



CS5002NI Software Engineering

MAIN-SIT 35% Coursework

AY 2024-2025

Credit: 30

Student Name: Riya Basnet

London Met ID: 23049033

College ID: np01cp4a230212

Assignment Due Date: 12th May 2025

Assignment Submission Date: 12th May 2025

Word Count: 4334

I confirm that I understand my coursework needs to be submitted online via MST Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.





Similarity Index Report:






15% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

-  **52 Not Cited or Quoted 12%**
Matches with neither in-text citation nor quotation marks
-  **8 Missing Quotations 3%**
Matches that are still very similar to source material
-  **0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 6%  Internet sources
- 0%  Publications
- 14%  Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Table of Contents

1	Introduction	1
1.1	Aim.....	2
1.2	Objectives	2
2	Planning Diagrams.....	3
2.1	WBS:.....	3
3	Planning Diagrams.....	4
3.1	WBS:.....	4
3.2	Gantt Chart:.....	5
4	OOAD Analysis	9
4.1	Use Case Diagram:.....	9
4.2	High-Level Use Case Description	11
4.3	Expanded Use Case Descriptions.....	13
4.3.1	Use Case: Register User	13
4.3.2	Use Case: Make Payment	14
5	Collaboration / Communication Diagram.....	15
6	Activity Diagram	17
6.1	Swimlane Roles and Diagram Component	18
7	Class Diagram.....	19
8	Further Development	22
8.1	Architectural Choice	23
8.2	Design Pattern	24
8.3	Development Plan	25
8.4	Testing Plan.....	26
8.5	Maintenance and Deployment.....	27
9	Prototype.....	28
9.1	Admin Dashboard:	28
9.2	Purchase Orders:	28
9.3	Create Purchase Order UI:	29
9.4	Add Product UI:.....	29
9.5	Home Page (Landing Page).....	30
9.6	Log In Page.....	30
9.7	Sign Up Page	31
9.8	Sales Report:	31
9.9	Purchase Report	32

9.10	Make Payment:	33
9.11	Dispatch Order.....	35
9.12	Compare Products	35
9.13	Transaction Report	36
9.14	Detailed Transaction View.....	36
9.15	User Profile Page.....	37
9.16	Contact Us Page.....	38
10	Conclusion.....	39
11	References.....	40

Table of Figures

Figure 1 WBS Diagram	4
Figure 2: Simplified View of Gantt Chart	5
Figure 3 Gantt Chart detailed subtasks- Initiation	6
Figure 4 Gantt Chart detailed subtasks- Planning	6
Figure 5 Gantt Chart detailed subtasks- Execution - Initial Prototyping (Iterative Ongoing Process).....	6
Figure 6 Gantt Chart detailed subtasks- Execution- Customer Evaluation & Prototype Refinement (Iterative Ongoing Process)	7
Figure 7 Gantt Chart detailed subtasks- Execution – 2 nd Customer Evaluation(Iterative Ongoing Process) & Development	7
Figure 8 Gantt Chart detailed subtasks- Monitoring and Control.....	8
Figure 9 Gantt Chart detailed subtasks- Closure	8
Figure 10 OOAD Analysis- Use Case Diagram	9
Figure 11 Collaboration / Communication Diagram: MakePayment	15
Figure 12 Activity Diagram: Register User.....	17
Figure 13 Class Diagram.....	20
Figure 14 Admin Dashboard.....	28
Figure 15 Purchase Order (Admin)	28
Figure 16 Create Purchase Order (Admin to Supplier).....	29
Figure 17 Add Product (Admin to Inventory)	29
Figure 18 Landing Page (User)	30
Figure 19 Login Page (User)	30
Figure 20 Sign Up page(User)	31
Figure 21 Sales Report (Admin)- 1.....	31
Figure 22 Sales Report (Admin)- 2.....	32
Figure 23 Purchase Report (Admin)- 1.....	32
Figure 24 Purchase Report (Admin)- 2.....	33
Figure 25 Make Payment 1	33
Figure 26 Make Payment 2	34
Figure 27 Dispatch Order Details	35
Figure 28 Compare Products	35

