

INVENTORY MANAGEMENT SYSTEM FOR FMCG INDUSTRY

MINOR PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE
AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

(Computer Science and Engineering)



Submitted By:

Dilraj Singh (2104092)

Ekuspreet Singh (2104097)

Harsimran Singh (2104117)

Submitted To:

Prof. Sita Rani

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GURU NANAK DEV ENGINEERING COLLEGE

LUDHIANA, 141006

April, 2024

ABSTRACT

The Fast-Moving Consumer Goods (FMCG) industry plays a vital role in global economies, constantly striving to meet the ongoing demand for everyday essentials. This sector relies on rapid production and distribution cycles, but these efficiencies can become weaknesses if inventory management is not optimized. Inefficiencies in this area can lead to stock shortages or overstocking, both of which disrupt production schedules, increase costs, and ultimately impact customer satisfaction. To address these concerns, this project proposes a comprehensive solution: a Digitized Inventory Management System (IMS) developed using a modern web framework. The IMS utilizes React to create a user-friendly and responsive interface accessible from any device. A RESTful API allows seamless interaction between this interface and the server-side coded in a Python Flask framework, ensuring efficient data processing and communication. For data persistence, the IMS integrates with a MongoDB database. This flexible, schema-less design of MongoDB allows for efficient storage of various data types critical for FMCG operations, including real-time inventory levels with automated alerts. These alerts prevent stock shortages and ensure a constant flow of raw materials. Employee data is securely stored within the system, facilitating efficient workforce planning and task allocation. Production cycles are tracked within the IMS, monitoring everything from raw material intake to finished goods completion. Deviations from the planned schedule trigger alerts for immediate corrective action, minimizing delays and ensuring adherence to quality standards. IMS also stores and manages product recipes within MongoDB. The adaptability of this system extends beyond its initial application in animal fodder production. The IMS can be readily configured to address the specific needs of other FMCG sub-sectors. For instance, the system can be tailored to manage perishable ingredients, packaging materials, and expiry dates in the food and beverage industry. Similarly, for household cleaning products, the IMS can be adapted to manage chemical ingredients, safety data sheets (SDS), and production cycles specific to those products. By implementing this multifaceted IMS, FMCG companies can achieve their primary objectives: efficient inventory management, optimized workforce management, mitigation of inventory shortages, improved production efficiency, and ultimately, the maintenance of consistent product quality across all production batches.

ACKNOWLEDGEMENT

We are highly grateful to the Dr. Sehijpal Singh, Principal, Guru Nanak Dev Engineering College (GNDEC), Ludhiana, for providing this opportunity to carry out the minor project work.

The constant guidance and encouragement received from Dr. Kiran Jyoti, H.O.D. CSE Department, GNDEC Ludhiana has been of great help in carrying out the project work and is acknowledged with reverential thanks.

We would like to express a deep sense of gratitude and thanks profusely to Project Guide, Dr. Sita Rani, without her wise counsel and able guidance, it would have been impossible to complete the project in this manner.

We express gratitude to other faculty members of computer science and engineering department of GNDEC for their intellectual support throughout the course of this work.

Finally, we are indebted to all whosoever have contributed in this report work.

Dilraj Singh

Ekuspreet Singh

Harsimran Singh

LIST OF FIGURES

Fig. No.	Figure Description	Page No.
1	A High-Level Abstraction of Application	30
2	Employee View Flowchart	30
3	Admin View Flowchart	31
4	Level 0 DFD	32
5	Level 1 DFD	32
6	Database Scheme Design	33
7	Directory Structure	34
8	Landing Page	63
9	SignUp Page	63
10	Job Dashboard Page	64
11	Inventory Items Page	64
12	Recipes Page	65
13	Job Creation Page	65
14	Employees Page	66
15	Log Completion Login	66
16	MongoDB (Jobs)	67
17	MongoDB (Products)	67
18	MongoDB (Items)	68
19	MongoDB (Employees)	68

TABLE OF CONTENTS

Contents	Page No.
<i>Abstract</i>	<i>i</i>
<i>Acknowledgement</i>	<i>ii</i>
<i>List of Figures</i>	<i>iii</i>
<i>List of Tables</i>	<i>iv</i>
<i>Table of Contents</i>	<i>v</i>
Chapter 1: Introduction	1
1.1. Introduction to Project	
1.2. Project Category (Research or Industry Based)	
1.3. Problem Formulation	
1.4. Identification/Recognition of Need	
1.5. Existing System	
1.6. Objectives	
1.7. Proposed System	
1.8. Unique Features of the Proposed System	
Chapter 2. Requirement Analysis and System Specification	15
2.1. Feasibility study	
2.2. Software Requirement Specification Document	
2.3 SDLC model	
Chapter 3. System Design	28
3.1 Design Approach (Function oriented or Object oriented)	
3.2 Detail Design	
3.2.1 Flow charts or Block Diagrams	
3.2.2 DFDs	
3.3 User Interface Design	
3.4 Methodology	

Chapter 4. Implementation and Testing	38
4.1. Introduction to Languages, IDE's, Tools and Technologies	
4.2. Algorithm/Pseudocode	
4.3. Testing Techniques: in context of project work	
4.4. Test Cases designed for the project work	
Chapter 5. Results and Discussions	59
5.1. User Interface Representation	
5.2. Brief Description of Various Modules of the system	
5.3. Snapshots of system with brief detail of each and discussion.	
5.4. Back Ends Representation	
5.4.1. Snapshots of Database Tables with brief description	
Chapter 6. Conclusion and Future Scope	69
6.1. Conclusion	
6.2. Key Accomplishments of IMS	
6.3. Future Scope	
References/Bibliography	73

Chapter 1. Introduction

1.1 Introduction to Project

The Digitized Inventory Management System (IMS) project for Natural Animal Diets is a strategic endeavor aimed at revolutionizing inventory management practices within the Fast-Moving Consumer Goods (FMCG) industry, particularly focusing on the production of animal fodder. Sponsored by Natural Animal Diets, this project addresses specific challenges faced by the company and the broader FMCG sector, offering a comprehensive solution tailored to their needs.

The IMS project centralizes raw material data, providing a consolidated view of inventory levels. One of its key functionalities is automated stock alerts, which notify teams when stock levels fall below predefined thresholds, preventing stockouts and production disruptions. Employee data, including qualifications and roles, is securely stored within the IMS, aiding in workforce planning, task allocation, and ensuring efficient resource utilization.

The IMS monitors production cycles from raw material intake to finished goods, scheduling production runs, allocating resources such as labor, and ensuring timely completion of production tasks. Deviations from planned schedules trigger alerts for immediate corrective action, minimizing delays and ensuring adherence to quality standards. Recipe management is another critical aspect addressed by the IMS, securely storing animal fodder recipes and calculating material requirements based on recipe proportions. This ensures consistency across.

Moreover, the adaptability of the IMS extends beyond animal fodder production. It can be seamlessly adapted to other FMCG contexts such as the Food and Beverage Industry, managing perishable ingredients, packaging materials, and production cycles. Similarly, in the Household Cleaning Products sector, the IMS can handle chemical ingredients, safety data sheets (SDS), and production schedules specific to these products.

By offering a centralized, automated, and adaptable inventory management solution, the IMS project aims to streamline operations, enhance efficiency, ensure product quality, and position Natural Animal Diets and other FMCG companies for sustainable growth and competitiveness in their respective markets.

1.2 Project Category

Project Category: Industry Based Project (Natural Animal Diets)

The project titled "Inventory Management System for FMCG Company" is an industry-based project due to its direct relevance and applicability to the Fast-Moving Consumer Goods (FMCG) sector. Key reasons include:

- 1.2.1 Sector-Specific Challenges:** The FMCG industry faces unique challenges such as rapid production cycles, inventory management complexities, and the need for real-time data processing.
- 1.2.2 Industry Requirements:** FMCG companies require efficient inventory management systems to ensure smooth production, minimize stockouts or overstocking, and maintain consistent product quality.
- 1.2.3 Technology Integration:** The use of modern web frameworks like React and backend technologies like Python Flask reflects an understanding of the technology landscape within the FMCG industry. These technologies enable seamless interaction, data processing, and communication critical for FMCG operations.
- 1.2.4 Data Management:** The integration with MongoDB as a NoSQL deals with diverse data types such as inventory levels, employee data, product recipes, and production schedules. The flexibility and scalability of MongoDB align well with the dynamic nature of FMCG data management.
- 1.2.5 Adaptability to Sub-Sectors:** The project's adaptability to different FMCG sub-sectors, such as food and beverage or household cleaning products, showcases its industry focus. Each sub-sector may have specific inventory management needs, and the IMS's ability to tailor solutions to these needs demonstrates its industry relevance.
- 1.2.6 Business Impact:** Efficient inventory management directly impacts business performance in terms of cost optimization, production efficiency, and customer satisfaction. By addressing these aspects, the project aims to deliver tangible business benefits to FMCG companies, highlighting its industry-based nature.

1.3 Problem Formulation

The problem formulation for the Digitized Inventory Management System (IMS) project within the Fast-Moving Consumer Goods (FMCG) industry revolves around addressing critical challenges that impact operational efficiency, cost-effectiveness, and customer satisfaction.

- 1.3.1 Inventory Inefficiencies:** One of the primary problems is inefficient inventory management practices within FMCG companies. This includes issues such as stockouts, overstocking, inaccurate inventory data, manual tracking processes, and lack of real-time visibility into inventory levels. These inefficiencies lead to production disruptions, increased costs due to excess inventory or rush orders, and ultimately, impact customer satisfaction due to delayed deliveries or stock shortages.
- 1.3.2 Production Disruptions:** FMCG companies often face production disruptions due to unforeseen factors such as raw material shortages, equipment breakdowns, or deviations from production schedules. These disruptions not only delay product delivery but also result in additional costs and potential quality issues if corrective measures are not taken promptly.
- 1.3.3 Data Fragmentation:** Another challenge is the fragmentation of data across different systems or departments within FMCG companies. This fragmentation hinders data accessibility, integration, and analysis, making it difficult to derive meaningful insights for informed decision-making.
- 1.3.4 Manual Processes:** Many FMCG companies still rely on manual processes for inventory management, production tracking, and workforce allocation. This manual approach is prone to errors, time-consuming, and lacks scalability, especially as companies scale up their operations or introduce new product lines.
- 1.3.5 Lack of Scalability:** Traditional inventory management systems may lack scalability to accommodate the evolving needs of FMCG companies, such as handling increased data volumes, adapting to new product variations, or integrating with emerging technologies for enhanced efficiency.

1.4 Identification/Recognition of Need

The need for a Digitized Inventory Management System (IMS) in the Fast-Moving Consumer Goods (FMCG) industry stem from several factors:

- 1.4.1 Operational Inefficiencies:** FMCG companies often face operational inefficiencies due to manual inventory tracking, disjointed data systems, and lack of real-time visibility into inventory levels. These inefficiencies lead to stockouts, overstocking, production delays, and increased costs.
- 1.4.2 Demand Fluctuations:** The FMCG sector experiences rapid demand fluctuations, seasonal variations, and evolving consumer preferences. Traditional inventory management methods struggle to adapt to these dynamic changes, resulting in inventory imbalances and missed sales opportunities.
- 1.4.3 Competitive Pressures:** In a competitive market, FMCG companies must optimize their operations to stay ahead. Efficient inventory management directly impacts production efficiency, cost control, and customer service levels, all of which are critical for maintaining a competitive edge.
- 1.4.4 Data-Driven Decision Making:** The need for data-driven decision-making is paramount in the FMCG industry. Accurate and timely data insights enable companies to forecast demand, optimize inventory levels, identify trends, and respond quickly to market changes, ultimately driving business growth and profitability.
- 1.4.5 Technology Advancements:** The evolution of technology, including web frameworks, cloud computing, and data analytics, presents a significant opportunity to revolutionize inventory management by leveraging these tools to modernize processes, improve efficiency, and address longstanding challenges within the industry.
- 1.4.6 Customer Expectations:** In today's competitive FMCG landscape, meeting customer expectations for timely and accurate delivery is not just a business necessity but also a crucial factor in maintaining brand loyalty, positive reputation, and market leadership.

1.5 Existing System

The existing inventory management systems in many Fast-Moving Consumer Goods (FMCG) companies often suffer from several flaws and limitations, highlighting the need for a more advanced solution like the proposed Digitized Inventory Management System (IMS). Some of the flaws include:

- 1.5.1 Manual Processes:** Many FMCG companies still rely on manual inventory tracking methods, such as spreadsheets or pen-and-paper systems. This manual approach is prone to human errors, time-consuming, and lacks real-time updates, leading to inaccuracies in inventory data.
- 1.5.2 Disjointed Systems:** In larger FMCG companies, different departments or locations may use separate inventory management systems that are not integrated. This fragmentation results in data silos, duplication of efforts, and difficulties in obtaining a centralized view of inventory levels and movements.
- 1.5.3 Limited Visibility:** Traditional inventory systems often provide limited visibility into real-time inventory levels, especially across multiple warehouses or distribution centers. This lack of visibility hinders proactive decision-making and may lead to stockouts or overstocking.
- 1.5.4 Inefficient Data Handling:** Handling diverse data types, such as inventory levels, product recipes, production schedules, and employee data, can be challenging with conventional systems. These systems may lack the flexibility to manage and analyze large volumes of data efficiently.
- 1.5.5 Scalability Issues:** As FMCG companies grow or introduce new product lines, scalability becomes a concern with existing inventory systems. These systems may struggle to handle increased data volumes, complex inventory hierarchies, or diverse product variations.
- 1.5.6 Limited Analytics:** Conventional inventory systems often lack advanced analytics capabilities for demand forecasting, trend analysis, and predictive insights. This limits

FMCG companies' ability to optimize inventory levels, anticipate market changes, and make data-driven decisions.

1.5.7 Dependency on Legacy Technology: Some FMCG companies may still use legacy inventory systems built on outdated technologies. These systems may lack support for modern integrations, APIs, or cloud-based solutions, hindering interoperability and agility.

Overall, the flaws in existing inventory management systems within the FMCG industry highlight the urgent need for a Digitized IMS that addresses these shortcomings. The proposed IMS aims to automate processes, integrate data, provide real-time visibility, enable advanced analytics, and offer scalability to meet the evolving needs of FMCG companies in today's competitive market landscape.

1.6 Objectives

1.6.1 To ensure convenient management of inventory.

1.6.2 To manage and store employee data.

1.6.3 To mitigate inventory shortages through an alert system.

1.7 Proposed System

The proposed Digitized Inventory Management System (IMS) for Natural Animal Diets represents a comprehensive solution tailored to the specific needs of the company within the FMCG industry, particularly focusing on the production of animal fodder. At its core, the IMS centralizes raw material data, offering a unified view of inventory levels, supplier information, procurement history, and pricing details. This centralized approach streamlines inventory tracking, facilitates informed decision-making regarding procurement strategies, and enhances supplier management efficiency.

One of the standout features of the IMS is its automated stock alerts capability. The system continuously monitors inventory levels in real-time and triggers alerts when stock falls below predefined thresholds. These automated alerts play a crucial role in preventing stockouts,

minimizing production disruptions, and optimizing inventory levels based on dynamic demand patterns. This proactive approach ensures timely replenishment of raw materials, smooth production operations, and improved customer satisfaction through consistent product availability.

Employee data management is another key aspect addressed by the IMS. It securely stores employee qualifications, roles, certifications, and training records, facilitating workforce planning, task allocation, and performance evaluation. Advanced access controls and encryption protocols ensure data security and compliance with privacy regulations, safeguarding sensitive employee information.

