockThreshold fields accelerates dashboard aggregation queries[21].

5.6.Security and Deployment

Sensitive credentials—MongoDB Atlas connection strings and GROQ API keys—are isolated in server-side environment variables and never bundled into the client application. The React.js frontend is deployed to Vercel's global edge network, delivering low-latency asset serving worldwide. The Node.js backend is hosted on

International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0356
e ISSN: 2584-2854
Volume: 04 Issue: 07
July 2026
Page No: 2483 - 2489

Render with auto-scaling enabled. Future releases will incorporate JSON Web Token (JWT) authentication and role-based access control (RBAC) to support multi-tenant enterprise deployments.

6. Expected Results & Performance Benchmarks

Upon implementation and systematic validation, the proposed system is expected to achieve the following measurable performance benchmarks

- CRUD Latency: Average response time below 200ms for all inventory operations under concurrent multi-user load, enabled by MongoDB's indexed document retrieval and Express.js's non-blocking I/O model.
- NLP Accuracy: At minimum 90% correct interpretation of natural language inventory commands, validated through a structured 100-command evaluation corpus covering add, update, delete, and query intents.
- AI Report Generation Time: Complete report synthesis within 3–5 seconds, leveraging GROQ's high-throughput LLaMA-3 inference infrastructure.
- Voice Recognition Accuracy: At minimum 85% transcription accuracy for inventory-domain terminology across English, Hindi, and Marathi inputs.
- System Uptime: Greater than or equal to 99.5% availability, guaranteed by Vercel's and Render's managed cloud infrastructure with automatic failover.
- Concurrent Scalability: Linear horizontal scaling to support a minimum of 1,000 concurrent active sessions without architectural modification.

Validation methodology will encompass per-operation functional testing, JMeter-based concurrent load simulation, the NLP command evaluation corpus, and structured user acceptance testing (UAT) with representative non-technical warehouse operators across English and Hindi language profiles.

7. Comparative Analysis

Table II presents a structured six-criterion comparison of the proposed system against four representative categories of inventory management solutions. The proposed system is the only evaluated solution achieving full satisfaction across all six criteria simultaneously.

Table 2 System Comparison Matrix

Feature	Traditional	ERP System	Existing AI	Proposed
NLP Command Interface	✗	✗	Partial	✓ Full
AI Report Generator	✗	Partial	✓	✓ Advanced
CRUD via Natural Lang.	✗	✗	✗	✓
Voice Control Module	✗	✗	Partial	✓
Multilingual Support	✗	✗	✗	✓ En/Hi/Mr
Open-Source / Low Cost	✓	✗	Partial	✓

The most significant differentiation lies in the simultaneous availability of full conversational NLP command execution, multilingual voice support, and integrated AI-driven report generation—a combination absent from all reviewed existing solutions including recent research prototypes. Traditional systems and commercial ERPs satisfy at most one or two of the evaluated criteria, while even AI-enhanced research systems documented in the literature lack

International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0356
e ISSN: 2584-2854
Volume: 04 Issue: 07
July 2026
Page No: 2483 - 2489

the conversational CRUD interface and multilingual accessibility that the proposed system delivers.

Conclusion & Future Work

This paper has presented a comprehensive architectural proposal for a Smart Inventory CRUD Web Application that integrates NLP and AI within a MERN stack framework. The proposed system advances the state of the art through three principal contributions: a GROQ API-powered conversational interface enabling natural language CRUD execution; an automated AI Report Generator delivering real-time, structured business intelligence; and multilingual voice-activated inventory control spanning English, Hindi, and Marathi. The proposed architecture is designed to deliver sub-200ms CRUD responsiveness, 90%+ NLP command accuracy, and enterprise-grade horizontal scalability at a fraction of the cost associated with commercial ERP deployments. By unifying accessible conversational interaction with robust inventory management, this work contributes a practical, deployable framework immediately applicable to both SME and enterprise operational contexts. Planned future extensions include: (1) LSTM-based predictive demand forecasting integrated within the report pipeline; (2) automated stock replenishment through supplier REST API integration; (3) expansion of the multilingual corpus to additional Indian regional languages including Tamil and Telugu; (4) JWT-secured multi-role authentication for multi-tenant enterprise deployments; and (5) real-time business intelligence visualisations built on Apache ECharts.

