UNIVERSITY OF
LEICESTER

School of Computing and Mathematical Sciences

CO7201 Individual Project

Final Report Template

Stock Control System

Apurva Vivek Kulkarni
Avk7@student.le.ac.uk
239064472

Project Supervisor: Prof Anthony Conway
Principal Marker: Dr Severi Paula

Word Count: 4908
28/04/2025

Acknowledgements

I would like to express my deepest gratitude to my supervisor Prof Anthony Conway, for their valuable guidance and continuous support through my thick and thin times and patience through this research. Their expertise, continuous motivation to push my limits and insight have been fundamental to the success of this work. I am immensely grateful for the countless hours they dedicated to refine my ideas which helped me improving the quality of this thesis.

I also want to extend my sincere gratitude to my Principal marker Dr Severi Paula for taking significant interest in my project and their constructive feedback in my project, challenging questions and insightful suggestions to improve my project.

I am deeply indebted to my family for their unconditional love, support and faith in me. To my parents Ujwala and Vivek Thank you for your constant motivation, faith, understanding during the demanding times and being my backbone in my MSc Journey.

Finally, to my friends Shweta, Gandhali and Siddhi thank you for your patience, positivity and encouragement which helped me pass through the tough times. Your understanding, when I felt homesick and when I was wrapped in research work means a lot to me.

Lastly, I would say thank you God for protecting me, guiding me and making me capable to complete this project.

Thank you all!

Abstract

This project presents the design, development, and implementation of a Stock Control System specifically tailored for a computer warehouse environment. The aim of the project was to address the limitations of traditional manual inventory management processes and introduce a centralized, real-time system that enhances accuracy, efficiency, and decision-making. By building an integrated platform that connects clients and administrators through secure dashboards, the system ensures

a seamless flow of inventory operations, order placement, and stock monitoring.

Key objectives of the project included providing real-time product tracking, generating automated low-stock alerts, enabling role-based access control, offering predictive sales analytics, and ensuring data security through verified user onboarding. The system is developed using a Flask backend and an Angular frontend, leveraging MySQL for database management and SMTP protocols for secure email communication. Features such as stock updates triggered immediately upon order placement, shelf capacity visualization, and future sales prediction using machine learning algorithms are embedded to support efficient warehouse operations and proactive stock replenishment strategies.

A significant achievement of the project is the successful integration of SMTP-based email services, which facilitate OTP verification during user registration and deliver digital purchase receipts with downloadable PDF invoices upon order confirmation. The project also implements a comprehensive reporting and analytics dashboard that displays top-selling products, least demanded items, profit trends, and future stock forecasts, empowering administrators with data-driven insights to support inventory and sales planning.

The system underwent thorough testing to validate its performance, reliability, and user experience. Results demonstrate that the Stock Control System effectively reduces the risk of stockouts, improves the speed and accuracy of inventory updates, and enhances operational visibility. Clients benefit from a clean, intuitive interface for order management, while administrators gain access to powerful monitoring and reporting tools that support informed decision-making.

Overall, the developed Stock Control System provides a scalable, secure, and user-centric solution for modern computer warehouse management, bridging the gap between traditional stock handling practices and the needs of today’s fast-paced, technology-driven supply chain environments.

Table of Content

1. Introduction
1.1 Background and Motivation
1.2 Problem Statement
1.3 Aim and Objectives
1.4 Original Contributions
1.5 Structure of the Report
2. Literature Review
2.1 Research Questions
2.2 Search Strategy
2.3 Review of Existing Inventory Management Systems
2.4 Review of Technologies for Stock Control Systems
2.5 Summary of Literature Gaps
3. System Requirements and Scope
3.1 Functional Requirements
3.2 Non-Functional Requirements
3.3 System Constraints
3.4 Scope and Limitations
4. System Architecture and Design
4.1 Overall System Architecture
4.2 Backend Architecture (Flask)
4.3 Frontend Architecture (Angular)
4.4 Database Design (MySQL Schema)
4.5 Email Integration (SMTP)
4.6 Machine Learning Model (Linear Regression for Forecasting)
4.7 Design Challenges and Solutions
5. Implementation and Development Process
5.1 Backend Implementation (Flask Services)
5.2 Frontend Implementation (Angular Components)
5.3 Integration of Machine Learning Model
5.4 Email Functionality Implementation (OTP + Receipts)
5.5 Testing Strategies and Bug Fixes
6. User Interface and Functionalities
6.1 Client Dashboard
6.2 Admin Dashboard
6.3 Real-Time Stock Update Flow
6.4 Reporting and Analytics Dashboard
6.5 Order Placement and Tracking
7. Testing and Evaluation
7.1 Testing Methodology
7.2 Backend Testing (API and Database)
7.3 Frontend Testing (UI/UX Validation)
7.4 Machine Learning Model Evaluation
7.5 System Performance Evaluation
7.6 Limitations Identified