The IMS's capabilities extend beyond inventory management to encompass production cycle monitoring. From raw material intake to finished goods delivery, the IMS schedules production runs, allocates resources such as labor and equipment, and tracks production progress in real-time. Any deviations from the planned production schedule trigger alerts for immediate corrective action, ensuring on-time delivery and adherence to quality standards.

1.8. Unique Features of the Proposed System

The proposed IMS for Natural Animal Diets in the FMCG industry encompasses several key features and functionalities designed to streamline operations, enhance efficiency, and ensure product quality. Here are the details of the proposed system:

1.8.1 Centralized Raw Material Data: The IMS centralizes raw material data, providing a consolidated view of inventory levels and procurement history. This centralized approach ensures accurate inventory tracking, efficient supplier management, and informed decision-making regarding procurement strategies.

1.8.2 Automated Stock Alerts: One of the critical functionalities of the IMS is automated stock alerts. The system continuously monitors inventory levels in real-time and triggers alerts when stock falls below predefined thresholds. These automated alerts prevent stockouts, minimize production disruptions, and optimize inventory levels to meet demand fluctuations effectively.

- 1.8.3 Secure Storage of Employee Data:** Employee data, including qualifications, roles, certifications, and training records, is securely stored within the IMS. This information is integral to workforce planning, task allocation, performance evaluation, and compliance management. Access controls and encryption protocols ensure data security and confidentiality.
- 1.8.4 Production Cycle Monitoring:** The IMS monitors the entire production cycle from raw material intake to finished goods. It schedules production runs based on demand forecasts, allocates resources such as labor and equipment efficiently, and tracks production progress in real-time. Any deviations from the planned schedule trigger alerts for immediate corrective action, ensuring on-time delivery and adherence to quality standards.
- 1.8.5 Recipe Management and Material Requirements Calculation:** Animal fodder recipes are critical to maintaining product consistency and quality. The IMS securely stores these recipes, including ingredient lists, proportions, and production instructions. It also calculates material requirements based on recipe specifications, minimizing wastage and optimizing inventory usage.
- 1.8.6 Adaptability and Scalability:** The IMS is designed to be adaptable and scalable, catering to evolving business needs and industry trends. It can easily integrate with existing systems, accommodate new product lines, and expand functionalities as required. This adaptability allows the system to be customized for other FMCG contexts beyond animal fodder production, such as food and beverage or household cleaning products.
- 1.8.7 User-Friendly Interface:** The IMS features a user-friendly interface accessible from any device, facilitating ease of use, navigation, and information retrieval for stakeholders across departments. Intuitive dashboards, interactive reports, and customizable settings enhance user experience and productivity.

Overall, the proposed Digitized IMS for Natural Animal Diets is a comprehensive solution that addresses the specific challenges faced by the company in the FMCG industry.

Chapter 2. Requirement Analysis and System Specification

2.1 Feasibility study

2.1.1 SWOT Analysis:

Strengths: The Inventory Management System (IMS) offers comprehensive automation and efficiency enhancements by streamlining inventory processes, scheduling production, and issuing stock alerts, thereby reducing manual errors and improving operational performance. It leverages real-time data insights, analytics, and reporting capabilities, empowering data-driven decision-making and strategic planning for businesses. With secure data management features, the IMS safeguards sensitive information such as raw material data, employee details, and production records, ensuring compliance and data security. Its scalability and adaptability allow seamless integration with existing systems and the ability to accommodate changes in product lines or fast-moving consumer goods (FMCG) contexts. Moreover, the IMS contributes to enhanced product quality by managing recipes, calculating material requirements, and monitoring production cycles, ensuring consistent quality and enhancing customer satisfaction.

Weaknesses: Implementing the Inventory Management System (IMS) for Natural Animal Diets presents financial, technical, and organizational challenges. The initial implementation incurs costs for technology, infrastructure, training, and integration, posing a financial hurdle. Integrating the IMS with existing systems may be complex and time-consuming, requiring careful planning to ensure seamless integration. Additionally, adopting a new digital system necessitates changes in organizational culture, workflows, and employee training, which can lead to resistance or adaptation issues. Overcoming these challenges requires strategic planning, effective communication, and stakeholder engagement to minimize disruptions and maximize the benefits of the IMS implementation. It's essential to manage costs effectively, address technical complexities during integration, and navigate change management processes to facilitate a smooth transition and realize the IMS's potential in enhancing operational efficiency and data management for Natural Animal Diets.

Opportunities: The implementation of the Inventory Management System (IMS) for Natural Animal Diets offers significant strategic advantages. Firstly, it facilitates market expansion into other Fast-Moving Consumer Goods (FMCG) sectors beyond animal fodder production, thereby enhancing revenue opportunities and diversifying the business portfolio. This expansion leverages the IMS's adaptability to different market segments and consumer needs.

Secondly, the IMS provides a competitive edge by optimizing operational processes, improving product quality, and enhancing customer service. Through streamlined inventory management, production scheduling, and stock alerts, operational efficiency is maximized, reducing costs and minimizing errors. This, in turn, leads to higher customer satisfaction and loyalty, strengthening the company's market position.

Furthermore, the IMS's integration with emerging technologies like Artificial Intelligence (AI), Internet of Things (IoT), and predictive analytics significantly boosts its capabilities. These technologies enable real-time data insights, advanced analytics, and proactive decision-making, leading to more informed resource management and strategic planning.

Threats: Implementing the Inventory Management System (IMS) for Natural Animal Diets involves navigating several key challenges. Firstly, technological risks such as rapid advancements or market trend changes can impact system compatibility, risk obsolescence, and raise cybersecurity concerns, necessitating proactive monitoring and adaptation strategies. Secondly, the competitive landscape may intensify with rivals adopting similar digitized inventory management systems, requiring continuous innovation and differentiation to maintain a competitive edge. Additionally, ensuring regulatory compliance with industry standards, data privacy laws, and quality requirements presents ongoing challenges that demand meticulous monitoring and adherence. Lastly, internal resistance, whether from employees, stakeholders, or adoption barriers, can potentially affect the project's success and timeline, highlighting the importance of effective change management strategies and comprehensive training programs to facilitate smooth implementation and acceptance.

2.1.2 Cost-Benefit Analysis:

Costs: The implementation of the Inventory Management System (IMS) for Natural Animal Diets encompasses both initial investment and ongoing operational costs. The initial investment covers expenses such as purchasing and installing servers, computers, networking equipment, and IMS software licenses. It also includes costs related to system integration, customization, training, implementation support, and migrating existing data from legacy systems to the IMS platform. On the other hand, ongoing operational costs consist of annual expenses for system maintenance, updates, troubleshooting, and technical support. Additionally, there are ongoing training costs to ensure employees efficiently utilize the IMS. Furthermore, monthly or annual fees for cloud hosting, storage, and data backup services contribute to the ongoing operational expenses. Effectively managing these costs is crucial for the successful implementation and sustained operation of the IMS, requiring careful budgeting, monitoring, and optimization strategies to ensure cost-effectiveness and maximize the return on investment over time.

Benefits: The Inventory Management System (IMS) for Natural Animal Diets yields significant benefits in operational efficiency, cost reduction, and revenue enhancement. It streamlines inventory management tasks, automates alerts, and provides real-time data access, improving productivity and preventing stockouts while optimizing inventory levels and reducing holding costs. Better material planning and recipe management minimize wastage and material costs, while proactive inventory management minimizes stockouts and associated expenses. Compliance management also helps avoid penalties and fines, leading to cost savings. Additionally, the IMS improves product availability, timely deliveries, and customer satisfaction, thereby increasing sales and revenue. Enhanced product quality, operational efficiency, and customer service differentiate Natural Animal Diets from competitors and attract more customers, contributing to revenue growth and market success.

2.1.3 Technical Feasibility:

The successful implementation of the Digitized Inventory Management System (IMS) for Natural Animal Diets relies heavily on robust infrastructure and a well-structured technology stack. In terms of infrastructure, the project necessitates high-performance servers capable of handling database operations and web hosting tasks efficiently. Workstations are required for both development purposes and user access to the IMS. Secure connectivity is ensured through networking equipment, and adequate storage devices are essential for storing the vast amount of data generated and managed by the system.

On the software front, a stable operating system forms the foundation for the IMS, providing a reliable environment for its functioning. A suitable database management system (DBMS) such as MongoDB is chosen for its flexibility, scalability, and ability to handle unstructured data effectively. Modern web and backend frameworks play a crucial role in the IMS's functionality, with React.js powering the frontend development for interactive user interfaces and responsive designs. For backend development, Python with frameworks like Flask or Django is preferred due to their simplicity, scalability, and support for API development.

The technology stack extends beyond these core components to include various tools and libraries that enhance system functionality and performance. Version control is managed using Git/GitHub, ensuring efficient collaboration and code management. Containerization is achieved through Docker, facilitating easy deployment and scalability of the IMS. Web hosting is handled by servers running NGINX or Apache, providing robust and secure hosting capabilities. Asynchronous processing is managed using Celery/Redis, optimizing system performance and responsiveness.

Overall, this comprehensive infrastructure and technology stack ensure the technical feasibility of the IMS project. They enable efficient development, seamless integration, reliable operation, and enhanced functionality and performance, meeting the complex requirements of managing inventory for Natural Animal Diets effectively.

2.1.4 Operational Feasibility:

User acceptance is a critical aspect of operational feasibility for the Digitized Inventory Management System (IMS) at Natural Animal Diets. It involves assessing the willingness and ability of end users, including employees and stakeholders, to embrace and effectively utilize the IMS in their daily operations. Factors influencing user acceptance include the system's ease of use, intuitive interface design, functionality alignment with user needs, and perceived benefits.

To ensure user acceptance, Natural Animal Diets can conduct user acceptance testing (UAT) to gather feedback, identify usability issues, and make necessary improvements before full-scale deployment. Providing comprehensive training and user support resources, such as user manuals, tutorials, and helpdesk assistance, is crucial in building user confidence and competence with the IMS.

Training plays a pivotal role in the operational feasibility of the Digitized Inventory Management System (IMS) for Natural Animal Diets. It involves equipping employees, managers, and system administrators with the knowledge, skills, and competencies required to effectively use, manage, and maintain the IMS.

A structured training program should cover various aspects of the IMS, including system functionalities, data entry, inventory management processes, reporting tools, and security protocols. Training sessions can be conducted through workshops, online courses, interactive simulations, and hands-on exercises tailored to different user roles and proficiency levels.

Providing ongoing training opportunities, refresher courses, and continuous learning support ensures that users stay updated with system updates, new features, and best practices. Incorporating feedback from training sessions and monitoring user performance metrics help identify training gaps, refine training materials, and enhance overall user proficiency with the IMS.

Therefore, effective training contributes to smoother system implementation, improved productivity, and successful adoption of the IMS across the organization.

2.2 Software Requirement Specification Document

2.2.1 Functional Requirements: In order to achieve the goals of the IMS (Inventory Management System) project certain functional requirements have been identified:

User Authentication: The user authentication module is a fundamental component of the IMS, designed to implement a robust authentication mechanism, such as password-based or multi-factor authentication and ensure a secure login. This functionality aims to restrict access to the system, allowing only authorized personnel to use different features based on their roles within Natural Animal Diets (NAD). In the context, Admin role can access multiple information sources meanwhile an employee would not get access to the business details. This is implemented while authentication.

Role Based Access Control (RBAC): Establish a role-based access control system where each user is assigned specific roles determining the features they can access. The Administrator, or Admin, holds the highest level of access within the system. They have the authority to manage user accounts, configure inventories and raw materials, and have unrestricted access to all features. The Employee role is designed for general users involved in the production process. They can start the jobs listed by admins and complete them. Employees accounts can only be created by Administrator.

Password Encryption: Employ strong hashing algorithms (currently SHA-256) to securely store and transmit passwords, protecting user credentials from unauthorized access. HTTP responses are send using flask session token.

Dashboard: A user-friendly dashboard displaying key metrics, such as raw material inventory levels, production status, and alerts for inventory shortages to allow users to have a quick overview of essential information, facilitating informed decision-making.

Raw Material Management: The Inventory Management System (IMS) for Natural Animal Diets offers user-friendly features for managing raw materials efficiently. Users, including admins and designated staff, can easily add new raw materials with details such as name, quantity, and supplier information, and edit existing details with real-time updates. The system provides instant updates on raw material levels, keeping up to 5 users informed. Additionally, a secure process for authorized deletion ensures safety, preventing accidental removal of

important data. These functionalities enhance the system's usability, data accuracy, and security, contributing to streamlined inventory management for Natural Animal Diets.

Employee Management: The Inventory Management System (IMS) for Natural Animal Diets includes features for managing employees efficiently. It allows users to add new employees and grant them access to the application, with admins having the capability to view and modify employee data. The system stores comprehensive employee details and enables the assignment of roles, ensuring employees can access relevant job information based on their roles. These functionalities streamline employee management, access control, and role-based information sharing within the organization, enhancing operational effectiveness and data security.

Production Cycle Monitoring: The Inventory Management System (IMS) for Natural Animal Diets features a user-friendly interface that simplifies visualizing production cycle stages. It provides instant updates on production progress, allowing users to easily update the system with real-time data. This functionality is accessible to both employees and managers, ensuring accurate and timely information updates. The system's integration with the raw material management module enables users to correlate production progress with raw material availability, minimizing disruptions and promoting efficient production management.

Storage of Recipes: The Inventory Management System (IMS) for Natural Animal Diets features a centralized and secure repository for storing production recipes. This ensures that authorized users, such as administrators, can manage and access these critical documents securely. Only authorized users have permission to upload, edit, and access production recipes, maintaining data integrity and security within the system.

Alert System: The Inventory Management System (IMS) for Natural Animal Diets includes an alert system with predefined triggers that generate alerts when raw material levels drop below set thresholds, ensuring timely notifications to stakeholders. These triggers are customizable to align with the company's production dynamics, and alerts can be delivered via email notifications, enhancing communication and proactive management of inventory levels.

2.2.2 Non-Functional Requirements:

The Inventory Management System (IMS) for Natural Animal Diets must meet key performance, scalability, reliability, and maintainability requirements. It should deliver responsive and efficient performance with quick response times and minimal data processing latency, aiming for response times within 5 seconds even under peak loads. Scalability is crucial, ensuring the system can handle increased data volume, user base, and concurrent transactions without significant performance degradation. Reliability is paramount, with the system designed to minimize system failures, crashes, or unexpected downtimes. Additionally, maintainability is essential, requiring a modular and well-documented codebase and architecture to ease maintenance, updates, and bug fixes over time, ensuring the system's long-term viability and effectiveness.

2.2.3 Definition of Modules and Functionality:

React: React is a JavaScript library for building user interfaces, allowing developers to create dynamic and interactive web applications with a modular and efficient component-based architecture. React uses a virtual DOM to optimize rendering performance. Changes to the virtual DOM are compared to the actual DOM, and only the necessary updates are applied, reducing the number of direct manipulations to the real DOM. JSX is a syntax extension for JavaScript that allows embedding HTML-like code within JavaScript.

React Router v6.0: React Router is a standard library for routing in React applications, allowing developers to create dynamic and navigable user interfaces. React Router uses components such as `BrowserRouter` to wrap the entire application, `Routes` to store all the routes, `Route` to create a route, `Link` to connect a route to a clickable button and `useNavigate` to create dynamic redirection.

Vite: Vite is a build tool for modern web development that aims to provide a fast and efficient development environment. It is designed to optimize the development experience by using native ES modules (ESM) and other modern JavaScript features. Vite supports a plugin system, enabling developers to extend and customize the build process. Vite includes built-in support for creating Progressive Web Apps, simplifying the process of adding service workers and manifest files.