Figure 29 Transaction Report.....	36
Figure 30 Detailed Transaction Report(Individual page)	36
Figure 31 Contact Us Page	38

Table of Tables

Table 1 Use Case: Register User	13
Table 2 Use Case: Make Payment	14
Table 3 Domain Classes based on Use Cases	19

1 Introduction

The Global Tech Corporation has undertaken an initiative to develop an automated Inventory Management System(IMS) to optimize warehouse operations. However, due to the lack of proper object-oriented analysis and Design (OOAD), the project failed, resulting in inefficiencies, order processing delays, financial losses, and customer dissatisfaction.

To address these shortcomings, this coursework focuses on using OOAD principles to redesign the IMS, ensuring a more structured, efficient, maintainable, and scalable system. The project involves the implementation of key functionalities such as user access management, purchase order creation, real-time stock updates, order dispatch, report generation, and secure payment processing.

OOAD is a software engineering methodology that designs systems using objects while defining their relationships within the system. This approach promotes modular, reusable, and maintainable system architecture, enabling better system organization and functionality. By breaking down the system's behavioral aspects and object interactions, OOAD ensures a more effective and scalable software design. This report follows a structured approach of the designing process, including the Work Breakdown Structure (WBS), Gantt Chart, Use Case Diagrams, Sequence Diagrams, Activity Diagrams, Class Diagrams, and system prototypes.

1.1 Aim

To design an improved, object-oriented Inventory Management System that streamlines warehouse operations and overcomes the shortcomings of the previously failed system.

1.2 Objectives

- To analyze the existing problems in the failed system using OOAD principles.
- To identify and define system functionalities.
- To model system behavior using Use Case, Activity, Sequence, and Class Diagrams.
- To develop a layered architecture for the system using appropriate design patterns.
- To build a functional prototype demonstrating key modules.
- To plan for full system implementation using Agile methodology for future development.

2 Planning Diagrams

2.1 WBS:

A Work Breakdown Structure(WBS) breaks the project into tasks and subtasks. The previous Inventory Management System failed due to poor system design and analysis. Since the requirements were unclear and the system needed improvements, the Prototype Model is the best choice. The Prototype Model is a software development approach where a working model or prototype is built, tested, and refined based on user feedback until the user is satisfied before fully developing and deploying the system. (codeacademy, 2023)

The following WBS has been divided into 5 major phases. They are initiation, planning, execution, monitoring and control, and closure. The 5 phases are divided based on the software project management steps, so it follows a very structured approach. Unlike traditional models where planning is completed before development, prototyping occurs within execution because the system is iteratively refined and improved before full-scale development begins. Stakeholders, including end-users and business managers, are involved in reviewing each prototype and providing real-time feedback, ensuring the final system meets their needs. This WBS shows the roadmap of the entire project from initiation to handover whilst also engaging the stakeholders, preventing failures like the previous IMS.