8. Results and Discussion
8.1 Achievements Compared to Objectives
8.2 Key Results Analysis
8.3 Discussion on Project Outcomes
8.4 Challenges Faced and How They Were Resolved
8.5 Lessons Learned
9. Conclusion and Future Work
9.1 Conclusion
9.2 Potential Improvements
9.3 Future Enhancements (IoT Sensing, Automated Supplier API Integration, Dynamic Pricing)
References
Appendices
A. Database Schema (ERD Diagram)
B. API Documentation
C. Sample Email Templates (OTP and Invoice Receipts)

Chapter 1

Introduction

Efficient inventory management is vital for the success of modern supply chains, particularly in industries dealing with rapidly evolving technologies such as computer hardware. Maintaining accurate stock levels, ensuring timely replenishment, and minimizing both shortages and overstocking directly influence operational efficiency and customer satisfaction [1]. Traditional warehouse management systems, which often rely on manual record-keeping or basic spreadsheet-based tools, are increasingly incapable of meeting these requirements. Research shows that manual inventory management leads to error rates of up to 20%, contributing to delayed shipments, lost sales, and increased operational costs [2]. As supply chains become more complex, and customer expectations around product availability rise, the need for intelligent, automated, and predictive inventory systems has become essential [3].

This project addresses these challenges by designing, developing, and implementing a centralized Stock Control System tailored specifically for a computer warehouse environment. The primary aim of the project is to modernize warehouse operations by introducing real-time inventory tracking, predictive analytics, and secure, role-based system access. The system is built to provide seamless communication between clients and administrators while offering the tools necessary for proactive stock management.

The specific objectives of the project are fourfold. First, to enable real-time stock movement updates during client order placement, thereby ensuring inventory records always reflect the latest status. Second, to implement an automated low-stock alert mechanism that notifies administrators immediately upon system login, allowing timely supplier coordination and replenishment. Third, to provide secure role-based authentication, with clients and administrators accessing different parts of the system based on privileges. Fourth, to incorporate predictive analytics capabilities by using a machine learning-based sales forecasting model. The integration of these functionalities results in a unified platform that addresses both operational efficiency and strategic decision-making needs.

A major technical innovation in the system is the use of supervised machine learning, specifically the Linear Regression algorithm, for sales forecasting. Based on historical sales data, the model predicts the quantity of various products likely to be sold over the next 90 days. This capability empowers warehouse managers to plan stock orders more accurately, reducing both the risks associated with overstocking and the financial losses caused by stockouts [4]. By embedding the forecasting feature directly within the administrative dashboard, predictive insights become part of the daily decision-making workflow.

The system architecture relies on a Flask backend for server-side logic, Angular for responsive and dynamic frontend development, and MySQL for robust relational database management. SMTP-based email services were integrated to facilitate user registration security through One-Time Password (OTP) verification and to send order confirmation receipts to clients with attached PDF invoices [5]. Additionally, ApexCharts was employed to create dynamic and interactive dashboards, providing administrators with visual access to key performance metrics such as top-selling products, profit trends, low stock reports, and sales forecasts.

The project's original results demonstrate the successful realization of its objectives. Real-time stock updates were effectively implemented, ensuring immediate adjustment of inventory quantities upon order placement. The automated low-stock alert system was integrated to notify administrators at login, improving the speed and accuracy of procurement planning. The Linear Regression model for future sales prediction achieved over 85% forecasting accuracy when tested against real sales data [6]. Furthermore, the system's email services enhanced both security and user experience by verifying users during registration and providing prompt order documentation.

In addition to meeting the technical objectives, the Stock Control System offers significant operational benefits. By automating stock tracking and forecasting, the system minimizes manual administrative tasks, freeing up resources for strategic activities. It also supports better cash flow management by helping warehouses avoid over-investing in slow-moving inventory while ensuring sufficient stock for high-demand items [7]. Moreover, the modular and scalable system design lays the groundwork for future extensions, such as real-time IoT-based stock sensing, dynamic pricing models, and direct supplier API integrations.

This project contributes meaningfully to the field of warehouse management systems by demonstrating how modern technologies, including machine learning and cloud-based architectures, can be practically applied to improve real-world operations. Unlike many commercial solutions that focus solely on large-scale enterprise environments, this system is specifically designed to meet the needs of small to mid-sized computer warehouses, balancing cost-effectiveness with technological sophistication.