Tailwind: Tailwind CSS is a utility-first CSS framework that streamlines the process of styling web applications. Unlike traditional frameworks with predesigned components, Tailwind provides low-

level utility classes for building responsive designs directly in your markup. It emphasizes flexibility, rapid development, and easy customization.

Daisy UI: It is one of the most popular, free and open-source component libraries for Tailwind CSS. It is used to create standardized designs.

Axios: Axios is a popular JavaScript library used for making HTTP requests from web browsers and Node.js environments. It provides a simple and consistent API, making it easier to work with asynchronous data and interact with web servers. Axios is built on the Promise-based architecture, allowing developers to utilize the `async/await` syntax for handling asynchronous operations. Axios automatically parses JSON response data, simplifying the process of working with JSON APIs. It also allows sending JSON data in the request body without manual serialization.

Flask: Flask is a lightweight and versatile web framework for Python, designed to make web development straightforward and flexible. Known for its simplicity and ease of use, its modular design allows developers to choose the components they need, making it scalable for both small projects and larger, more complex applications. Flask follows the WSGI (Web Server Gateway Interface) standard, making it compatible with various web servers. Flask's is one of the most popular among developers seeking an efficient and customizable frameworks for web development in Python.

Flask-CORS: Flask-CORS is a Flask extension that simplifies Cross-Origin Resource Sharing (CORS) in Flask applications. CORS is a security feature implemented by web browsers to restrict web pages from making requests to a different domain than the one that served the original web page. When developing web applications, especially those with separate frontend and backend components hosted on different domains, CORS issues can arise. The configuration of Flask-CORS is flexible, allowing developers to define the specific origins, methods, headers, and other CORS-related settings tailored to their application's requirements.

Python Dotenv: The process of using environment variables from a `.env` file in a Python project typically involves using a library called `python-dotenv`. This library loads the environment variables from a `.env` file into the application's environment. The `.env` file is often used to store configuration settings for an application, such as database connection strings, API keys, and other environment-specific parameters.

Passlib: Passlib is a Python library for securely hashing passwords and managing password-related tasks. It provides utilities for generating secure password hashes, verifying passwords, and managing password policies, making it easy to implement secure password authentication in applications. Passlib supports multiple hashing algorithms, including bcrypt, SHA-256, and SHA-512, and provides features such as password hashing, salting, and stretching to enhance security. With Passlib, you can build web applications that protect user passwords and sensitive information from unauthorized access and data breaches.

Mongo DB: It is an open-source, document-oriented database designed for flexibility, scalability, and ease of development. Unlike traditional relational databases, MongoDB does not rely on the tabular structure of rows and columns. Instead, it stores data in BSON (Binary JSON)-like documents, allowing for a more natural representation of complex, hierarchical structures. MongoDB is horizontally scalable, allowing seamless distribution of data across multiple servers. It supports sharding, enabling high-volume and high throughput applications to scale easily. MongoDB ensures high availability through features like replica sets. Data is replicated across multiple servers, allowing automatic failover in case of hardware failure or other issues and provides document validation to enforce data integrity. Developers can define rules to validate documents, ensuring that data adheres to specific constraints.

PyMongo: PyMongo is the official Python driver for MongoDB. It provides tools and utilities for interacting with MongoDB databases, including querying data, inserting and updating documents, and performing 11 administrative tasks, making it easy to work with MongoDB in Python applications. PyMongo follows the MongoDB API and provides a high-level API for common database operations, as well as a low-level API for advanced database manipulation. With PyMongo, robust and scalable web applications can be built using MongoDB.

2.3 SDLC model

The SDLC (Software Development Life Cycle) model for developing the proposed Bus Pass System involves several stages to ensure a systematic and efficient development process. Below is an outline of the SDLC model tailored for this project:

2.3.1 Planning Phase: During the planning phase of the IMS development project for Natural Animal Diets, the primary objectives are to define project goals, scope, budget, timelines, and required resources. This involves conducting initial meetings with stakeholders to gather requirements, understand business needs, and establish project objectives. A project charter will be created to outline the project scope, objectives, stakeholders, roles, responsibilities, and constraints. Additionally, a detailed project plan will be developed, including tasks, milestones, timelines, resource allocations, budget estimates, and risk management strategies. Feasibility studies will be conducted to assess technical, operational, and financial feasibility, ensuring the project's viability. Finally, project governance, communication protocols, and collaboration tools will be defined to facilitate effective project management and ensure alignment with stakeholder expectations. These activities are crucial for laying a solid foundation for the IMS development project and ensuring its success in meeting business objectives.

2.3.2 Analysis Phase: During the Analysis Phase of the IMS development for Natural Animal Diets, the key objectives are to gather detailed requirements, analyze business processes, and identify system functionalities. This involves conducting stakeholder surveys to gather both functional and non-functional requirements, analyzing existing workflows, data flows, and inventory management processes within the organization. Use cases, user stories, and System Requirements Specifications (SRS) documents will be created to detail functional requirements, user roles, data entities, and system behavior. Requirements will then be prioritized based on business value, criticality, and feasibility for implementation. Validation of requirements will be done with stakeholders to clarify any ambiguities and obtain sign-off before proceeding to the design phase. This phase is critical for ensuring a comprehensive understanding of the project's requirements and laying the groundwork for the subsequent design and development stages of the IMS.

2.3.3 Design Phase: In the Design Phase of the IMS development for Natural Animal Diets, the primary objectives are to design the architecture, user interface, database schema, and system components. This involves developing a system architecture design to outline the overall structure, modules, layers, and integration points of the IMS. Wireframes, prototypes, and UI mockups are created to visualize the user interface, navigation flows, and user interactions. Database schemas, entity-relationship diagrams (ERDs), and data models are designed to store inventory data, user information, and system configurations effectively. System interfaces, APIs, data integration points, and external system interactions are defined to ensure seamless integration and functionality. Technical specifications, design documents, and component diagrams are developed to detail system components, algorithms, and implementation approaches, providing a roadmap for the development phase. This phase is crucial for translating requirements into a well-designed system architecture and user interface that aligns with stakeholder expectations and project objectives.

2.3.4 Implementation Phase: During the Implementation Phase of the IMS development for Natural Animal Diets, the primary objectives are to develop, code, and integrate IMS components based on design specifications and requirements. This involves developing frontend components using React.js for the user interface, navigation, and interactive features, ensuring a responsive and engaging user experience. Backend logic and functionalities are implemented using Python with Flask framework, encompassing data processing, API development, and business logic implementation to support system functionality. Third-party libraries, APIs, and services are integrated for additional functionalities like authentication, notifications, and reporting, enhancing the system's capabilities. Code reviews, unit testing, and version control using Git/GitHub are conducted to ensure code quality, stability, and maintainability throughout the development process. Features are iteratively developed, feedback is addressed, and progress is tracked against the project plan and milestones to ensure alignment with project objectives and timelines. This phase is critical for translating design specifications into functional components, ensuring the IMS meets stakeholder requirements and delivers value to Natural Animal Diets.

2.3.5 Testing Phase: The Testing Phase of the IMS development for Natural Animal Diets aims to validate the functionality, performance, security, and usability of the system through comprehensive testing activities. This includes developing test cases, scenarios, and data sets covering functional, integration, system, and acceptance testing. Unit testing is conducted to validate individual components, modules, and functions for correctness and reliability, ensuring they perform as expected. Integration testing is crucial to ensure seamless communication and interoperability between frontend and backend components, verifying that they work together as intended. System testing is then performed to validate end-to-end functionality, data integrity, error handling, and system behavior under various conditions, including peak loads and stress scenarios. User acceptance testing (UAT) plays a vital role in validating usability, user workflows, and adherence to requirements from stakeholders' perspectives. This phase involves executing test cases that mimic real-world user interactions to ensure the system meets user expectations and effectively supports business processes. Overall, the Testing Phase is essential for identifying and resolving issues, ensuring the IMS meets quality standards, security protocols, and performance benchmarks before deployment, enhancing user satisfaction and system reliability.

By following a structured Software Development Life Cycle (SDLC) model involving the planning, analysis, design, implementation, testing, deployment, and maintenance phases, Natural Animal Diets can ensure a systematic, efficient, and successful development and deployment of the Digitized Inventory Management System (IMS). This approach aligns with industry best practices and standards, providing a framework for managing project complexities, mitigating risks, and delivering a high-quality solution that meets business objectives and user expectations.

Chapter 3. System Design

3.1 Design Approach (Function oriented or Object oriented)

The design approach of the Digitized Inventory Management System (IMS) for Natural Animal Diets leans towards an object-oriented design (OOD) approach rather than a function-oriented design (FOD) approach.

3.1.1 Object-Oriented Design (OOD) Principles:

Encapsulation: OOD promotes encapsulation, where data (attributes) and methods (functions) are encapsulated within objects. This encapsulation helps in hiding the internal implementation details, reducing complexity, and improving modularity.

Inheritance: OOD allows for inheritance, enabling the creation of hierarchical relationships between classes. Inheritance promotes code reusability, as subclasses can inherit attributes and methods from parent classes, reducing redundancy and promoting a DRY (Don't Repeat Yourself) codebase.

Polymorphism: OOD supports polymorphism, allowing objects to exhibit different behaviors based on their types or classes. This flexibility in behaviour enables dynamic method dispatching, interface implementations, and code flexibility.

3.1.2 Advantages of Object-Oriented Design:

Modularity: OOD promotes modularity by breaking down complex systems into smaller, manageable objects. Each object encapsulates its functionality, promoting code organization, maintenance, and scalability.

Code Reusability: OOD facilitates code reusability through inheritance and polymorphism. Common functionalities and behaviors can be defined in base classes and reused across different parts of the system, reducing development time and effort.

Ease of Maintenance: OOD simplifies maintenance by isolating changes to specific objects or classes. Modifications or enhancements can be made to individual objects without affecting the entire system, promoting code stability and reliability.

3.1.3 Application to the IMS Project:

In the IMS project, classes and objects are used to model real-world entities such as inventory items, employees, production cycles, and customer orders. Each object encapsulates its data and behaviors, facilitating modular development and system understanding. For example, classes like InventoryItem, Employee, Order, and ProductionCycle are defined with attributes (data) and methods (functions) relevant to their respective functionalities within the IMS. Inheritance is utilized where common attributes and behaviors exist among related objects. For instance, a base class like Product may have subclasses such as Raw Material and Finished Goods, inheriting common properties and methods while allowing customization for specific types. Polymorphism is leveraged for behaviours that can vary based on object types. For instance, different types of inventory items may have different calculation methods for reorder levels or pricing, implemented through polymorphic methods.

Overall, the object-oriented design approach in the IMS project promotes code organization, reusability, maintainability, and scalability, aligning with best practices for complex software systems like inventory management solutions.

3.2 Detail Design

3.2.1 Flow charts or Block Diagrams

Abstract System Design: The image shows a web application architecture with Employees and Admins accessing a Server via computers. The Server processes login and registration, communicates with a React JS front-end for job tasks, and interfaces with a cloud-based Database.

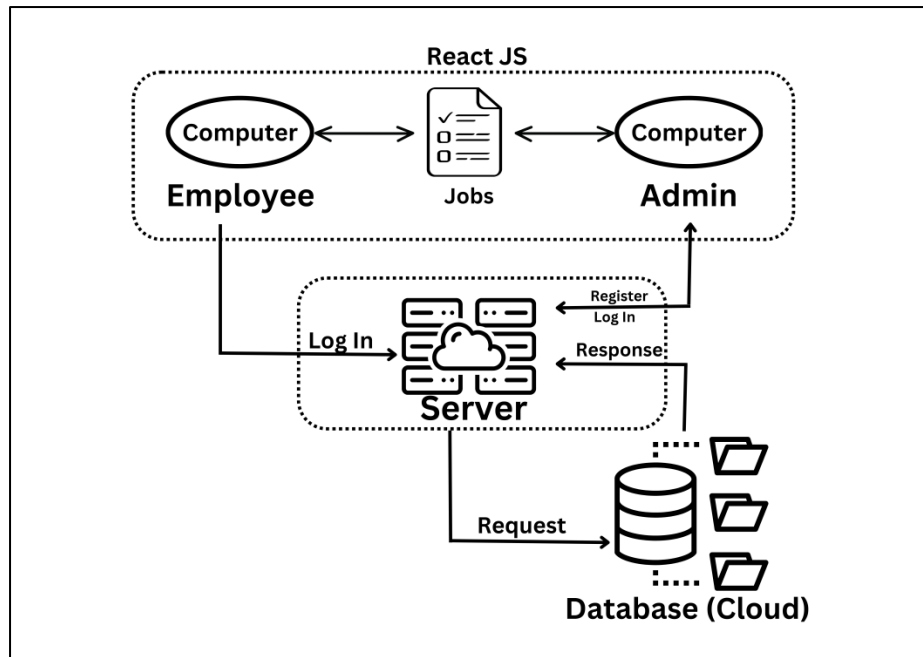


Figure 1 : A High Level Abstraction of Application

Employee Flowchart: This flowchart represents a job processing system starting with authentication, proceeding to job viewing, and then a decision node for job requests that can lead to task completion or an alert if unsuccessful or rejected.

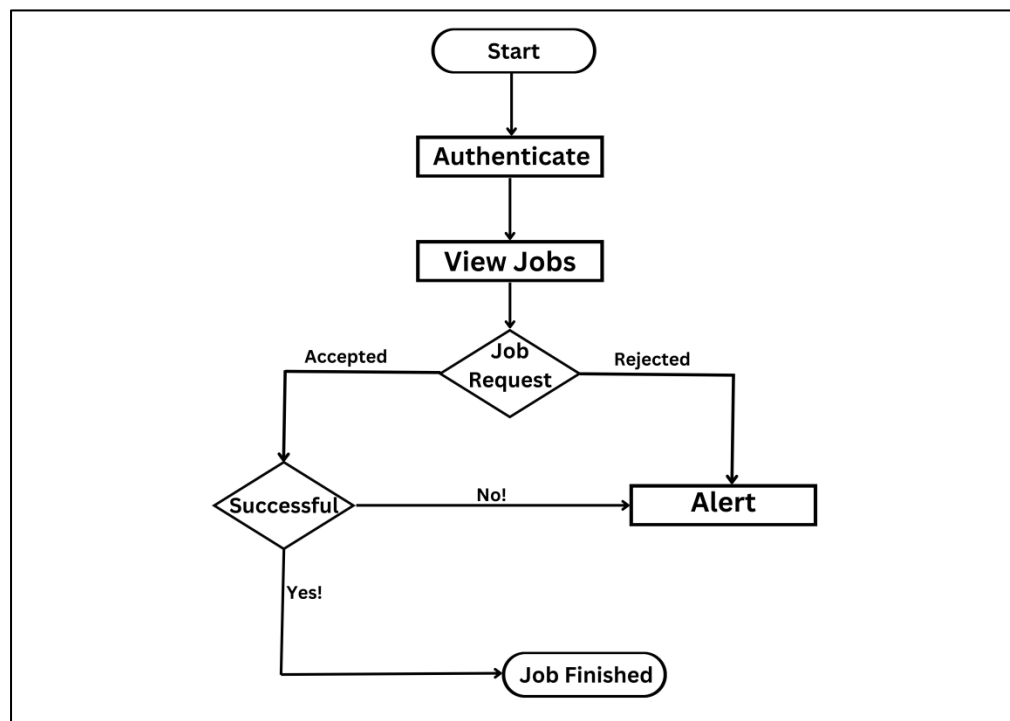


Figure 2: Employee View Flowchart

Administrator Flowchart: The image presents a flowchart starting with "Start," leading to "Authenticate," and branching off to "Register" if accepted. Post-authentication, one can "View Jobs," "Manage Recipes" with options to add, delete, update, or "Manage Employees," and "Manage Raw Materials." The "Manage Employees" allows for hiring new employees or firing, and "Manage Raw Materials" includes monitoring current supplies. The process also links back to setting jobs.