3 Planning Diagrams

3.1 WBS:

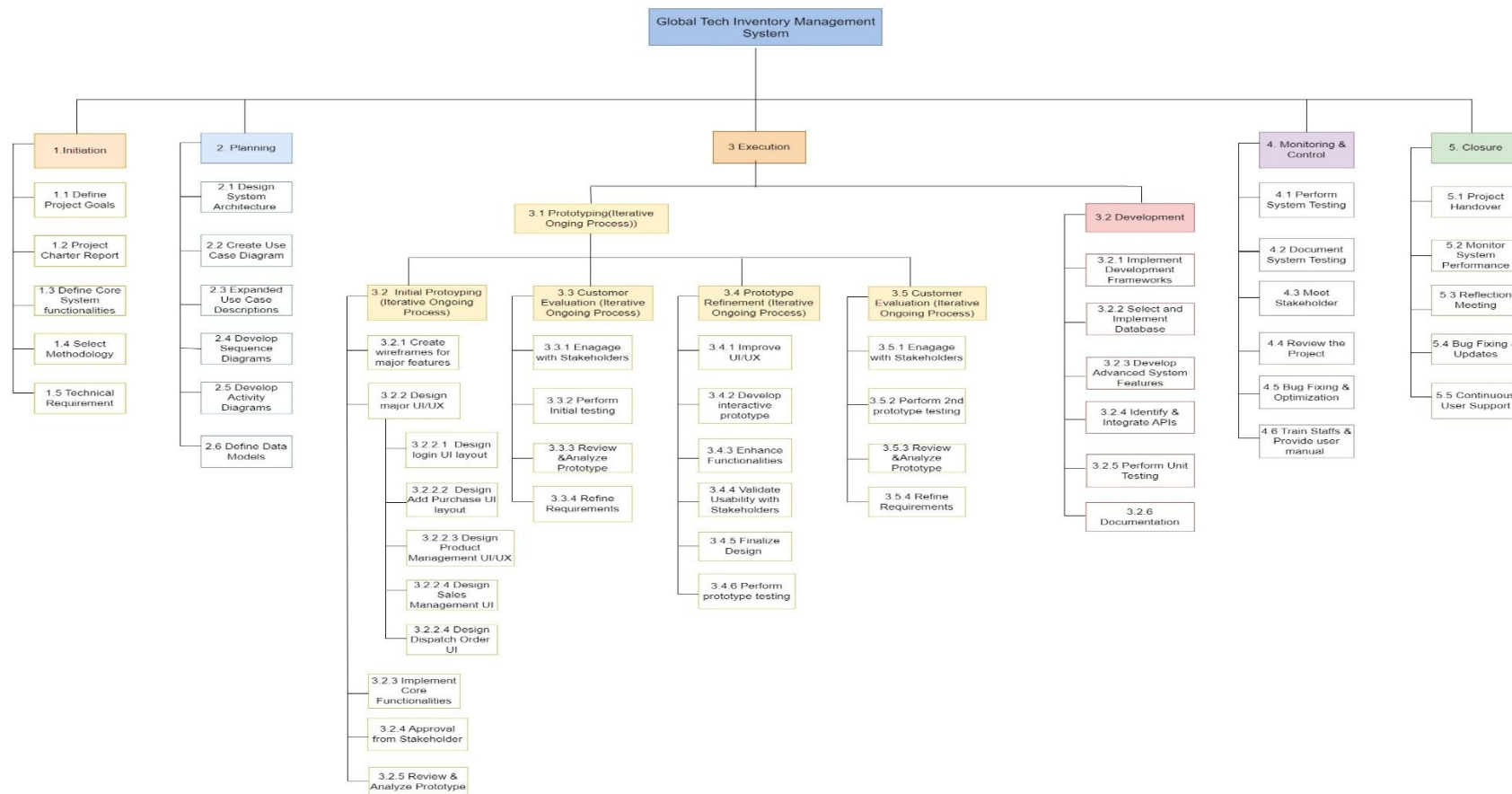


Figure 1 WBS Diagram

3.2 Gantt Chart:

A Gantt chart is a project management tool that visualizes work progress over time against a planned schedule. (Trovato, 2020) It provides a clear and visual way to communicate project plans, dependencies, and progress to stakeholders. The software used to develop the following Gantt Chart is ClickUp.

Since I've followed the Prototype Model and structured WBS into 5 phases based on Software Project Management Principles, the Gantt chart reflects these phases clearly. In the given Gantt Chart, the left side lists all tasks or activities, while the right side shows a horizontal timeline with bars representing each task's duration.