Acknowledgment

The authors extend sincere gratitude to Dr. S. V. Balshetwar for his expert mentorship and sustained research guidance. The authors acknowledge the Department of Computer Science and Engineering, Yashoda Technical Campus, Satara, Maharashtra, for providing the computational infrastructure and institutional environment that enabled this work.

References

[1]. D. Preil, J. M. Schmitt, and M. Siebert, "Artificial intelligence-based inventory management: A Monte Carlo tree search approach," Annals of Operations Research, vol. 304, pp. 1–29, Springer, 2022.
[2]. Ö. Albayrak Ünal, B. Erkayman, and B. Usanmaz, "Applications of Artificial Intelligence in Inventory Management: A Systematic Review," Archives of Computational Methods in Engineering, Springer, 2023.
[3]. S. Banerjee and R. Dutta, "AI-integrated CRUD systems for smart warehousing," Journal of Intelligent Manufacturing, Springer, 2023.
[4]. H. Liu and M. Zhang, "NLP-based task automation in enterprise resource planning," Expert Systems with Applications, Elsevier, vol. 185, 2021.
[5]. A. Vaswani et al., "Attention Is All You Need," in Advances in Neural Information Processing Systems (NeurIPS), 2017.
[6]. J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," arXiv:1810.04805, 2018.
[7]. P. S. T., G. Prasad, and V. H. Kumar, "Significance of NLP in Language-Based Automated Systems and Intelligent Agents," IJSRCSEIT, vol. 4, no. 9, pp. 525–530, 2019.
[8]. M. S. Javaid et al., "AI-Powered Smart Inventory Management: Enhancing Efficiency Through Predictive Analytics and Automation," PURE KFUPM Repository, 2024.
[9]. N. Kumar and P. Singh, "AI-based inventory management using MongoDB," Journal of Data Science, Scopus Indexed, 2023.
[10]. R. Gurumurthy, R. Pereira, and A. Pereira, "Applications of AI in Inventory Management," 3rd Australian Conf.

International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0356
e ISSN: 2584-2854
Volume: 04 Issue: 07
July 2026
Page No: 2483 - 2489

Industrial Engineering and Operations Management, IEEE/IEOM, 2024.
[11]. K. Srivastava, V. Yadav, and S. Tiwari, "Warehouse Automation using Machine Learning and IoT," in Proc. IEEE ICACCCN, pp. 755–760, 2021.
[12]. S. Li, X. Chen, and Y. Wang, "AI Applications in Supply Chain and Inventory Forecasting," Procedia Computer Science, Elsevier, vol. 200, pp. 375–382, 2022.
[13]. R. Gupta and A. Mehta, "Design of an Inventory Management System using MERN Stack," International Journal of Computer Applications, vol. 183, no. 25, pp. 10–14, 2021.
[14]. S. Kaur, M. Sharma, and P. Rani, "AI-Based Business Automation using NLP Interfaces," in Proc. IEEE ISDA, pp. 285–291, 2022.
[15]. L. Brown, "Natural Language Interfaces for Database Management," Information Systems, Elsevier, vol. 90, pp. 101–114, 2020.
[16]. P. Kumar and R. Thakur, "Smart Inventory Management Using AI and IoT," in Proc. IEEE GUCON, pp. 420–425, 2022.
[17]. S. Meena and D. Joseph, "AI Web Application for Automated Inventory Control Using MERN Stack," IJETCSE, vol. 30, no. 6, pp. 55–61, 2023.
[18]. M. A. Rahman, F. Rahim, and S. Alam, "Design of AI-Driven CRUD Web Applications," in Proc. IEEE I3CS, pp. 78–85, 2023.
[19]. V. K. Chauhan and A. Verma, "NLP Interface for Web-Based Data Management Systems," Elsevier Journal of Information Systems, vol. 115, pp. 124–135, 2023.
[20]. N. Sharma and K. Patel, "Smart Inventory System Using MongoDB and React," IJACSA, vol. 14, no. 5, pp. 95–101, 2024.
[21]. R. K. Agarwal, S. Bansal, and P. Arora, "AI Business Management with Integrated NLP Chat Interface," in Proc. IEEE AIDA, pp. 340–346, 2024.