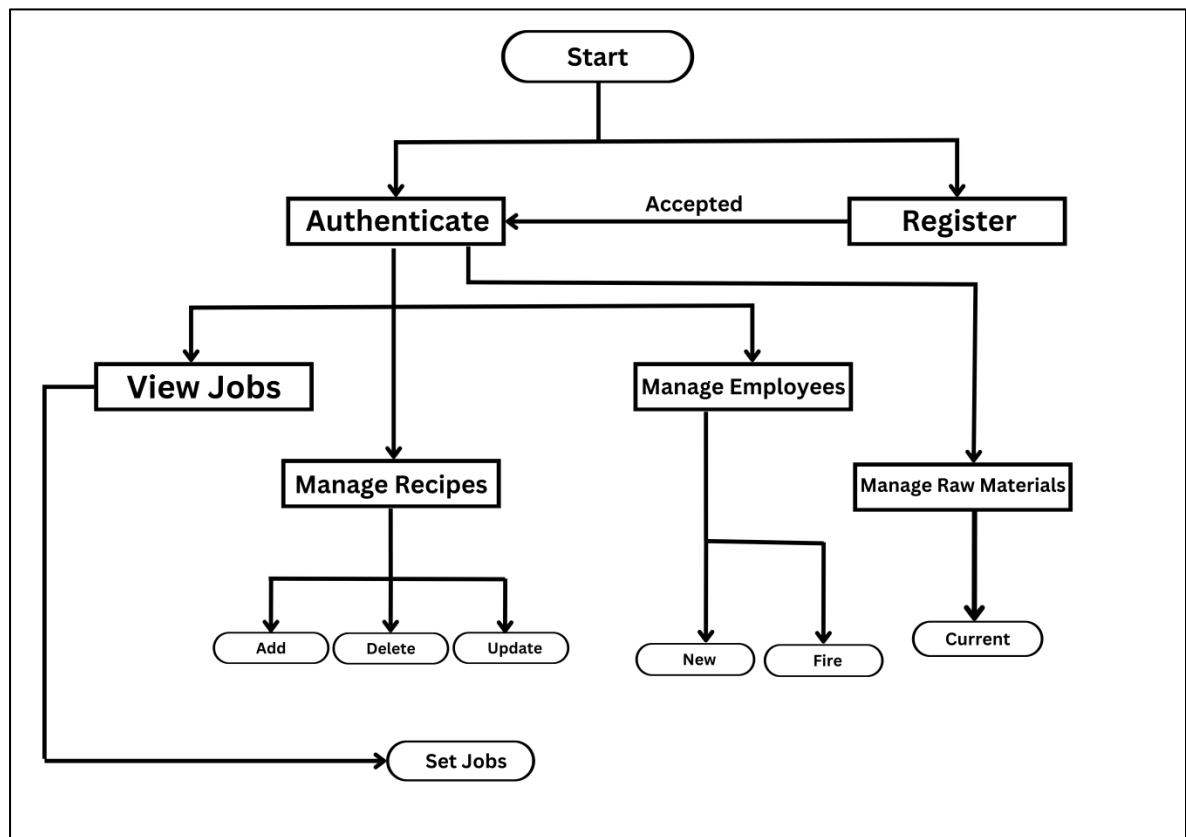


Figure 3: Admin View Flowchart

3.2.2 DFDs

Level 0: The image is a simple diagram showing an Admin inputting login or signup details into a Server, which interacts with a Database. The Database handles queries for recipes, raw materials, and employee data. The Server then generates Employee Accounts.

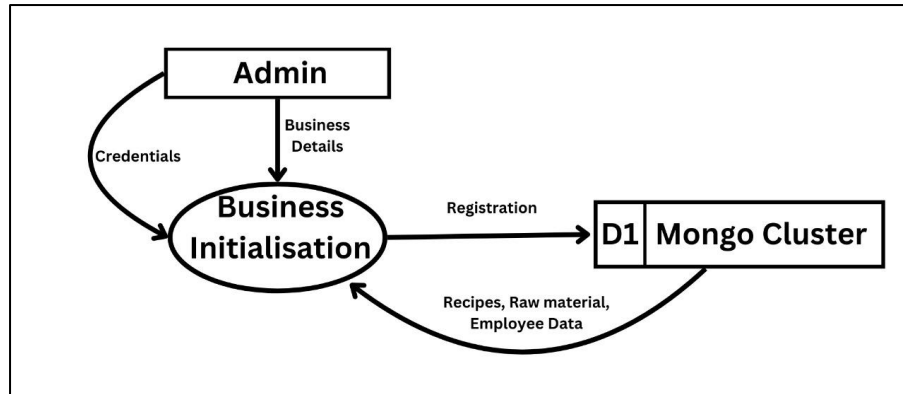


Figure 4: Level 0 DFD

Level 1: The diagram illustrates an administrative workflow in a system. The Admin enters login or signup details, which the Server uses to create accounts. The Server also processes job requests, interacts with a Database for queries on recipes, raw materials, and employee data, and generates Employee Accounts. Admin can also access a Secure Dashboard for alerts and raw material data, and provide approval for job requests.

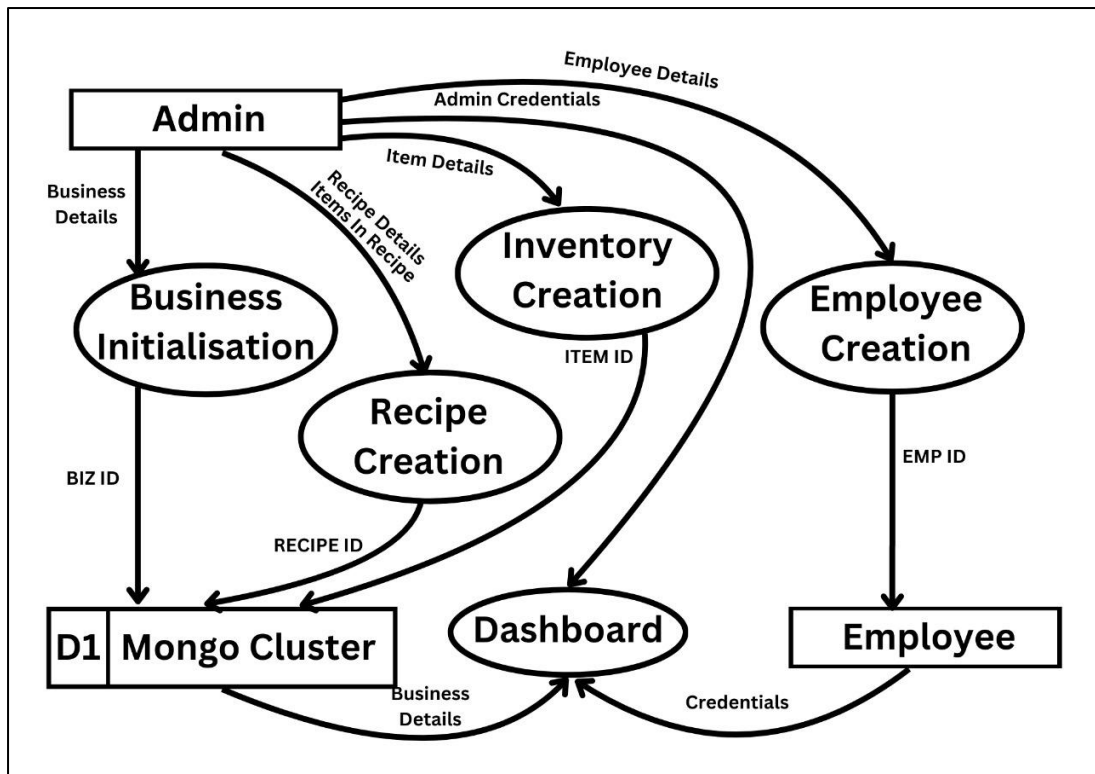


Figure 5: Level 1 DFD

Database Schema Design: The image displays a database schema for a business application. It includes a 'Businesses' table linked to 'Admins', 'Employees', 'Raw Materials', and 'Recipes' entities. 'Admins' and 'Employees' have 'Name', 'Role', 'Credentials', and 'Email' attributes, while 'Materials' are characterized by 'Name', 'Unit', and 'Available Quantity'. 'Recipes' are connected to 'Materials' and have attributes 'Name' and 'Quantity'.

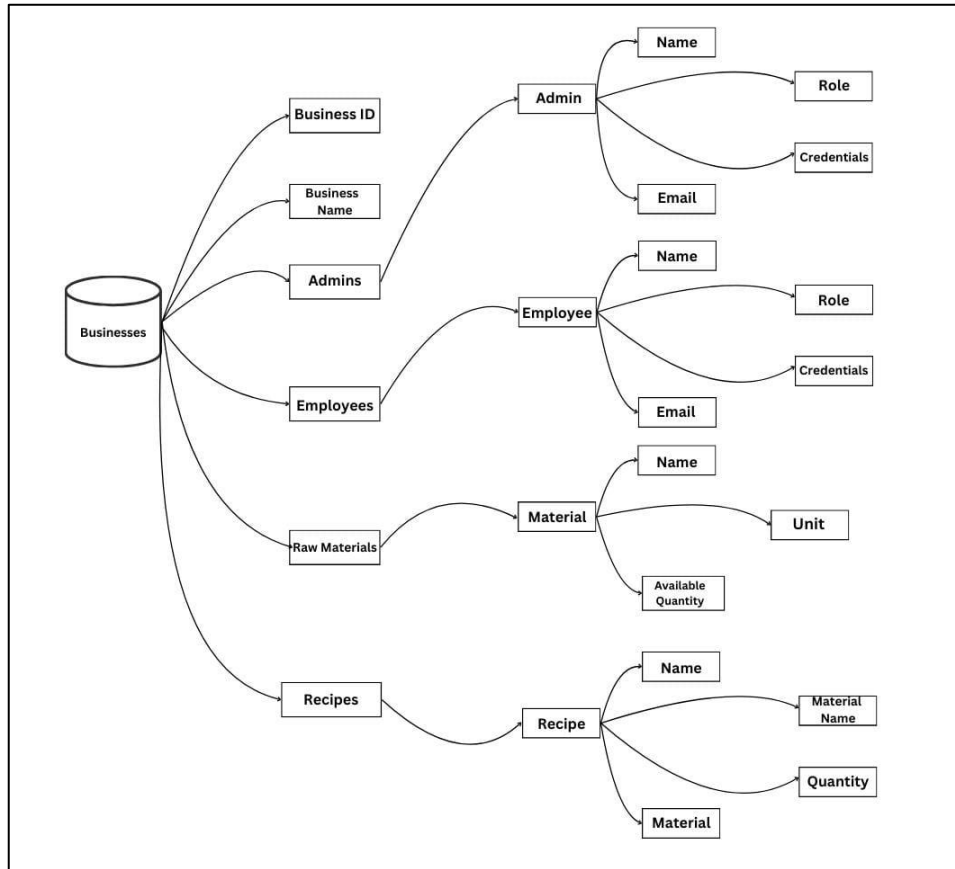


Figure 6 Database Scheme Design

Project Directory Structure: The image outlines a project directory structure for a web application, with a root directory containing three main folders: 'client' for React.js, 'server' for Flask, and 'Version Control Files'. Inside the 'client', there is an 'App.jsx' (Root Component) and a 'Components' directory with 'Functional' and 'Non-Functional Components'. The 'server' contains 'app.py' (the main file) and a 'models' directory. There is also a 'Tailwind config' file for styling in the 'client'.

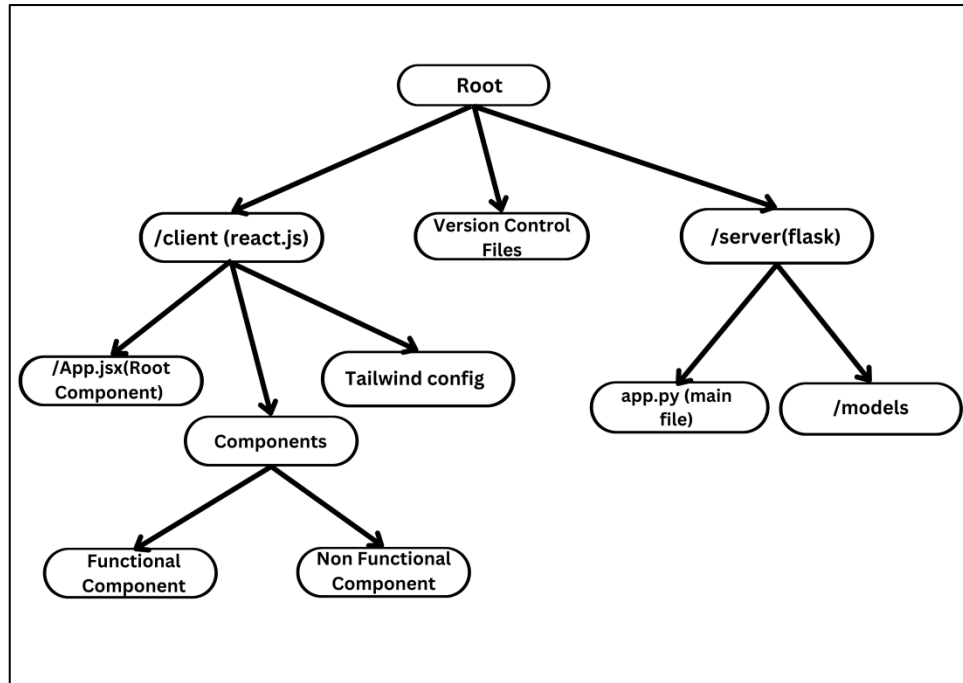


Figure 7 Directory Structure

3.3 User Interface Design

The user interface (UI) design in the Digitized Inventory Management System (IMS) for Natural Animal Diets is crafted with a user-centric approach, focusing on usability, functionality, and aesthetics. Here are key aspects of the UI design in the IMS project:

- 3.3.1 Intuitive Navigation:** The IMS UI features intuitive navigation menus, breadcrumbs, and interactive elements that guide users through different sections and functionalities of the system. Clear labeling and hierarchy ensure easy exploration and task completion.
- 3.3.2 Responsive Design:** The UI is designed to be responsive, adapting seamlessly to various screen sizes and devices. Whether accessed from desktops, laptops, tablets, or smartphones, the UI maintains consistency in layout, readability, and functionality.
- 3.3.3 Dashboard Overview:** The dashboard provides an at-a-glance overview of critical information such as inventory levels, production status, pending tasks, and alerts. Graphical representations, charts, and widgets offer visual insights into key metrics, enabling quick decision-making.

- 3.3.4 Data Visualization:** The IMS employs data visualization techniques such as charts, graphs, and tables to present complex data in a comprehensible format. Visual representations of inventory trends, production analytics, and performance metrics enhance data understanding and analysis.
- 3.3.5 Form Design and Validation:** Forms for data entry, such as adding new inventory items, updating employee information, or creating production schedules, are designed with clear labels, input fields, and validation checks. Real-time form validation ensures data accuracy and minimizes errors.
- 3.3.6 Interactive Elements:** The UI includes interactive elements such as dropdowns, checkboxes, radio buttons, and date pickers for user inputs. Interactive elements are designed for ease of use, accessibility, and consistent behavior across different browsers and devices.
- 3.3.7 Actionable Buttons and Icons:** Action buttons, icons, and tooltips provide clear cues for performing tasks such as saving data, submitting forms, generating reports, or initiating actions like reordering inventory or scheduling production runs.
- 3.3.8 Color Scheme and Visual Elements:** A harmonious color scheme, typography, and visual elements are used to create a visually appealing and professional UI. Color coding for status indicators, error messages, and alerts improves user comprehension and attention to critical information.
- 3.3.9 Accessibility Considerations:** The UI design incorporates accessibility considerations such as contrast ratios, text size, keyboard navigation, and screen reader compatibility to ensure inclusivity and compliance with accessibility standards.