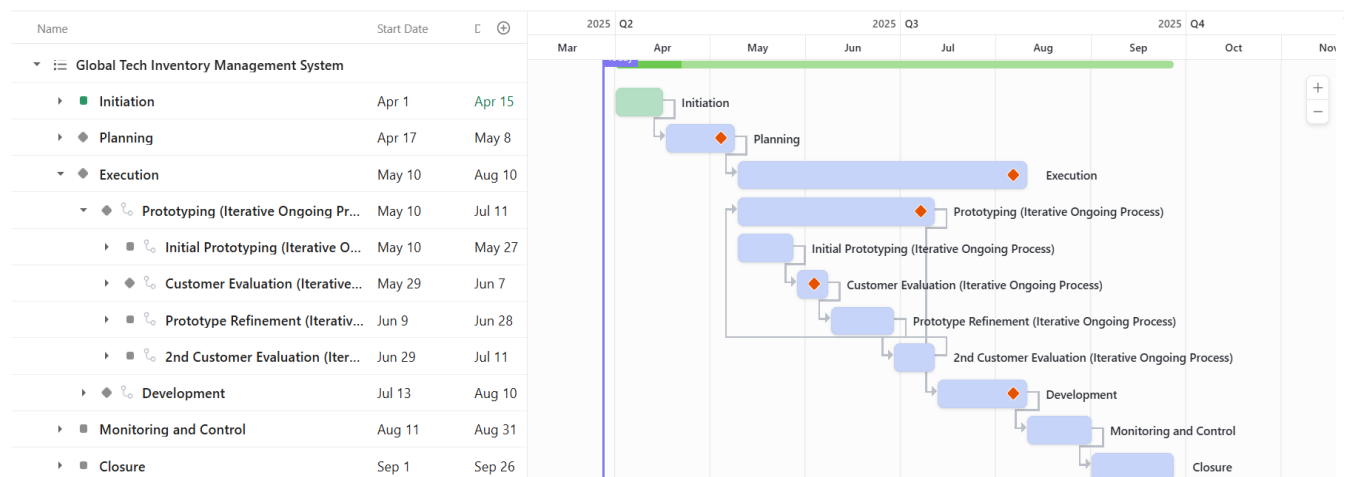


Figure 2: Simplified View of Gantt Chart

The above figure is the simplified view of Gantt Chart. It outlines the project's timeline from the Initial Phase and completes in Closure. The project follows a structured approach, progressing through Planning, Prototyping, and Development phases. The milestones are carefully placed, ensuring everyone remains on the same track. After the completion of each main phase, planning, prototyping, and development, there will be a meeting with the stakeholders and the developers, as indicated by the milestone, a red diamond-shaped symbol. The whole project is set to start from April 1 and end on Sep 26, taking a total of 6 months to complete this inventory management system officially.

The following figures break down the Gantt Chart's different SDLC phases into detailed subtasks. Each phase is carefully structured to ensure a smooth workflow.