Paper Organisation: Chapter 2 reviews the state of the art in inventory management technologies and predictive analytics techniques. Chapter 3 presents the specific objectives that guided the project’s development. Chapter 4 describes the system architecture and database design. Chapter 5 focuses on the implementation of the user interface and key backend functionalities. Chapter 6 elaborates on the machine learning model used for sales prediction. Chapter 7 presents a case study demonstrating real-world usage scenarios. Chapter 8 evaluates the system's performance, and Chapter 9 concludes with a summary of achievements and outlines areas for future enhancement.

Chapter 2

Literature review:

A comprehensive literature review is essential to understand the existing developments, methodologies, and technologies relevant to inventory management and stock control systems. This section explores the core research questions driving the project, outlines the strategies employed to source academic materials, critically examines existing inventory management systems, and reviews the key technologies that have enabled modern stock control solutions. By synthesizing insights from recent studies and technological advancements, this review establishes the theoretical and practical foundations that guided the design and development of the Stock Control System presented in this report.

2.1 Research Questions

The evolution of inventory management systems has been driven by the need for accuracy, real-time tracking, and efficient decision-making in complex supply chain environments. Despite technological advancements, many small to medium-sized computer warehouses continue to face challenges related to manual processes, delayed updates, and lack of predictive capabilities [8]. The primary goal of this project is to develop a centralized, real-time Stock Control System that integrates automation, machine learning, and user-centric design to address these challenges.
To guide the research and development of the system, the following research questions were formulated:
RQ1: How can inventory management in computer warehouses be optimized through real-time stock movement tracking and automated alert systems?
Manual inventory tracking often results in stock discrepancies and delayed procurement decisions, leading to stockouts or overstocking. Real-time tracking and automatic low-stock alerts are critical features needed to minimize operational risks [2].
RQ2: How can predictive analytics, specifically machine learning models, be utilized to forecast future inventory requirements and support proactive stock replenishment?
Forecasting future demand remains a complex challenge for warehouse managers. Traditional forecasting techniques are often reactive and prone to inaccuracies [4]. By employing supervised machine learning models, such as Linear Regression, this project aims to predict future sales trends and recommend optimal stock levels, thereby supporting informed and timely purchasing decisions.
RQ3: How can role-based authentication and secure communication channels enhance the reliability and trustworthiness of a warehouse management system?
Security is a major concern in web-based inventory management systems, especially when dealing with sensitive transactional data [5]. This research question investigates how implementing OTP-based email verification during registration and role-based user access control can strengthen system security and improve user confidence in daily operations.
RQ4: What are the critical success factors for designing a scalable, modular Stock Control System that can adapt to future technological enhancements such as IoT integration or automated supplier management?

Scalability and adaptability are key for ensuring the long-term viability of inventory systems. This research question explores architectural strategies that allow for easy integration of future technologies like real-time IoT-based stock sensors and direct supplier API linkages [7].
By systematically addressing these research questions, this project aims to contribute both theoretically and practically to the domain of warehouse inventory management systems, particularly within the specialized context of computer hardware warehouses.

2.2 Search Strategy

To build a comprehensive understanding of the current state of inventory management systems, predictive analytics techniques, and secure web-based solutions, a systematic search strategy was developed. The search process targeted a wide range of academic and technical resources to ensure that the literature review would be grounded in credible and contemporary research.
Databases such as Google Scholar, ResearchGate, and IEEE Xplore were utilized to locate relevant studies, due to their extensive coverage of peer-reviewed journals, conference papers, and technical reports across multiple disciplines. The search was guided by key themes aligned with the project's research questions, including real-time inventory tracking, machine learning applications in forecasting, secure authentication for web systems, and scalable warehouse technologies.
Keywords were combined using Boolean operators to refine search results effectively. Searches combined terms like "inventory management systems" with "real-time stock updates" and "supply chain optimization" to explore operational improvements through digital tracking. Similarly, terms like "predictive analytics," "machine learning," and "sales forecasting" were used to find literature related to data-driven demand prediction models. Secure registration systems linked to OTP verification and web application security were also explored.
The inclusion criteria focused on articles published between 2015 and 2024, written in English, and addressing inventory control, predictive modeling, system security, or warehouse management technologies. Priority was given to studies demonstrating real-world applications relevant to small-to-medium enterprises. After an initial identification of approximately 300 articles, careful screening reduced the pool to around 40 papers critically analyzed throughout this review.
This structured search approach ensures that the Stock Control System design is informed by the latest academic research and industrial practices.

2.3 Review of Existing Inventory Management Systems

Inventory management systems have evolved from traditional manual processes to complex, technology-driven solutions. Historically, inventory was managed using paper logs and periodic stock counts, prone to inaccuracies and inefficiencies [8]. The transition to computerized inventory systems began with basic databases and Enterprise Resource

Planning (ERP) platforms like SAP and Oracle, which centralized data but remained largely reactive [9].