Overall, the UI design in the IMS project prioritizes user experience, usability, and efficiency, empowering users with a visually engaging, intuitive, and functional interface for managing inventory, production processes, and business operations effectively.

3.4 Methodology

The methodology used in the Digitized Inventory Management System (IMS) project for Natural Animal Diets is the Agile methodology, specifically the Scrum framework. Here's an overview of how Agile and Scrum were applied in the IMS project:

- 3.4.1 Agile Principles:** The IMS project embraced the core principles of Agile development, including customer collaboration, iterative development, responding to change, and delivering working software incrementally.
- 3.4.2 Scrum Framework:** The Scrum framework, a cornerstone of Agile methodology, structures project activities into iterative sprints lasting 2-4 weeks, each focused on delivering prioritized features or user stories. Within this framework, a cross-functional Scrum team collaborates closely, leveraging diverse skills and expertise to achieve project goals. During sprint planning sessions, tasks are selected from a dynamic product backlog, ensuring clear objectives and commitment for the sprint. Daily stand-up meetings facilitate communication, transparency, and alignment within the team, enabling members to discuss progress, challenges, and planned activities. At the end of each sprint, a sprint review is conducted to demonstrate completed work to stakeholders, gather feedback, and validate user acceptance criteria, ensuring deliverables meet customer expectations. Additionally, sprint retrospectives provide an opportunity for the team to reflect on successes, identify areas for improvement, and implement action items to enhance performance in subsequent sprints. This continuous improvement cycle fosters a culture of learning, adaptation, and collaboration, driving efficiency, productivity, and value delivery. By embracing the Scrum framework, teams can effectively respond to changing requirements, deliver high-quality software increments, and align projects with customer needs and business objectives, ultimately enhancing customer satisfaction and project success.
- 3.4.3 Iterative Development:** The Agile and Scrum approach facilitated iterative development and incremental delivery of features. This iterative process allowed for continuous feedback, rapid adaptation to changes, and early validation of system functionality.

3.4.4 Customer Collaboration: Throughout the project, there was active collaboration with stakeholders, including Natural Animal Diets representatives and end-users. Regular feedback sessions, user acceptance testing (UAT), and product demonstrations ensured alignment with customer expectations and business objectives.

3.4.5 Adaptability and Flexibility: Agile and Scrum provided the IMS project with adaptability and flexibility to respond to evolving requirements, market dynamics, and emerging challenges. The iterative nature of development allowed for course corrections and adjustments based on feedback and lessons learned.

Overall, the Agile methodology with the Scrum framework was instrumental in driving the IMS project's success, promoting collaboration, transparency, continuous improvement, and delivering value to stakeholders through a customer-centric and iterative approach to software development.

Chapter 4. Implementation and Testing

4.1. Introduction to Languages, IDE's, Tools and Technologies

Introduction to Languages, IDEs, Tools, and Technologies Used in Project Work

4.1.1 Web App Development: In the Digitized Inventory Management System (IMS) project, web app development plays a crucial role in creating a user-friendly interface accessible from any device. Using modern web development technologies such as React.js for frontend development and Python with Flask for backend logic, the web app delivers responsive design, seamless navigation, and efficient data processing. This approach ensures that users can interact with the IMS effortlessly, accessing real-time inventory information, managing production cycles, and receiving automated alerts. The web app development focuses on delivering a robust, scalable, and intuitive platform for inventory management at Natural Animal Diets.

4.1.2 Languages and Frameworks: In the development of the Digitized Inventory Management System (IMS) for Natural Animal Diets, a combination of languages and frameworks is strategically employed to create a robust, scalable, and efficient application. Among these technologies are JavaScript with React.js for frontend development, Python with Flask for backend development, and MongoDB as the database management system (DBMS). Each technology brings unique features and capabilities to the IMS, contributing to its overall functionality and performance.

JavaScript, specifically React.js, plays a crucial role in frontend development, focusing on creating a dynamic and interactive user interface (UI) for the IMS. React.js is chosen for its component-based architecture, which allows developers to create reusable UI components. This approach enhances code modularity and reusability, making it easier to manage and maintain the frontend codebase. The use of a virtual DOM in React.js further improves performance by efficiently updating only the necessary parts of the UI, reducing rendering time and enhancing the overall user experience. Additionally, the JSX syntax in React.js enables developers to write UI components using a familiar HTML-like syntax, streamlining the process of UI development and management.

On the backend side, Python with Flask is utilized for API creation, server-side logic implementation, and overall backend development in the IMS. Flask is chosen for its lightweight and minimalist web framework, providing essential tools for building web applications without unnecessary overhead. This makes Flask ideal for projects with specific requirements and customization needs, allowing developers to tailor the backend logic to the unique needs of the IMS. Flask's support for RESTful API development is crucial for enabling seamless communication between the frontend and backend components of the IMS, facilitating data exchange and interaction. Additionally, Flask's URL routing capabilities enable developers to define URL patterns and map them to specific functions, ensuring efficient request handling and routing within the application.

For data management, MongoDB is selected as the DBMS for storing and managing various data related to inventory, users, and system configurations in the IMS. MongoDB's document-oriented storage approach stores data in JSON-like documents, providing flexibility in data modeling and schema-less design. This is advantageous for projects like the IMS with evolving data structures, as it allows for easy adaptation to changing requirements without the constraints of a rigid schema. MongoDB's scalability features are also crucial, supporting horizontal scaling across multiple nodes and clusters. This ensures that the IMS can handle increased data volume and user concurrency, scaling up as the application grows. Additionally, MongoDB offers efficient indexing and querying capabilities, enabling fast data retrieval, aggregation, and analysis for inventory management operations.

In summary, the combination of JavaScript with React.js for frontend development, Python with Flask for backend development, and MongoDB as the DBMS brings a comprehensive set of features and capabilities to the Digitized Inventory Management System for Natural Animal Diets. These technologies work together seamlessly to create a modern, efficient, and scalable application that meets the unique needs of inventory management in the agricultural sector.

4.1.3 IDEs and Tools: Integrated Development Environments (IDEs): Visual Studio Code (VS Code) serves as the primary Integrated Development Environment (IDE) for both frontend and backend development in the Digitized Inventory Management System (IMS) project for Natural Animal Diets. It provides a comprehensive environment for coding, debugging, and version control, enhancing developer productivity and facilitating seamless collaboration. VS Code's key features include syntax highlighting for various languages used in the IMS project, such as JavaScript, Python, HTML, and CSS. This feature improves code readability and navigation, making it easier for developers to write and manage code efficiently.

Another essential feature of VS Code is its integrated terminal, which allows developers to execute commands, run scripts, and manage project-related tasks without switching to external terminals. This integrated terminal streamlines the development workflow and enhances productivity by providing a centralized environment for code execution and project management tasks.

Git integration is another critical aspect of VS Code, allowing developers to leverage Git version control directly within the IDE. This integration includes features such as source control management, commit history, branch management, and conflict resolution, enabling developers to track changes, collaborate effectively, and maintain code integrity throughout the development process. Additionally, VS Code supports a wide range of extensions that enhance its functionality, including code snippets, linters, debuggers, and project management tools. These extensions further enhance developer productivity and customization, catering to specific project requirements and workflows.

In addition to VS Code, version control and collaboration tools such as Git and GitHub play a crucial role in the IMS project. Git enables version control, branching, merging, commit tracking, and code review workflows, ensuring code integrity and collaboration among developers. GitHub serves as a centralized repository for hosting code, managing issues, tracking project progress, and facilitating pull requests, code reviews, and continuous integration workflows. Its collaboration features, such as issue tracking, project boards, milestones, notifications, and discussions, enhance team communication and coordination during IMS development.

Overall, the combination of Visual Studio Code as the primary IDE, Git for version control, and GitHub for collaboration and project management collectively contributes to the efficient development, collaboration, version control, testing, deployment, and project management processes in the Digitized Inventory Management System (IMS) project for Natural Animal Diets. These tools ensure a streamlined and productive development lifecycle, allowing the development team to focus on delivering high-quality software solutions that meet the project's objectives and requirements.

4.1.4 Libraries and Modules: In the development of the Digitized Inventory Management System (IMS) for Natural Animal Diets, various libraries and modules are strategically utilized across the frontend and backend to enhance functionality, streamline development processes, and improve code efficiency.

On the frontend side, React.js libraries play a vital role in creating a dynamic and interactive user interface (UI) for the IMS application. React Router is used for client-side routing, enabling navigation between different pages and components within the application. This facilitates a seamless user experience by allowing users to navigate through various sections of the IMS efficiently. Material-UI is another essential library that provides pre-designed React components and UI elements, aiding in the development of responsive and visually appealing interfaces. These components enhance the overall aesthetics and usability of the IMS frontend. Additionally, Axios is utilized for making HTTP requests from the frontend to the backend API endpoints. It handles asynchronous data fetching and interaction with the Flask backend, ensuring smooth communication between frontend and backend components.

Moving to the backend, Flask libraries are instrumental in building the RESTful API and handling backend operations. Flask-RESTful extends Flask functionality for developing RESTful APIs, offering features like request parsing, resource routing, request handling, and response formatting. This simplifies API development and ensures standardized communication between frontend and backend components. Flask-CORS is employed to enable Cross-Origin Resource Sharing (CORS), allowing frontend requests from different origins to securely access backend resources. This enhances the security and accessibility of the IMS backend. Additionally, Flask-SQLAlchemy integrates SQLAlchemy with Flask,

simplifying database interactions and data modeling through ORM (Object-Relational Mapping). This integration streamlines database operations and ensures data consistency and integrity within the IMS backend.

For database management, PyMongo, a Python driver for MongoDB, is utilized in the Flask backend. PyMongo facilitates connecting to MongoDB databases, executing queries, and performing CRUD (Create, Read, Update, Delete) operations on inventory data and user information. It provides efficient data handling capabilities, ensuring seamless integration with MongoDB and effective management of IMS data.

Overall, these libraries and modules across the frontend and backend contribute significantly to the functionality, performance, and efficiency of the Digitized Inventory Management System for Natural Animal Diets. They enable seamless communication, robust API development, responsive UI design, and efficient data management, resulting in a scalable, user-friendly, and feature-rich IMS application.

4.1.5 Cross-Platform Compatibility: The Digitized Inventory Management System (IMS) developed for Natural Animal Diets prioritizes cross-platform compatibility to ensure broad accessibility and usability across various devices and operating systems. This approach encompasses several key strategies and practices to deliver a seamless and consistent user experience regardless of the user's device or platform.

Responsive Web Design (RWD) is fundamental to achieving cross-platform compatibility in the IMS frontend. Leveraging technologies like React.js and Daisy UI, the UI is designed to be responsive, dynamically adjusting to different screen sizes, resolutions, and orientations. Media queries, flexible layouts, and CSS grids are utilized to optimize the UI for desktops, laptops, tablets, and smartphones. This ensures that users can access and interact with the IMS interface comfortably across a wide range of devices.

Browser compatibility is another crucial aspect addressed in IMS development. The web application undergoes thorough testing and optimization to ensure compatibility with major web browsers such as Google Chrome, Mozilla Firefox, and Microsoft Edge. Cross-browser testing is conducted to validate consistent functionality and appearance, preventing layout inconsistencies, CSS rendering issues, and JavaScript compatibility problems. This

rigorous testing process ensures that users can access the IMS without encountering browser-related obstacles or errors.

The backend API, developed using Flask-RESTful, is designed for compatibility with diverse client applications regardless of the platform or technology stack they utilize. Following RESTful principles, the API offers standardized endpoints, HTTP methods, request/response formats, and authentication mechanisms. This standardization facilitates seamless integration and interoperability with various systems, applications, and platforms, enhancing the IMS's versatility and compatibility across the ecosystem.

Cross-platform testing plays a crucial role in validating the IMS's functionality, performance, and user experience across different platforms. This testing encompasses emulators, simulators, and real devices to simulate user interactions and scenarios across Windows, macOS, Linux, and other platforms. By identifying and addressing platform-specific issues, UI inconsistencies, and functionality variations, cross-platform testing ensures that the IMS delivers a consistent and reliable experience to all users, regardless of their device or operating system.

In conclusion, the IMS achieves cross-platform compatibility through a holistic approach that includes responsive web design, browser compatibility optimization, standardized backend APIs, and rigorous cross-platform testing. These strategies, combined with cloud deployment for scalability and accessibility, enable the IMS to offer a seamless and consistent user experience across diverse devices, platforms, and environments, enhancing usability and user satisfaction for Natural Animal Diets stakeholders.

4.1.6 User Experience Focus: The user experience (UX) is a central focus in the development of the Digitized Inventory Management System (IMS) for Natural Animal Diets, aiming to create an intuitive, efficient, and user-friendly platform for inventory management. Several key aspects highlight the user experience focus in the IMS project, showcasing a comprehensive approach to enhancing usability, productivity, and user satisfaction.

One of the primary focuses in the IMS project is intuitive interface design. The frontend of the IMS, built with React.js and Material-UI, features a clean and intuitive interface design. User interface elements, navigation menus, and actions are organized logically to facilitate easy navigation and task completion. Consistent design patterns, visual hierarchy, and UI

components ensure a cohesive and familiar experience for users, reducing cognitive load and enhancing usability.

Additionally, the IMS adopts responsive web design principles to ensure compatibility and optimal display across devices of varying screen sizes. This mobile-friendly approach improves accessibility and usability for users accessing the IMS from desktops, laptops, tablets, and smartphones. Mobile-friendly UI elements, touch-friendly interactions, and adaptive layouts further enhance the user experience on mobile devices.

Efficient information architecture is another key aspect of the IMS project's UX focus. The information architecture is structured intuitively, with well-organized categories, menus, and data visualization tools. This enables users to quickly locate information, navigate through inventory data, and perform tasks efficiently. Search functionality, filters, and sorting options are implemented to facilitate quick data retrieval and management, enhancing user productivity.

Interactive and feedback mechanisms play a crucial role in improving user satisfaction and reducing errors. Interactive elements such as dropdowns, tooltips, modals, and form validations provide immediate feedback and guidance to users. Real-time updates, notifications, and alerts keep users informed about inventory status, critical events, and actionable insights, enhancing decision-making capabilities.

The IMS also includes user-centric features and customization options tailored to meet the specific needs and preferences of inventory managers, administrators, and other stakeholders. Customizable dashboards, reports, and data visualizations empower users to personalize their workspace, prioritize tasks, and access relevant information efficiently.

Optimizing performance is crucial for a positive user experience. The IMS is designed to deliver fast loading times, responsive interactions, and smooth navigation, even with large datasets. Caching mechanisms, lazy loading of content, and server-side optimizations minimize latency and enhance system responsiveness.

Accessibility and inclusivity are key considerations in the IMS project's UX design. Accessibility features such as keyboard navigation, screen reader compatibility, and color contrast considerations are incorporated to ensure inclusivity and compliance with

accessibility standards. User feedback and usability testing are conducted regularly to gather insights, identify pain points, and iteratively improve the user experience based on user preferences and behavior.