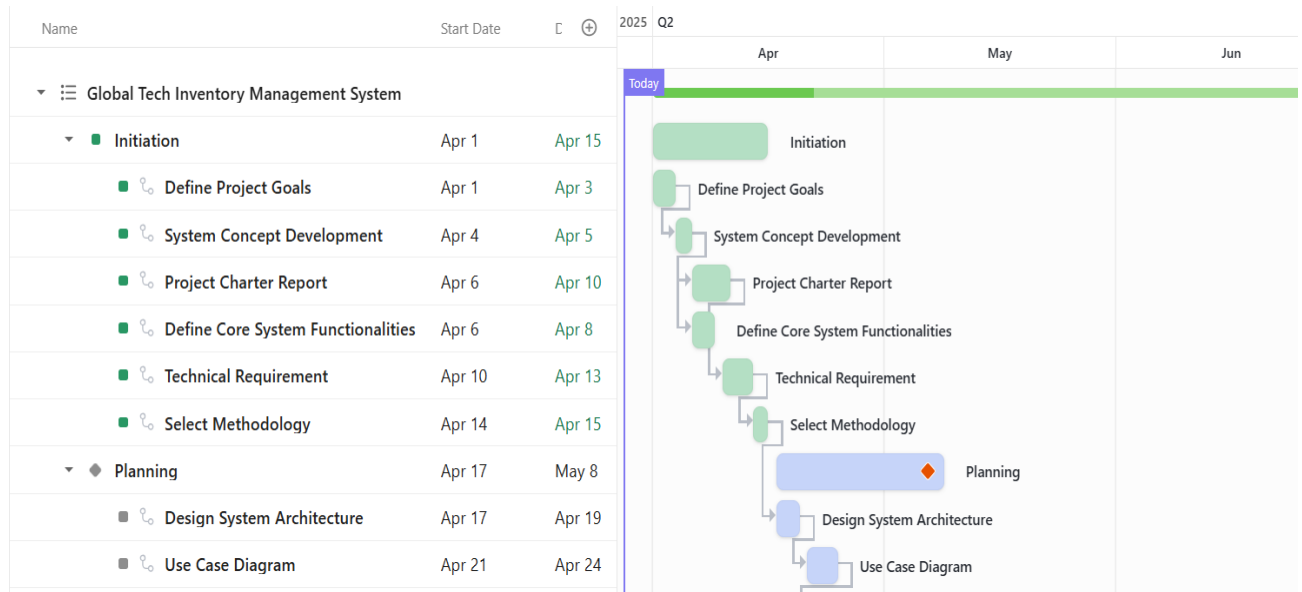


Figure 3 Gantt Chart detailed subtasks- Initiation

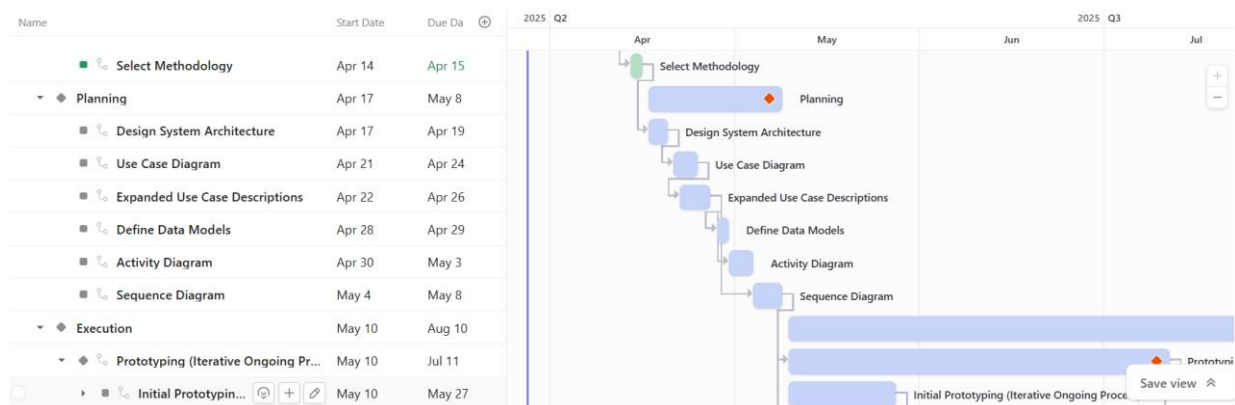


Figure 4 Gantt Chart detailed subtasks- Planning

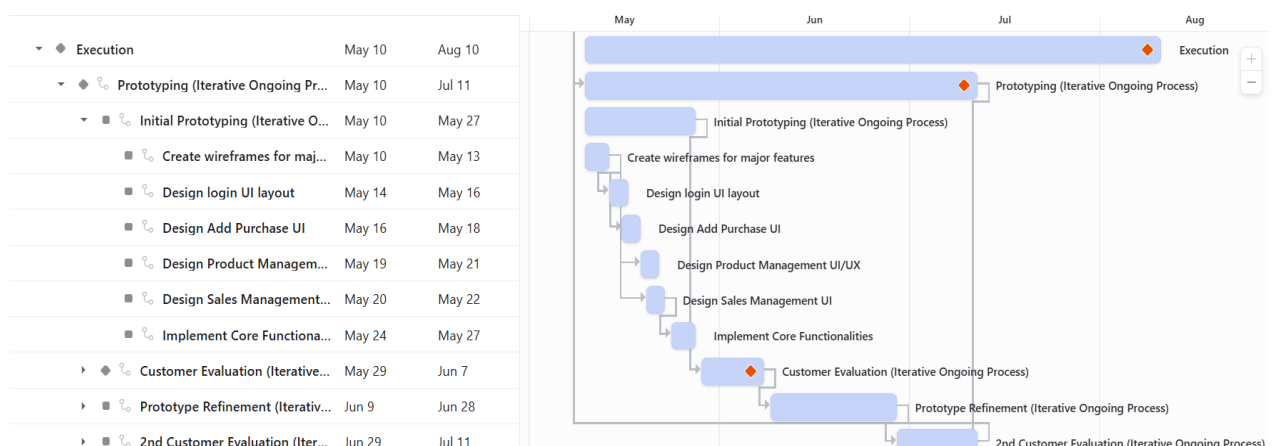


Figure 5 Gantt Chart detailed subtasks- Execution - Initial Prototyping (Iterative Ongoing Process)

In figure 5, the Initial Prototyping dependencies are set in an iterative manner, ensuring the prototyping is an ongoing process till the stakeholders are satisfied with it. The dependencies are also set logically to reflect the ongoing process.