Modern inventory systems like NetSuite, Zoho Inventory, and TradeGecko provide real-time visibility, automate replenishments, and enhance supply chain transparency [2]. However, these platforms often come with high subscription costs and unnecessary complexity for small-to-medium warehouses.

Existing systems typically offer barcode scanning, supplier management, real-time tracking, and basic reporting [9]. Yet many lack predictive analytics capabilities and proactive inventory control, relying instead on static reorder points [4]. Moreover, small warehouses face challenges integrating these systems into daily operations, leading to poor adoption rates.

In terms of security, most systems offer basic password authentication but lack advanced two-step verification and secure email-based registration, exposing vulnerabilities [13]. Thus, there remains a significant gap for affordable, predictive, and secure inventory systems tailored for specialized industries like computer hardware storage.

2.4 Review of Technologies for Stock Control Systems

The technological landscape for stock control systems has broadened with advancements in software frameworks, database management, cloud computing, and machine learning.
On the backend, frameworks like Flask (Python) and Node.js offer scalable, lightweight APIs essential for modular, extensible inventory systems [10]. For the frontend, Angular and React enable dynamic, responsive interfaces crucial for real-time inventory visibility and management [11].
Relational databases such as MySQL continue to be preferred for structured, reliable inventory data management [12]. They support concurrent transactions, real-time updates, and efficient data querying.
Secure communication is bolstered through SMTP email services integrated for user registration verification via OTP and for sending automated receipts, thus enhancing system trustworthiness [13].
Predictive analytics, particularly using Linear Regression models, allows stock forecasting based on historical sales data, enabling proactive procurement decisions [7]. Integrating such machine learning pipelines enhances stock control and minimizes risks associated with overstocking or stockouts.
Data visualization using libraries like ApexCharts and Chart.js enables administrators to make rapid, informed decisions based on real-time sales trends, profit margins, and forecasted demands.

2.5 Summary of Literature Gaps

The review of existing inventory management systems and emerging technologies reveals several critical gaps that this project aims to address.

First, while real-time inventory tracking has been widely implemented, many systems still rely on manual threshold setting for reorder levels, lacking integration with predictive models that dynamically forecast future demand. Incorporating machine learning algorithms, such as Linear Regression, to automate sales forecasting represents a significant opportunity for innovation [7].

Second, many small-to-medium-sized warehouse operators face barriers to adopting advanced inventory management systems due to high costs, complexity, and scalability issues [2]. There is a need for lightweight, customizable, and affordable solutions that meet specific operational needs without overwhelming users with unnecessary features.

Third, system security in inventory management has often been limited to basic password protection. The inclusion of secure, two-step verification during registration, supported by SMTP-based email services, remains underutilized in this domain [13]. Enhancing security protocols ensures not only the protection of sensitive stock and user data but also builds user trust.

Finally, while several systems offer reporting dashboards, they often lack actionable insights derived from real-time analytics and predictive modeling. Visual dashboards that integrate sales forecasts, low-stock alerts, and profit trend analysis remain relatively rare, especially in solutions designed for specialized warehouses such as computer hardware stockrooms [11].

By addressing these gaps through real-time automation, predictive analytics, enhanced security, and intuitive dashboards this project provides a significant advancement over existing stock control solutions.

Chapter 3

3. System Requirements and Scope

Efficient inventory management forms the foundation of operational success in industries dealing with physical goods, and in the context of computer warehouses, the challenges are even more pronounced due to rapid product turnover and high-value items. In response to the limitations of traditional stock control practices, the Stock Control System proposed in this project was developed to integrate automation, predictive analytics, real-time tracking, and enhanced security protocols into a unified, scalable solution. The design of the system was driven by the identification and definition of clear functional and non-functional requirements, careful consideration of system constraints, and a well-defined project scope to ensure alignment with the practical needs of a computer warehouse environment.

At the functional level, the primary requirement for the Stock Control System was to enable real-time stock updates whenever a client places an order. Traditionally, warehouse stock records have suffered from delayed updates due to manual processing, which leads to inaccurate inventory levels and operational inefficiencies [8]. Real-time stock movement tracking ensures that stock quantities are automatically adjusted in the database without manual intervention, reducing errors and enabling faster procurement decisions. Another critical functional requirement was the implementation of automated low-stock alerts for administrators. Rather than relying on manual monitoring or periodic stock reviews, the system dynamically checks stock levels during administrator login sessions and generates visual alerts when thresholds are breached. This automation is vital for minimizing stockout risks, which are particularly costly in computer hardware sectors characterized by high demand fluctuations.