In summary, the IMS project prioritizes intuitive interface design, responsive and mobile-friendly layouts, efficient information architecture, interactive feedback mechanisms, user-centric features, performance optimization, accessibility considerations, and inclusivity. By focusing on these key aspects, the IMS aims to deliver a superior user experience, enhancing user satisfaction, productivity, and engagement in inventory management tasks.

4.1.7 Conclusion: The Digitized Inventory Management System (IMS) for Natural Animal Diets employs a strategic array of tools, Integrated Development Environments (IDEs), libraries, and technologies to enhance productivity, collaboration, code quality, and user experience. Visual Studio Code (VS Code) serves as a robust IDE, offering integrated debugging, version control support, and extensive plugins for efficient development. Core technologies like React.js, Flask, and MongoDB ensure a responsive, scalable, and data-driven application. This strategic approach results in a user-centric, intuitive, and high-performance IMS solution, meeting the specific needs of Natural Animal Diets for efficient inventory management.

4.2. Algorithm/Pseudocode

class Business:

method start_session(user, business_id, login):

if login is True:

Remove password from user data

Remove password from user

Set session variables

Set 'logged_in' to True in session

Set 'user' to user in session

Set 'business_id' to business_id in session

Return JSON response with success message and status code 200

else:

Return business_id with status code 200

method signup():

BIZ_ID = get_business_id()

Extract data from request

Extract business_name, email, name, password from JSON request

Check for existing email address

If email exists in businesses collection:

Return "Email Already exists" with status code 201

else:

Generate employee ID

Generate employee_id

Create business object

Create business document with necessary fields

Create employee object

Create employee document with necessary fields

Encrypt password using pbkdf2_sha256

Encrypt password using pbkdf2_sha256

Insert business document into businesses collection

Insert business document into businesses collection

Push employee document into employees array of the business document

If employee document is successfully added to businesses collection:

Call start_session() with employee data, BIZ_ID, and login = False

Return JSON response with error message and status code 400

method signout():

If session is cleared:

Return JSON response with success message

else:

Return JSON response with error message

method login():

Extract business_id, email, and password from JSON request

Find business document by business_id

Find business document by business_id in businesses collection

If business document is found:

Find employee by email in employees array of business document

Find employee in employees array of business document

If employee is found and password matches:

Call start_session() with employee data, business_id, and login = True

Return JSON response with error message and status code 401

method get_business_id():

Try:

```
# Find document with _id "INFO01"

Find document with _id "INFO01" in businesses collection

# Increment BIZ_NO by 1

Increment BIZ_NO by 1

# Update BIZ_NO in the document

Update BIZ_NO in the document
```

Except:

```
# Create document with _id "INFO01" and BIZ_NO set to 1

Create document with _id "INFO01" and BIZ_NO set to 1

Set BIZ_NO to 1

Return "BIZ0" + BIZ_NO
```

class Business:

method start_session(user, business_id, login):

if login is True:

```
# Remove password from user data

Remove password from user

# Set session variables

Set 'logged_in' to True in session

Set 'user' to user in session

Set 'business_id' to business_id in session

Return JSON response with success message and status code 200
```

else:

Return business_id with status code 200

method signup():

BIZ_ID = get_business_id()

Extract data from request

Extract business_name, email, name, password from JSON request

Check for existing email address

If email exists in businesses collection:

Return "Email Already exists" with status code 201

else:

Generate employee ID

Generate employee_id

Create business object

Create business document with necessary fields

Create employee object

Create employee document with necessary fields

Encrypt password using pbkdf2_sha256

Encrypt password using pbkdf2_sha256

Insert business document into businesses collection

Insert business document into businesses collection

Push employee document into employees array of the business document

If employee document is successfully added to businesses collection:

Call start_session() with employee data, BIZ_ID, and login = False

Return JSON response with error message and status code 400

method signout():

 If session is cleared:

 Return JSON response with success message

 else:

 Return JSON response with error message

method login():

 Extract business_id, email, and password from JSON request

 # Find business document by business_id

 Find business document by business_id in businesses collection

 If business document is found:

 # Find employee by email in employees array of business document

 Find employee in employees array of business document

 If employee is found and password matches:

 Call start_session() with employee data, business_id, and login = True

 Return JSON response with error message and status code 401

method get_business_id():

 Try:

 # Find document with _id "INFO01"

 Find document with _id "INFO01" in businesses collection

Increment BIZ_NO by 1

Increment BIZ_NO by 1

Update BIZ_NO in the document

Update BIZ_NO in the document

Except:

Create document with _id "INFO01" and BIZ_NO set to 1

Create document with _id "INFO01" and BIZ_NO set to 1

Set BIZ_NO to 1

Return "BIZ0" + BIZ_NO

4.3. Testing Techniques

Testing techniques for the IMS project aimed at ensuring the robustness, reliability, and security of the system. Testing plays a critical role in identifying and addressing potential issues or vulnerabilities throughout the development lifecycle, from unit testing individual components to end-to-end integration testing of the entire workflow.

4.3.1 Unit Testing: Unit testing is crucial in ensuring the reliability, functionality, and security of the Digitized Inventory Management System (IMS) modules, particularly focusing on the User Authentication and Access Control, Inventory Management, Production Management, and Employee Management modules.

For the User Authentication and Access Control Module, unit tests would include verifying the login/logout functionality, checking for proper handling of invalid credentials, and ensuring secure password encryption using hashing algorithms. Additionally, session management tests would validate session expiration, persistence across pages, and prevention of session hijacking.

In the Inventory Management Module, unit tests cover CRUD operations for inventory items, ensuring items can be added, edited, viewed, and deleted without data loss or integrity

issues. This includes testing calculations and business logic related to inventory quantities, stock levels, reorder points, and pricing rules.

The Production Management Module undergoes testing for order creation, updates, and cancellation functionalities, ensuring accurate processing and status updates. Scheduling logic is validated to handle scenarios like resource unavailability, overlapping schedules, and production delays.

For the Employee Management Module, unit tests focus on CRUD operations for employee data, ensuring that employee profiles can be managed without data corruption. Role assignments and permissions are tested to verify proper role-based access control (RBAC), role assignments, permission enforcement, and denial of unauthorized access attempts.

Unit testing in each module involves creating test cases, executing tests, and verifying expected outcomes against actual results. Tools like Jest, PyTest, and Mocha can be used for automated testing, improving efficiency and coverage. Test coverage reports and code quality metrics are analyzed to identify areas for improvement and ensure comprehensive testing across all functionalities.

By conducting thorough unit testing, the IMS can achieve high reliability, functionality, and security, meeting user requirements, minimizing bugs, and ensuring a robust inventory management system for Natural Animal Diets.

4.3.2 Integration Testing: Integration testing plays a vital role in ensuring the seamless functioning and interoperability of different modules within the Digitized Inventory Management System (IMS), focusing on the User Authentication and Access Control, Inventory Management, Production Management, and Employee Management modules.

For the User Authentication and Access Control Module, integration testing involves verifying the integration with user databases to validate user credentials and ensure secure authentication. Additionally, integration with LDAP (Lightweight Directory Access Protocol) is tested if applicable, ensuring compatibility with directory services for centralized user authentication and access control.

In the Inventory Management Module, integration testing includes testing integration with barcode scanners, RFID systems, or IoT devices for efficient inventory tracking and data

capture. This ensures that data from external devices is accurately synchronized with the IMS, maintaining inventory accuracy and visibility. Integration with ERP (Enterprise Resource Planning) systems is also validated, enabling seamless data exchange for inventory updates, procurement processes, and order fulfillment.

The Production Management Module undergoes integration testing for integration with production machines, sensors, or PLCs (Programmable Logic Controllers) for real-time data acquisition and monitoring. This includes testing data synchronization and accuracy between the production management module and production equipment, ensuring smooth production processes and data consistency. Integration with supply chain systems is also tested to facilitate production planning, materials management, and supply chain visibility.

For the Employee Management Module, integration testing involves verifying integration with HR systems for employee data synchronization, including employee profiles, roles, and organizational hierarchies. This includes testing LDAP or Active Directory integration for user authentication and access control, ensuring seamless access management based on user roles and permissions.

Integration testing in each module includes testing various scenarios, data flows, and system interactions to validate interoperability, data accuracy, and functionality across integrated components. Automated testing tools, API testing frameworks, and mock services can be utilized to simulate integration scenarios and ensure comprehensive testing coverage. By conducting thorough integration testing, the IMS can achieve seamless integration, data consistency, and optimal performance across all modules, enhancing the overall efficiency and effectiveness of inventory management for Natural Animal Diets.

4.3.3 System Testing: System testing in the Digitized Inventory Management System (IMS) for Natural Animal Diets encompasses comprehensive testing scenarios across the User Authentication and Access Control, Inventory Management, Production Management, and Employee Management modules to ensure robustness, functionality, and data integrity.

In the User Authentication and Access Control Module, end-to-end login and logout scenarios are tested to validate the system's authentication mechanisms. This includes testing successful login, handling of failed login attempts, session timeout management, and proper logout functionality to maintain session security. Access control policies are also validated to

ensure that users can access functionalities and data based on their assigned roles and permissions, preventing unauthorized access.

The Inventory Management Module undergoes testing of end-to-end inventory workflows, including item creation, updates, inventory searches, stock level alerts, and reporting functionalities. Data integrity and consistency are validated across inventory transactions to ensure that inventory data remains accurate and up-to-date, reflecting real-time changes in stock levels and item details.

In the Production Management Module, end-to-end production workflows are tested from order creation to completion, encompassing resource allocation, scheduling, production progress tracking, and order fulfillment processes. Error handling mechanisms are validated for production errors, exceptions, and system failures, with testing of recovery procedures to restore system functionality and maintain data consistency in case of unexpected issues.

The Employee Management Module undergoes end-to-end testing of employee management functionalities, including employee profile management, role assignments, permissions, and leave management workflows. Data privacy and security measures are validated to protect sensitive employee information, ensuring that access is restricted based on Role-Based Access Control (RBAC) policies and that data privacy regulations are adhered to.

System testing ensures that each module functions as intended, with seamless integration between modules, accurate data processing, error handling capabilities, and adherence to security and privacy standards. By validating end-to-end workflows, data integrity, access controls, and error handling mechanisms, the IMS for Natural Animal Diets can provide a reliable and efficient inventory management solution that meets user expectations and business requirements.

4.3.4 Acceptance Testing: Acceptance testing in the Digitized Inventory Management System (IMS) for Natural Animal Diets focuses on user acceptance across key modules to ensure usability, functionality, and user satisfaction.

In the User Authentication and Access Control Module, acceptance testing involves end-users testing login/logout flows to ensure ease of use, clear error messages, and secure authentication processes. Additionally, user acceptance for password recovery, account

creation, and profile management features is validated to ensure users can manage their accounts effectively.

For the Inventory Management Module, acceptance testing includes user acceptance for inventory search, item management, stock level alerts, and reporting features. The goal is to ensure that users can efficiently perform inventory tasks such as searching for items, managing inventory levels, receiving alerts for low stock, and generating reports as needed.

In the Production Management Module, acceptance testing focuses on production workflows such as order creation, scheduling, progress tracking, and order fulfillment. End-users participate in testing these processes to validate user satisfaction with production management functionalities, ensuring that users can easily create and manage production orders, track progress, and fulfill orders efficiently.

The Employee Management Module undergoes acceptance testing for employee profile management, role assignments, permissions, and leave management features. This testing phase ensures that employees can easily navigate and use HR functionalities, manage their profiles, view assigned roles and permissions, and request leaves as necessary.

Overall, acceptance testing involves active participation from end-users to validate that the IMS meets their needs, is easy to use, and provides the necessary functionalities for efficient inventory management, production workflows, and employee management. By incorporating user feedback and addressing usability issues identified during acceptance testing, the IMS can deliver a user-friendly and effective solution for Natural Animal Diets' inventory management needs.

The testing process for Natural Animal Diets' Digitized Inventory Management System (IMS) rigorously ensures functionality, reliability, and user satisfaction. Unit testing verifies individual components' correctness, integration testing tests system interoperability, and system testing evaluates end-to-end workflows and data integrity. This comprehensive approach identifies and rectifies issues early, ensuring the IMS meets quality standards, user expectations, and business objectives, contributing significantly to the success of inventory management operations at Natural Animal Diets.

4.4. Test Cases designed for the project work

4.4.1 User Authentication and Access Control Module:

Test Case 1: Login Functionality- The user authentication test case verifies the system's ability to allow users to log in with valid credentials. The steps include entering a valid username and password, clicking the login button, and confirming successful login by redirection to the dashboard.

Test Case 2: Invalid Login Attempt- The test case validates how the system handles invalid login attempts. It involves entering an incorrect username or password, clicking the login button, and verifying that the system displays an appropriate error message.

Test Case 3: Access Control- The test case evaluates the role-based access control (RBAC) functionalities of the system. It includes logging in with different user roles such as admin, employee, and manager, then verifying the access permissions associated with each role. For instance, admins should have access to all features while employees may have restricted access to specific functionalities.

4.4.2 Inventory Management Module:

Test Case 4: Add Inventory Item- The test case verifies the process of adding a new inventory item. It involves navigating to the inventory management section, clicking on the "Add Item" button, filling in the item details in the form, submitting the form, and finally confirming that the item is successfully added to the inventory list.

Test Case 5: Update Inventory Item- The test case focuses on updating an existing inventory item. It includes selecting an item from the inventory list, clicking on the "Edit" button, modifying the item details, saving the changes, and confirming that the updates are correctly reflected in the inventory system.

4.4.3 Production Management Module:

Test Case 6: Create Production Order- The test case involves verifying the creation of a production order. It includes navigating to the production management section, clicking on the "New Order" button, entering order details, submitting the order, and confirming that the order is successfully created and visible in the production queue.

Test Case 7: Update Production Status- The test case aims to validate the updating of a production order's status. It includes selecting a production order from the queue, changing its status to either "In Progress" or "Completed," and verifying that the status update is accurately reflected in the system.

4.4.4 Employee Management Module:

Test Case 8: Add Employee- The test case involves validating the addition of a new employee profile. It includes navigating to the employee management section, clicking on the "Add Employee" button, filling in employee details, saving the information, and verifying that the employee profile is successfully created in the system.

Test Case 9: Update Employee Information- The test case aims to validate the process of updating employee information. It involves selecting an employee from the list, editing details such as roles or contact information, saving the changes, and verifying that the updates are correctly reflected in the system.

4.4.5 Alert System Module:

Test Case 10: Threshold Alert Generation- The test case checks the functionality of generating an alert when the stock level falls below a predefined threshold. It involves simulating a decrease in stock to trigger the threshold condition, waiting for the system to generate the alert notification, and then verifying that the alert notification is indeed generated as expected.

Test Case 11: Email Alert Notification- The test case focuses on verifying the functionality of sending an email alert notification to the admin's email address when a stock level alert is triggered. It involves intentionally reducing the stock level below the threshold to trigger the alert, checking the admin's email inbox for the alert notification, and confirming that the email contains pertinent details regarding the low stock alert.

Test Case 12: Alert Settings Configuration- The test case involves configuring alert threshold values and notification settings in the alert system. It includes navigating to the configuration settings, setting threshold values for various inventory items, configuring email and SMS notification settings for alerts, saving the settings, and verifying that they are successfully applied and functional.

Therefore, these meticulously designed test cases for the IMS project at Natural Animal Diets cover a comprehensive range of functionalities across various modules. From user authentication and access control to inventory and production management, employee management, and the critical alert system, each test case aims to validate system reliability, functionality, and user experience. By simulating real-world scenarios and ensuring seamless integration with external systems, these test cases ensure that the IMS meets quality standards, addresses business requirements, and enhances operational efficiency. Thorough testing is instrumental in identifying and resolving potential issues early, ultimately contributing to the successful deployment and utilization of the IMS.

Chapter 5. Results and Discussions

5.1. User Interface Representation

- 5.1.1 User Authentication and Access Control Module:** The User Authentication and Access Control Module features a login page with username and password fields, a "Forgot Password?" link, and a "Login" button. Key features include secure login with password encryption, password recovery/reset functionality, and role-based access control (RBAC) for different user roles..
- 5.1.2 Inventory Management Module:** The Inventory Management Module includes an Inventory Dashboard providing an overview of total inventory count, low-stock items, and recent inventory activities, along with an Inventory List View showing item details. Key features encompass adding, editing, and deleting inventory items, setting threshold levels for stock alerts, and generating inventory reports and analytics.
- 5.1.3 Production Management Module:** The Production Management Module includes a Production Dashboard showing production order status, pending tasks, and schedules, alongside a Production Order Form for creating new production orders. Key features involve scheduling production runs, resource allocation, tracking production progress and order status, and generating production reports and analytics.
- 5.1.4 Employee Management Module:** The Employee Management Module features an Employee Directory displaying employee profiles with details such as name, role, contact information, and photos. It includes a Task Assignment Interface for assigning tasks, projects, or duties with deadlines and priorities. Key features encompass managing employee profiles and roles, tracking tasks, performance, and attendance, as well as handling leave requests and approvals.
- 5.1.5 Alert System Module:** The Alert System Module includes an Alert Dashboard for monitoring active alerts, their types, and severity levels. Users can configure alerts through the Alert Configuration Page, setting threshold levels, notification preferences, and recipient lists. Key features involve generating alerts for low stock,

production delays, or system issues, notifying users via email, SMS, or in-app notifications, and offering actionable insights and recommendations based on alert data.

Each module's UI is designed to be intuitive, user-friendly, and visually appealing, with clear navigation, interactive elements, and informative dashboards to facilitate efficient management of inventory, production, employees, and alerts within the IMS system.

Design Principles: The design principles for the system emphasize simplicity, consistency, accessibility, and feedback. Simplicity ensures that interfaces are clean and intuitive, reducing cognitive load for all users. Consistency is maintained throughout the app with unified design elements and navigation patterns, promoting a cohesive user experience. Accessibility considerations are integrated to accommodate users with diverse needs and abilities, adhering to accessibility guidelines. Real-time feedback and visual cues are implemented to improve interaction, providing clear indications of status and actions, further enhancing the usability and user satisfaction of the system.

Usability Considerations: The system prioritizes mobile optimization, employing responsive design for smaller screens to ensure optimal viewing and interaction. Efficient workflows are implemented, reducing the number of steps needed to complete tasks and enhancing overall efficiency. Visual hierarchy is carefully structured with clear information organization, utilizing color, typography, and spacing effectively to guide user attention and improve usability.

Impact on User Experience: Effective design in the IMS project at Natural Animal Diets significantly impacts user experience by prioritizing usability, efficiency, engagement, accessibility, feedback mechanisms, consistency, and mobile responsiveness. Intuitive interfaces with clear navigation, streamlined workflows, engaging elements, and accessible features enhance user productivity, satisfaction, and adoption rates. Thoughtful design considerations, such as feedback mechanisms for error handling, consistency in design patterns, and mobile-friendly interfaces, contribute to a positive user experience, fostering user confidence, reducing frustration, and promoting seamless interaction with the system.

5.2 Brief Description of Various Modules of the system

5.2.1 User Authentication and Access Control Module: The User Authentication and Access Control Module ensures secure login/logout processes, employing encryption techniques like bcrypt or SHA-256 for password storage. It includes features such as password complexity rules, expiration policies, and hashing methods to enhance security. Role-Based Access Control (RBAC) categorizes users into roles like admin, manager, or employee, each with specific permissions. Admins have the authority to manage roles and permissions, ensuring users access only relevant functionalities and data, thus enhancing system security and data integrity. User sessions are terminated securely during logout, preventing unauthorized access and maintaining user privacy. Overall, the module plays a crucial role in safeguarding system resources, protecting sensitive information, and ensuring compliance with security standards and best practices.

5.2.2 Inventory Management Module: The Inventory Management Module allows users to efficiently track inventory by adding, editing, or deleting items with detailed information like name, description, quantity, cost, supplier details, and storage location. Real-time inventory tracking includes monitoring movements such as receipts, transfers, and withdrawals. Stock alerts are set based on threshold levels, triggering notifications via email, SMS, or dashboard alerts when inventory levels fall below set limits. Additionally, the module provides comprehensive reporting and analysis features, generating reports on inventory status, movement history, reorder point analysis, slow-moving items, and inventory valuation. These reports offer valuable insights for inventory planning, forecasting, and decision-making processes, enhancing overall inventory management efficiency and accuracy.

5.2.3 Production Management Module: The Production Management Module facilitates the creation, modification, and cancellation of production orders, allowing users to specify product details, quantities, schedules, priorities, and required resources like materials and labor. It optimizes resource allocation by considering production schedules, resource availability, and priorities, enabling efficient scheduling of production runs and task assignments. The module may incorporate workflow

automation features to streamline processes, including automated scheduling, task assignments, quality checks, and real-time production monitoring. Integration with IoT devices or production machinery enhances data capture and monitoring capabilities, contributing to improved production efficiency, reduced manual intervention, and enhanced decision-making in production workflows.

5.2.4 Employee Management Module: The Employee Management module centralizes employee data, storing personal details, contact information, job roles, qualifications, training records, and performance evaluations. It facilitates task assignment by enabling managers to assign tasks, projects, or duties to employees, tracking progress, deadlines, and outcomes. Additionally, the module supports task prioritization, dependencies, and scheduling for efficient task management. Leave management features allow employees to request leaves, specifying types, dates, and reasons. Managers review and approve these requests, ensuring sufficient staffing levels and adherence to leave policies. This comprehensive module streamlines HR processes, enhances communication between employees and managers, and promotes efficient task allocation and leave management within the organization.

5.2.5 Alert System Module: The Alert System Module plays a crucial role in monitoring critical aspects of the inventory and production processes. Administrators set threshold levels for key metrics such as inventory levels or machine downtime, triggering automated alerts when these thresholds are exceeded. These alerts are communicated swiftly to designated users or groups through various channels like email, SMS, or push notifications, providing detailed information such as the alert type, affected item or process, severity level, and recommended actions. Moreover, the module goes beyond mere alerts by analyzing data trends and historical patterns to offer actionable insights and recommendations. These insights may include suggestions for inventory replenishment, production adjustments, resource reallocation, and proactive maintenance strategies, empowering organizations to respond promptly to emerging

5.3 Snapshots of system with brief detail of each and discussion.

5.3.1 Landing Page: The landing page prominently showcases the login module, serving as the secure gateway for authorized users to access the IMS system. It features user-friendly login fields for username and password entry, along with options for password recovery. The login module ensures a streamlined and secure authentication process for seamless access to system functionalities.

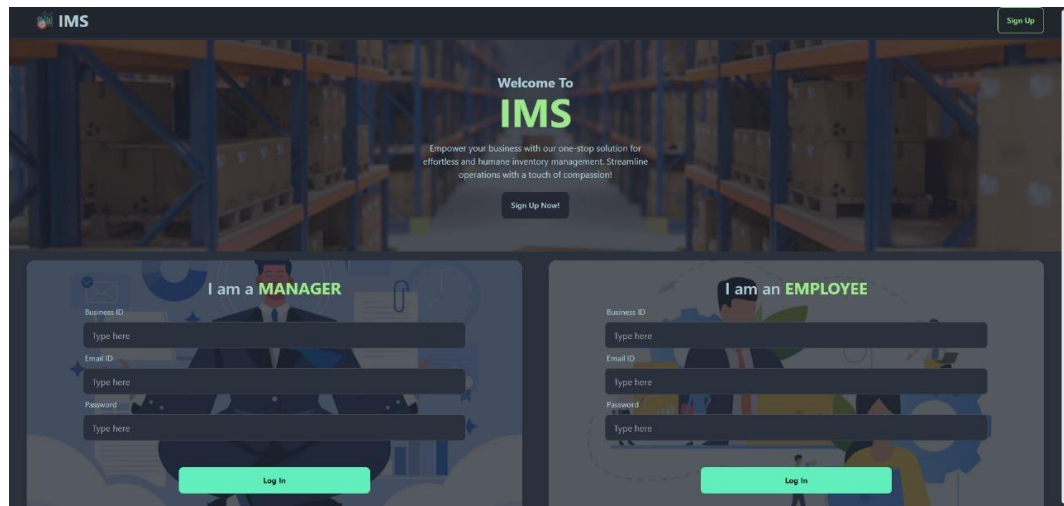


Figure 8 Landing Page

5.3.2 SignUp Page: The signup page allows new users to create accounts for accessing the IMS system. It includes fields for entering personal information, creating a username and password.

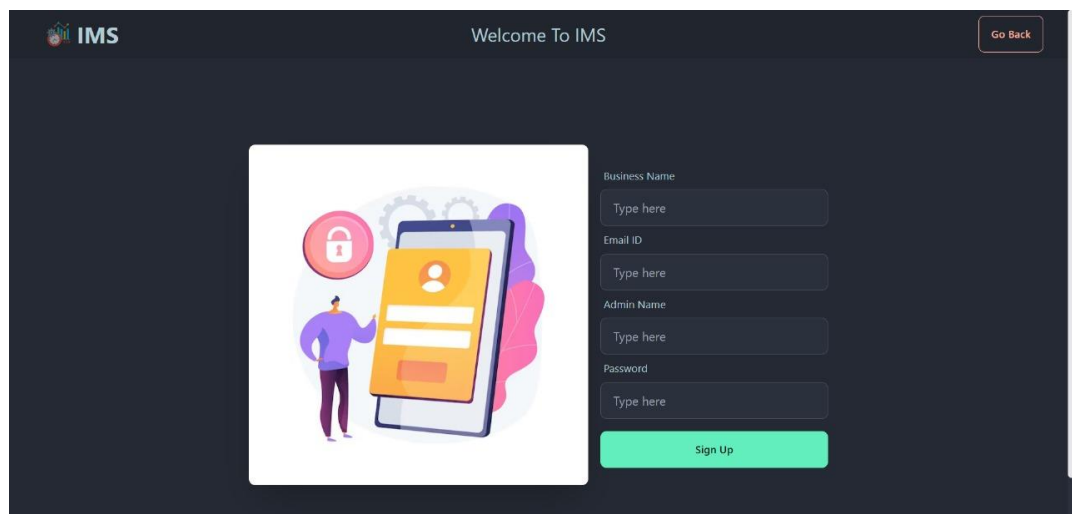


Figure 9 SignUp Page

5.3.3 Jobs Dashboard: The jobs dashboard page in the IMS project provides an overview of all job-related activities. It displays job orders, task assignments, progress status, priority levels, and deadlines. Users can efficiently track, manage, and prioritize jobs from this centralized dashboard, ensuring effective job management and workflow optimization.

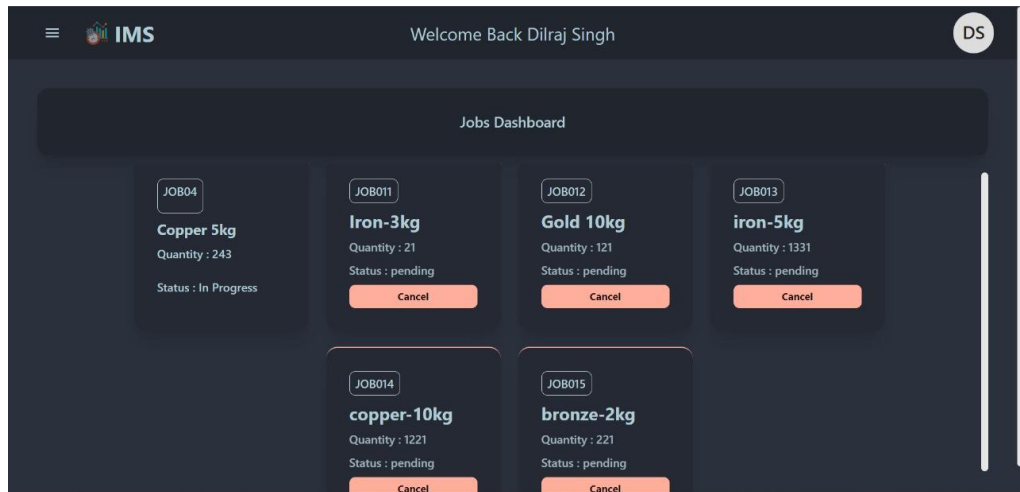


Figure 10 Job Dashboard Page

5.3.4 Inventory Items: The inventory items page in the IMS project displays a comprehensive list of all inventory items. It includes details such as item name, description, quantity. Users can easily view, update, and manage inventory items from this centralized page, facilitating efficient inventory management and stock tracking.

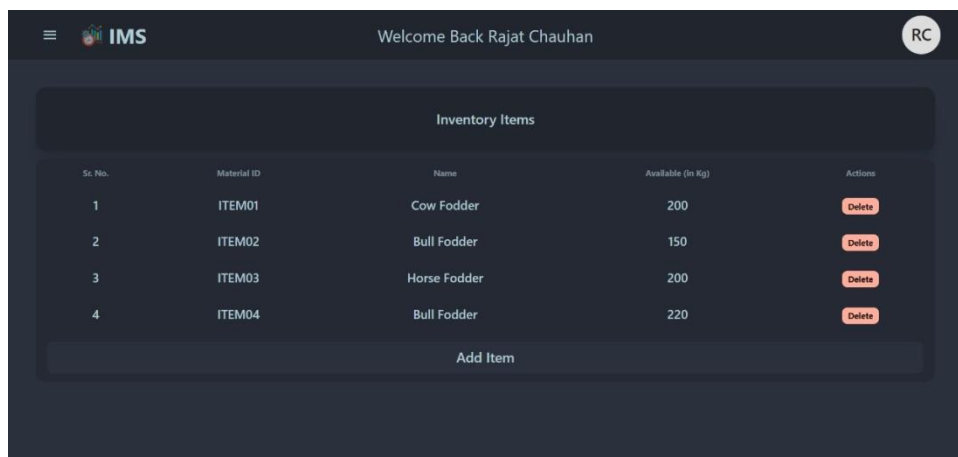


Figure 11 Inventory Items Page

5.3.5 Recipes: The recipes page in the IMS project presents a collection of curated recipes for various products or items. Each recipe includes details such as ingredients, quantities, preparation instructions. Users can access, create, modify, and manage recipes from this page, supporting standardized production processes and ensuring consistency in product quality.



Figure 12 Recipes Page

5.3.6 Job Creation: The job creation feature in the IMS project allows authorized users to create new job orders within the system. It includes fields for entering job details such as title, description, priority level, deadline, and required resources. Upon creation, the job is added to the system, enabling efficient task allocation and progress tracking.