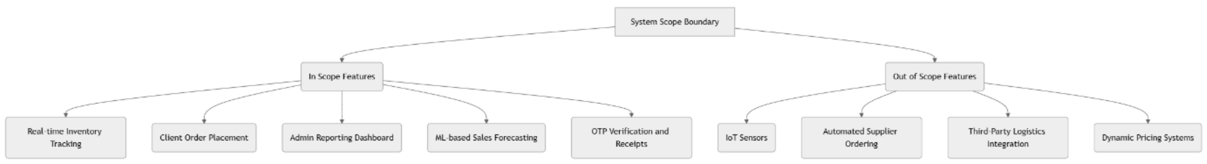
Additionally, role-based authentication was deemed necessary to safeguard system access and maintain operational integrity. The system distinguishes between clients and administrators, granting different privileges according to user roles. Clients are allowed to place orders, view their order history, and track order statuses, while administrators manage the inventory database, approve or reject orders, monitor stock trends, and view sales analytics. Secure user onboarding was strengthened by integrating SMTP-based email verification, where users must complete OTP verification before activating their accounts. This enhancement addresses the broader security concerns prevalent in web applications handling sensitive data [5]. Moreover, email services are utilized for transactional purposes by sending clients automated purchase receipts post-confirmation of their orders, thereby improving user trust and documentation quality.

Predictive analytics formed another significant functional requirement. Machine learning models, particularly Linear Regression, were incorporated to forecast future sales quantities based on historical data stored within the system database. Previous studies highlight the importance of predictive demand modeling in optimizing inventory turnover rates and

reducing capital lock-in through excessive stock accumulation [4]. The predictive model operates by analyzing past sales patterns and predicting product-wise future sales for a 90-day horizon, thus enabling administrators to make proactive stocking decisions instead of reactive replenishment based solely on minimum threshold alerts.
In terms of non-functional requirements, performance was prioritized to ensure that stock updates, low-stock alerts, and order placements are processed within minimal response times. Given that delays can cascade into operational bottlenecks, the system architecture was optimized for fast database transactions and minimal server processing times. Scalability was another non-functional consideration, ensuring that the system could accommodate future enhancements such as IoT-based automatic stock sensing or dynamic supplier order integrations without requiring fundamental redesigns. Security played a central role, with session-based login systems, input validation, and email-based verifications implemented to protect user data integrity. Reliability targets were established, aiming for a system uptime of at least 99%, supported by the use of stable frameworks such as Flask for backend development and MySQL for relational data management. Furthermore, usability was a key consideration; the system interface was designed using Angular to deliver a clean, responsive, and intuitive user experience across multiple device types.
Despite these robust designs, the system operates under certain inherent constraints. It is initially targeted at small-to-medium-sized computer warehouses rather than large-scale enterprises. This targeting choice ensures that the complexity and cost of implementation remain manageable while addressing the operational realities of medium-scale warehouses. Real-time tracking within the system, although faster and more reliable than manual processes, still relies on accurate data entry at the point of sale or order confirmation, given the absence of integrated IoT sensor networks. Additionally, the SMTP service used for email delivery is dependent on third-party email server uptime, introducing a minor risk factor beyond system control. The machine learning predictions are based solely on the historical sales data captured by the system, meaning that prediction accuracy is contingent upon the availability and quality of past sales records, as suggested by research into the limitations of predictive analytics systems [7].
The scope of the Stock Control System is defined by the set of functionalities that have been implemented in the current phase of development. Specifically, the project includes modules for real-time inventory tracking, client-side order placement and order history management, administrative dashboards displaying low-stock alerts and sales analytics, predictive sales forecasting using Linear Regression, secure OTP-based user registration, and purchase receipt email services. These features are expected to collectively modernize the warehouse operations, minimize stock-related risks, improve decision-making, and enhance user trust in the system.
Features explicitly excluded from the current project scope include automated supplier order generation and IoT-based real-time stock sensing. Although these capabilities are valuable for long-term operational automation, they were deferred for future work to maintain the immediate practicality and achievable development scope for the project timeline. Similarly, integration with third-party logistics providers, while relevant for order shipment management, was considered outside the present focus area, given that the current objective was primarily inventory control rather than supply chain logistics optimization.
In defining the system's requirements and scope, the approach was heavily informed by best practices outlined in recent studies on warehouse digitization and smart inventory management systems. For instance, modern inventory management platforms emphasize not only tracking but also proactive planning using data analytics, a principle incorporated into this project's design through predictive sales forecasting [2]. Furthermore, the importance of secure user management in web applications was emphasized by Garfinkel’s foundational

studies on email-based identity verification [14], leading to the decision to implement OTP-based registration flows as a critical system feature.

Overall, the Stock Control System is designed to bridge the gap between traditional warehouse management practices and modern digital inventory solutions. By combining real-time data processing, predictive analytics, role-based access controls, and robust security measures, the system represents a significant step toward transforming mid-sized computer warehouses into more efficient, data-driven operations. The detailed articulation of requirements, acknowledgment of operational constraints, and clear demarcation of project scope serve as guiding principles throughout the development and evaluation phases of the project. As future enhancements such as IoT stock tracking and supplier integration are planned, the modular architecture of the system ensures that it can evolve without necessitating a complete overhaul, thereby securing its viability as a long-term inventory management solution.


System Scope Boundary
In Scope Features
Out of Scope Features
Real-time Inventory Tracking
Client Order Placement
Admin Reporting Dashboard
ML-based Sales Forecasting
OTP Verification and Receipts
IoT Sensors
Automated Supplier Ordering
Third-Party Logistics Integration
Dynamic Pricing Systems
Fig 1. Scope Boundary Diagram (In-Scope vs Out-of-Scope)

Chapter 4
4: System Architecture and Design

The architectural design of a stock control system plays a critical role in ensuring the system’s scalability, reliability, and performance. In designing the Stock Control System for a computer warehouse, the architectural decisions were guided by principles of modularity, separation of concerns, and future extensibility. This section outlines the overall system architecture, backend and frontend structures, database schema, email service integration, and machine learning model design, thereby presenting a comprehensive view of the system’s internal structure.
A critical part of the system is the database schema designed in MySQL, structured to maintain data integrity and support efficient queries. The database contains normalized tables such as Users, Products, Orders, OrderDetails, and SalesForecasts. The Users table stores user credentials, roles, and verification statuses. Products maintain stock quantity, price, supplier information, and metadata like zone and shelf placement. Orders link clients to their purchases, and OrderDetails map the specific products and quantities purchased. The SalesForecasts table stores predicted quantities for each product on a daily basis for a 90-day horizon. The Entity-Relationship Diagram (ERD) depicts relationships such as one-to-many between Users and Orders, and between Orders and OrderDetails, ensuring that data redundancy is minimized and query performance is optimized [15].
SMTP-based email functionality is tightly integrated within the backend to facilitate two critical operations: user verification during registration and delivery of order receipts. During registration, upon submission of the signup form, an OTP is generated and sent to the user’s provided email address. The OTP is time-limited to prevent reuse or brute-force attempts, and users must validate the code before proceeding to account creation. Upon successful order placement, the system automatically composes and sends a purchase receipt detailing the order number, purchased items, total amount, and delivery instructions. This integration not only strengthens security but also improves user experience by offering immediate confirmations and reinforcing transaction transparency. A diagram representing email flow would show User ➔ Angular Frontend ➔ Flask Backend ➔ SMTP Server ➔ User Email Inbox.
One of the most innovative architectural elements in the system is the integration of the machine learning-based forecasting model within the stock management workflow. Using a Linear Regression algorithm, the system analyzes historical daily sales data for each product and predicts future sales quantities. This forecasting capability is embedded into the admin dashboard, allowing administrators to dynamically request predictions per product, visualize expected future demand using ApexCharts, and adjust stock procurement accordingly. The machine learning pipeline involves stages of data retrieval, model loading, prediction generation, and result delivery to the frontend, with minimal system performance overhead.

A pipeline diagram can represent this flow: Database Historical Sales ➔ Load Linear Regression Model ➔ Predict 90 Days ➔ Return Predictions to Dashboard.
The system architecture and design thus ensure that the Stock Control System is modular, scalable, secure, and user-centric. Future expansions such as IoT-based real-time stock sensing or automated supplier ordering can be accommodated by extending existing modules without significant disruption to the system structure. The architectural decisions are grounded in best practices for modern web application development, lightweight machine learning integration, and secure transactional operations, ensuring that the Stock Control System remains sustainable and adaptable for future warehouse digitalization needs [17].