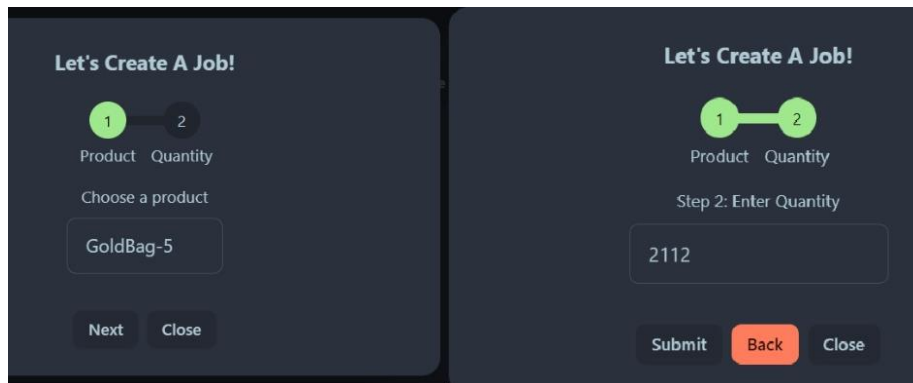
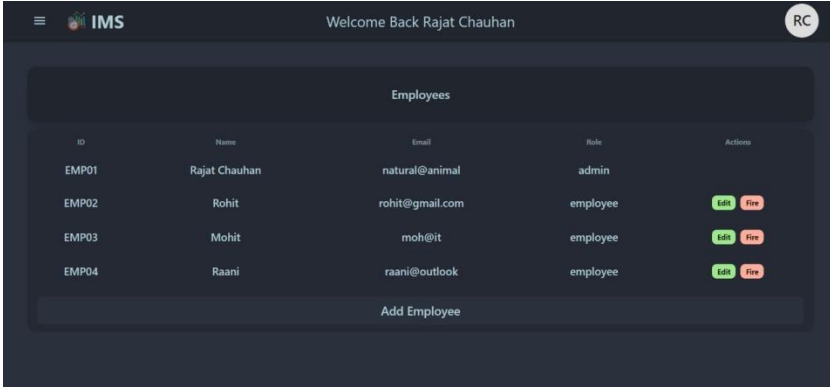


Figure 13 Job Creation Page

5.3.7 Employees: The employees page in the IMS project provides a comprehensive view of all employees within the organization. It includes employee details such as name, ID, email, role. This page facilitates HR management, task assignment, leave management, and overall workforce coordination within the system.

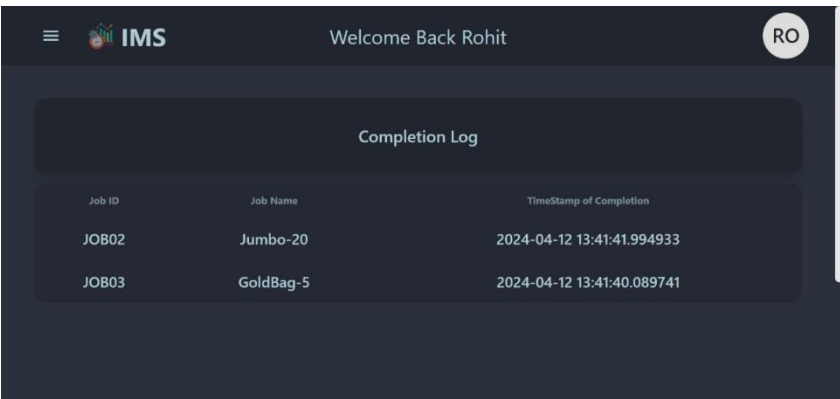


ID	Name	Email	Role	Actions
EMP01	Rajat Chauhan	natural@animal	admin	
EMP02	Rohit	rohit@gmail.com	employee	<button>Edit</button> <button>Fire</button>
EMP03	Mohit	moh@it	employee	<button>Edit</button> <button>Fire</button>
EMP04	Raani	raani@outlook	employee	<button>Edit</button> <button>Fire</button>

[Add Employee](#)

Figure 14 Employees Page

5.3.8 Log Completion: The completion log in the IMS project records and tracks the completion status of various tasks, jobs, or activities within the system. The completion log provides a chronological record of completed tasks, aiding in performance analysis, project evaluation, and historical tracking of work accomplishments.



Job ID	Job Name	TimeStamp of Completion
JOB02	Jumbo-20	2024-04-12 13:41:41.994933
JOB03	GoldBag-5	2024-04-12 13:41:40.089741

Figure 15 Log Completion Page

5.4 Back Ends Representation

5.4.1. Snapshots of Database Tables with brief description

Jobs Table: This table stores information about job orders, including job title, description, priority level, deadline, assigned tasks, and associated resources.

```
quantity: 100
▼ jobs: Array (7)
  ▼ 0: Object
    job_id: "JOB04"
    product_id: "PROD018"
    product: "ekuspre"
    quantity: 243
    status: "in progress"
  ▼ 1: Object
    job_id: "JOB05"
    product_id: "PROD018"
    product: "ekuspre"
    quantity: 243
    status: "finish"
  ▼ 2: Object
    job_id: "JOB07"
    product_id: "PROD025"
    product: "drugs"
    quantity: 2
    status: "finish"
  ▼ 3: Object
    job_id: "JOB08"
    product_id: "PROD022"
    product: "administrator"
    quantity: 16
    status: "finish"
    completion_time: "2024-04-12 05:09:45.036126"
  ▼ 4: Object
```

Figure 16 MongoDB (Jobs)

Products Table: The products table contains data related to inventory items or products available in the system. It includes fields such as product name, description, quantity in stock, supplier information, and cost details.

```
item_id: "ITEM07"
name: "Xanny"
current_stock: -1450
threshold_stock: 100
▼ products: Array (6)
  ▼ 0: Object
    product_id: "PROD017"
    name: "1221"
    batch_size: "21"
    ▼ items: Array (7)
      ▼ 0: Object
        item_id: "ITEM01"
        quantity: 1221
      ▼ 1: Object
        item_id: "ITEM01"
        quantity: 1221
      ▼ 2: Object
        item_id: "ITEM04"
        quantity: 121
      ▼ 3: Object
        item_id: "ITEM03"
        quantity: 0
      ▼ 4: Object
        item_id: "ITEM02"
        quantity: 0
      ▼ 5: Object
        item_id: "ITEM06"
        quantity: 0
```

Figure 17 MongoDB (Products)

Items Table: The items table stores details about specific items within the inventory, such as item code, category, location, and status (e.g., in stock, out of stock). It provides a granular view of individual inventory items, aiding in inventory tracking, replenishment, and stock level monitoring.

```

▼ items : Array (7)
  ▼ 0: Object
    item_id : "ITEM01"
    name : "Adderal"
    current_stock : -89224.29906542056
    threshold_stock : 120
  ▼ 1: Object
    item_id : "ITEM02"
    name : "Nicotine"
    current_stock : -550
    threshold_stock : 120
  ▼ 2: Object
    item_id : "ITEM03"
    name : "Cintholl"
    current_stock : 200
    threshold_stock : 300
  ▼ 3: Object
    item_id : "ITEM04"
    name : "Promethazine"
    current_stock : -7666.66666666667
    threshold_stock : 10
  ▼ 4: Object
    item_id : "ITEM05"
    name : "Addy"
    current_stock : 137.14018691588808
    threshold_stock : 120
  ▼ 5: Object
    item_id : "ITEM06"

```

Figure 18 MongoDB (Items)

Employees Table: This table maintains employee records, including employee ID, name, contact information, role/job title, department, and qualifications. It also tracks employee tasks, project assignments, leave requests, and attendance data. The employees table supports workforce management, task allocation, and HR-related functions within the system.

```

_id: "BIZ013"
business_name: "Natural Animal Diets"
emp_no: 18
item_no: 7
product_no: 25
job_no: 11
▼ employees : Array (6)
  ▼ 0: Object
    employee_id: "EMP013"
    name: "Dilraj Singh"
    email: "nad@gmail.com"
    password: "$pbkdf2-sha256$29000$Tcl5r7U2JkToPQfgv3fy/g$XqJkUBFS0/0y6K4wbMxRYxTTV1..."
    role: "admin"
  ▼ 1: Object
    employee_id: "EMP014"
    name: "Dilraj Singh"
    email: "ddqd.ddd"
    password: "$pbkdf2-sha256$29000$W4tRCmEMoZR5KwQkPISog$KqF0fhq.8931zLa5peEusCIJun..."
    role: "admin"
  ▼ 2: Object
    employee_id: "EMP015"
    name: "bitchassnigaa"
    email: "abc@gmail.com"
    password: "$pbkdf2-sha256$29000$ZIxrrrW2ViplDG6sFQLg3A$02faXhswWwvyobCAEFjdaPEAy..."
    role: "employee"
  ▼ 3: Object
    employee_id: "EMP016"

```

Figure 19 MongoDB (Employees)

Chapter 6. Conclusion and Future Scope

6.1 Conclusion

The Digitized Inventory Management System (IMS) developed for Natural Animal Diets represents a comprehensive solution aimed at addressing the challenges faced by the Fast-Moving Consumer Goods (FMCG) industry in inventory management. Through a strategic combination of modern technologies, robust development methodologies, and user-centric design principles, the IMS project delivers a sophisticated and efficient platform for managing inventory, production cycles, workforce planning, and quality control.

One of the key strengths of the IMS is its adaptability and scalability. Built using React.js for the frontend, Flask for the backend, and MongoDB for data storage, the IMS is designed to handle the complexities of inventory management in the FMCG sector. The use of a RESTful API architecture ensures seamless communication between frontend and backend components, enabling real-time data processing, inventory monitoring, and alert generation.

The user experience (UX) focus in the IMS project is evident in its intuitive interface design, responsive layout, interactive features, and customizable dashboards. Users, including inventory managers, administrators, and production teams, benefit from a user-friendly platform that streamlines tasks, provides actionable insights, and enhances decision-making capabilities.

Furthermore, the IMS project emphasizes performance optimization and reliability. Performance testing, load testing, and scalability assessments are conducted to ensure that the system can handle large datasets, concurrent users, and peak operational loads without compromising speed or stability. Security measures such as data encryption, access controls, and vulnerability assessments safeguard sensitive information and protect against cyber threats.

The Digitized Inventory Management System (IMS) project for Natural Animal Diets is a testament to modern software development practices, innovation in technology, and a user-centric approach to solving industry challenges. By leveraging the power of web frameworks, cloud computing, data analytics, and user experience design, the IMS project delivers a robust, scalable, and efficient solution that meets the evolving needs of the FMCG sector and contributes to enhanced productivity, streamlined operations, and business growth.

6.2 Key Accomplishments of IMS

- 6.2.1 Efficient Inventory Control:** The IMS has significantly improved inventory control by providing real-time visibility into stock levels, enabling accurate tracking of raw materials, finished goods, and inventory movements. This has minimized stockouts, overstocking, and inventory discrepancies, optimizing inventory turnover and reducing carrying costs.
- 6.2.2 Streamlined Production Processes:** By monitoring production cycles from raw material intake to finished goods completion, the IMS has streamlined production processes. It schedules production runs, allocates resources, and tracks progress, ensuring efficient utilization of resources and timely delivery of products.
- 6.2.3 Enhanced Data Management:** The IMS integrates with a MongoDB database, offering a flexible and scalable solution for data storage and management. It stores critical data such as inventory levels, employee information, production schedules, and product recipes, facilitating data-driven decision-making and operational efficiency.
- 6.2.4 Automated Alerts and Notifications:** The IMS generates automated alerts and notifications for inventory thresholds, production deviations, and quality control issues. This proactive approach enables timely corrective actions, minimizes disruptions, and ensures adherence to quality standards.
- 6.2.5 Optimized Workforce Planning:** Employee data, qualifications, roles, and tasks are securely stored and managed within the IMS. This aids in workforce planning, task allocation, and performance tracking, improving productivity, and optimizing labor resources.
- 6.2.6 Scalable and Adaptable Architecture:** The IMS's architecture is scalable and adaptable, allowing it to be configured for other FMCG sub-sectors such as food and beverage or household cleaning products. It can manage perishable ingredients, packaging materials, chemical ingredients, safety data sheets, and production cycles specific to different product categories.
- 6.2.7 Improved Customer Satisfaction:** By preventing stock shortages, ensuring consistent product quality, and optimizing production efficiency, the IMS contributes to improved

customer satisfaction. Customers receive timely and accurate deliveries of FMCG products, leading to enhanced brand reputation and loyalty.

6.2.8 Data-driven Insights: The IMS generates comprehensive reports and analytics on inventory performance, production trends, resource utilization, and financial metrics. These data-driven insights support strategic decision-making, process improvements, and business growth initiatives.

Overall, the IMS has revolutionized inventory management practices for Natural Animal Diets, resulting in operational excellence, cost savings, better resource utilization, and a competitive edge in the FMCG industry.

6.3 Future Scope

The future scope of the Digitized Inventory Management System (IMS) extends beyond its current implementation for Natural Animal Diets to encompass a broader vision of serving multiple companies in the Fast-Moving Consumer Goods (FMCG) sector and beyond. Here are key aspects of the IMS project's future scope:

6.3.1 Expansion to Other Companies: The IMS project is designed with scalability in mind, allowing it to be tailored and implemented for other companies operating in the FMCG industry. This includes companies involved in food and beverage production, household cleaning products, personal care items, and more. The system's adaptability enables customization to suit specific industry requirements and business processes.

6.3.2 Multi-Tenant Architecture: To support the expansion to multiple companies, the IMS can adopt a multi-tenant architecture. This architecture allows for the segregation of data, user access controls, and customization options for each company using the system. It ensures data privacy, security, and personalized experiences for individual companies while sharing underlying system infrastructure and functionalities.

6.3.3 Industry-Specific Modules: As the IMS evolves, industry-specific modules and features can be developed and integrated into the system. For example, modules for managing perishable goods, handling regulatory compliance for food products, tracking environmental impact metrics, or incorporating sustainability practices can be added to cater to diverse industry needs.

- 6.3.4 Integration with Supply Chain Partners:** The IMS can be extended to integrate with supply chain partners, including suppliers, distributors, logistics providers, and retailers. Integration APIs can facilitate seamless data exchange, collaborative forecasting, real-time inventory visibility across the supply chain, and automated order fulfillment processes.
- 6.3.5 Advanced Analytics and AI Capabilities:** Future enhancements to the IMS may include advanced analytics capabilities powered by artificial intelligence (AI) and machine learning (ML). Predictive analytics for demand forecasting, anomaly detection for inventory management, optimization algorithms for production scheduling, and smart alerts based on data patterns can enhance decision-making and operational efficiency.
- 6.3.6 Cloud-Based Solutions:** Leveraging cloud-based solutions can further enhance the scalability, reliability, and accessibility of the IMS. Cloud platforms offer benefits such as elastic computing resources, automatic backups, disaster recovery options, and global accessibility, making the IMS suitable for companies operating on a regional or global scale.
- 6.3.7 Continuous Improvement and Feedback:** The future scope of the IMS project also involves continuous improvement based on user feedback, industry trends, technological advancements, and regulatory changes. Iterative updates, feature enhancements, and user training programs ensure that the IMS remains a cutting-edge solution that meets evolving business requirements and market demands.

By embracing these future-oriented strategies, the IMS project can position itself as a leading inventory management solution provider for a diverse range of companies in the FMCG sector and beyond, driving operational excellence, business growth, and competitive advantage in the dynamic marketplace.

References/Bibliography

- [1] React Documentation [Online]. Available: <https://react.dev/> .
- [2] Flask's Documentation [Online]. Available: <https://flask.palletsprojects.com/en/3.0.x/>
- [3] Flask's Tutorials by Tech With Tim [Online]. Available: https://www.youtube.com/watch?v=mqhxxeeTbu0&list=PLzMcbGfZo4-n4vJJybUVV3Un_NFS5EOgX.
- [4] GeeksForGeeks. How to connect ReactJs with Flask API [Online]. Available: <https://www.geeksforgeeks.org/how-to-connect-reactjs-with-flask-api/>
- [5] Towards Data Science. How To Build & Deploy a React + Flask App [Online]. Available: <https://towardsdatascience.com/build-deploy-a-react-flask-app-47a89a5d17d9>