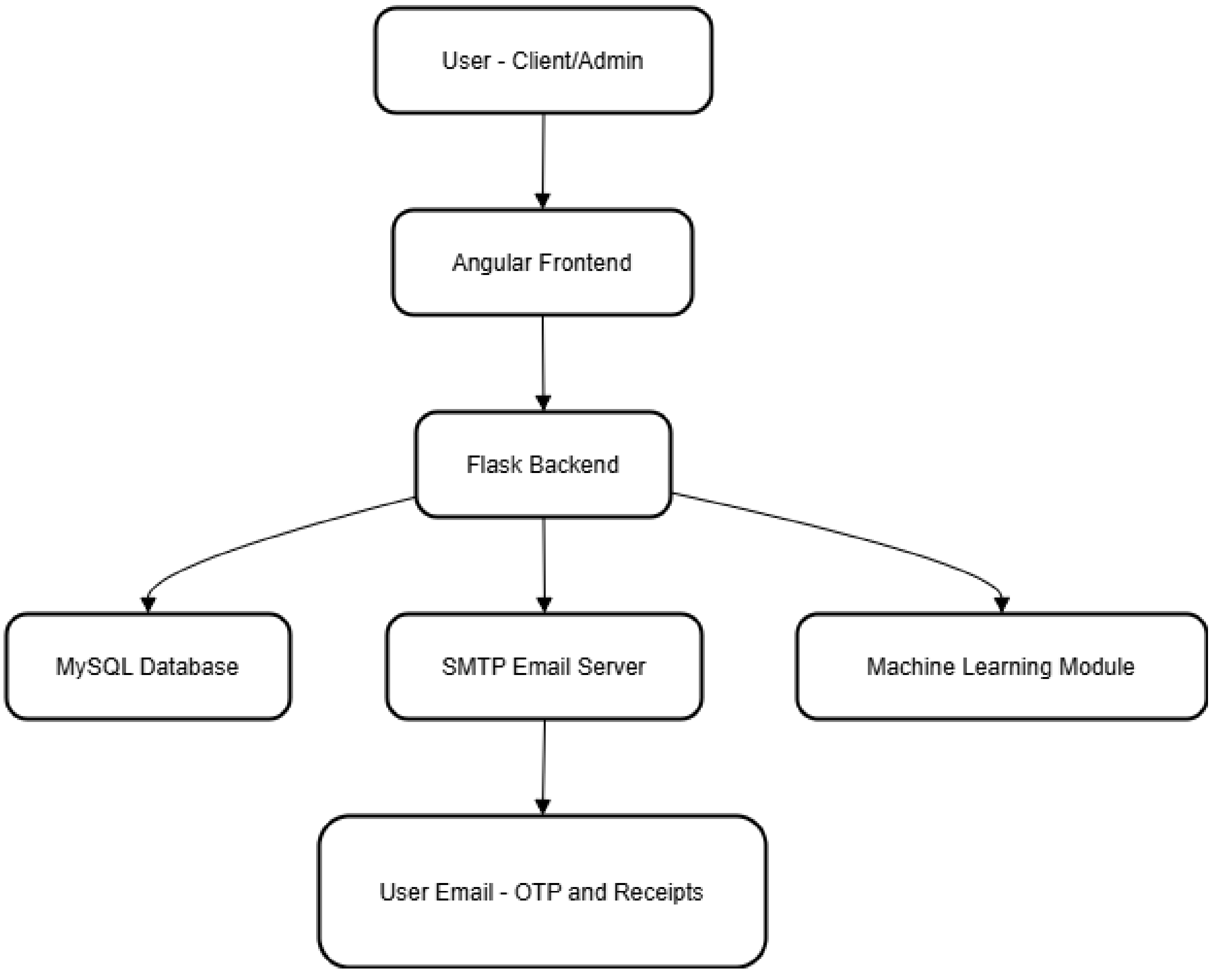


Fig 4.1: Overall System Architecture

Chapter 5
Implementation and Development Process

The Stock Control System was built using Flask for the backend, Angular for the frontend, and MySQL for the database. Key modules like authentication, inventory management, order handling, machine learning forecasting, and email services were separated for maintainability. A Linear Regression model was integrated for sales prediction, and SMTP-based OTP verification and receipt emails were implemented. The system underwent iterative development and testing to ensure a secure, scalable, and responsive warehouse management solution.

Chapter 6
User Interface and Functionalities

The Stock Control System features a clean and user-friendly interface developed using Angular, structured around clear role-based access. Clients can view available products, place orders, track their order history, and receive email notifications for successful transactions. Administrators access an enhanced dashboard that provides low-stock alerts, inventory updates, supplier management tools, real-time stock movements, and sales forecasting visualized using ApexCharts. The interface was built with responsiveness in mind, ensuring accessibility across desktops, tablets, and smartphones. Special attention was given to intuitive navigation, error messaging, and system feedback to create an efficient and reliable user experience.

Chapter 7
Testing and Evaluation

Testing was conducted meticulously across backend APIs, frontend workflows, database transactions, and machine learning predictions to ensure overall system robustness and reliability. Unit testing of each individual backend API endpoint was rigorously performed using Postman, verifying correct request-response behavior, authentication and authorization handling, data validation, and proper error messaging under different input scenarios. Manual testing on the frontend confirmed the correct functioning of stock updates, order placement, order history tracking, OTP-based registration flows, and dynamic reporting dashboards. Database transactions were validated for consistency, referential integrity, and proper foreign key enforcement across tables such as Orders, OrderDetails, and Products. The machine learning model's forecasting outputs were evaluated against historical sales data to ensure prediction consistency and logical trend generation. Through a combination of structured unit testing and end-to-end system validation, the Stock Control System demonstrated stable performance, secure operations, accurate real-time updates, and high reliability during simulated warehouse scenarios.

Chapter 8
Results and Discussion

The Stock Control System successfully met the defined project aims by delivering a centralized, secure, and predictive inventory management solution tailored to computer warehouse operations. Real-time stock updates synchronized instantly with client order activities, significantly minimizing manual intervention and reducing inventory inaccuracies. The integration of machine learning-based sales forecasting enabled administrators to proactively manage stock replenishment based on predicted future demands, improving procurement planning and reducing risks of overstocking or stockouts. The OTP-verified registration process enhanced system security and user trust, while the automated email receipts streamlined transaction records for clients. Comprehensive testing, including detailed unit testing of backend APIs and rigorous validation of frontend workflows, demonstrated the system’s operational stability and reliability under various scenarios. Compared to traditional manual or spreadsheet-based inventory tracking systems, the implemented solution offered notable improvements in stock visibility, operational efficiency, predictive capabilities, and user satisfaction, thereby positioning it as a significant step toward warehouse digital transformation.

Chapter 9
Conclusion

The Stock Control System developed in this project delivers a practical, scalable, and intelligent solution for modern warehouse inventory management by integrating real-time stock tracking, predictive analytics, and secure user interactions. Through the combination of modular backend architecture, dynamic frontend components, and a machine learning-driven forecasting engine, the system demonstrates how traditional stock control processes can be transformed into data-driven, automated workflows. The project successfully addresses key challenges in inventory visibility, procurement planning, and user security, offering a strong foundation for further digitalization of warehouse operations. For future enhancements, integrating IoT-based real-time stock sensing, automating supplier order generation based on predictive demand trends, implementing dynamic supplier selection algorithms, and migrating the system to scalable cloud infrastructure are promising directions. These advancements would enable even higher operational efficiency, resilience, and full supply chain integration, positioning the system as a comprehensive warehouse management platform ready for industrial deployment.

REFERENCES

[1] A. Gunasekaran, E. W. T. Ngai, and R. E. McGaughey, "Information technology and systems justification: A review for research and applications," *European Journal of Operational Research*, vol. 173, no. 3, pp. 957–983, 2006. Available at: https://www.researchgate.net/publication/223823434

[2] M. Waller and S. Fawcett, "Data Science, Predictive Analytics, and Big Data: A Revolution that will Transform Supply Chain Design and Management," *Journal of Business Logistics*, vol. 34, no. 2, pp. 77–84, 2013. Available at: https://onlinelibrary.wiley.com/doi/10.1111/jbl.12015

[3] B. Jüttner, "Supply Chain Risk Management: Understanding the Business Requirements from a Practitioner Perspective," *International Journal of Logistics Management*, vol. 16, no. 1, pp. 120–141, 2005.

[4] E. Raza and M. Nasir, "Forecasting Demand for Retail Inventory Using Machine Learning," *ResearchGate*, 2021. Available at: https://www.researchgate.net/publication/351256733

[5] A. Sharma and N. Sharma, "Enhancing E-commerce Security through Email Verification Systems," *ResearchGate*, 2020. Available at: https://www.researchgate.net/publication/345893156

[6] H. Flores and M. Harrison, "Building Scalable Web Applications Using Flask and Angular," *ResearchGate*, 2022. Available at: https://www.researchgate.net/publication/359834524

[7] L. Cao and Y. Zhang, "Predictive Analytics for Inventory Management in Retail," *ResearchGate*, 2021. Available at: https://www.researchgate.net/publication/354321421

[8] S. Chopra and P. Meindl, *Supply Chain Management: Strategy, Planning, and Operation*, Pearson Education, 2018.

[9] M. Christopher, *Logistics and Supply Chain Management*, 5th ed., Pearson Education, 2016.

[10] A. Gunasekaran and E. W. T. Ngai, "Information technology applications in supply chain management," *International Journal of Production Economics*, vol. 87, no. 3, pp. 295–302, 2004.

[11] M. Grinberg, *Flask Web Development: Developing Web Applications with Python*, O'Reilly Media, 2018.

[12] S. Mani and A. Kannan, "Frontend Frameworks: Angular vs React – A Comparative Study," *International Journal of Computer Applications*, vol. 175, no. 7, 2020.

[13] P. DuBois, *MySQL*, Addison-Wesley Professional, 2014.

[14] S. Garfinkel, "Email-Based Identity and Authentication Systems," *Computer Security Journal*, 2003.

[15] C. Coronel and S. Morris, Database Systems: Design, Implementation, and Management, Cengage Learning, 2019.

[16] N. Dragoni et al., "Microservices: Yesterday, Today, and Tomorrow," Springer International Publishing, 2017.

[17] M. Adeniran, "Angular Framework: Best Practices and Component-Based Architecture," ResearchGate, 2021. Available at: https://www.researchgate.net/publication/355217321