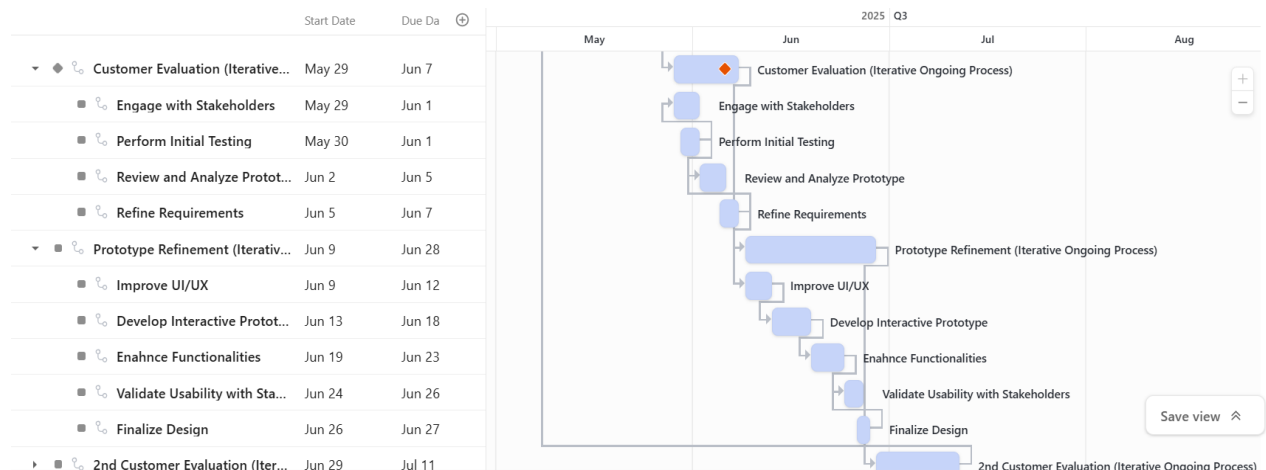


Figure 6 Gantt Chart detailed subtasks- Execution- Customer Evaluation & Prototype Refinement (Iterative Ongoing Process)

From fig 6, the initial testing phase will only end when we have started engaging with the Stakeholders. This represents a start-to-end dependency, meaning that testing results are reviewed by stakeholders before moving forward. Also, the review and analyze prototype will only start after the initial testing is completed. Thus, the dependency on the fig is end-to-start.

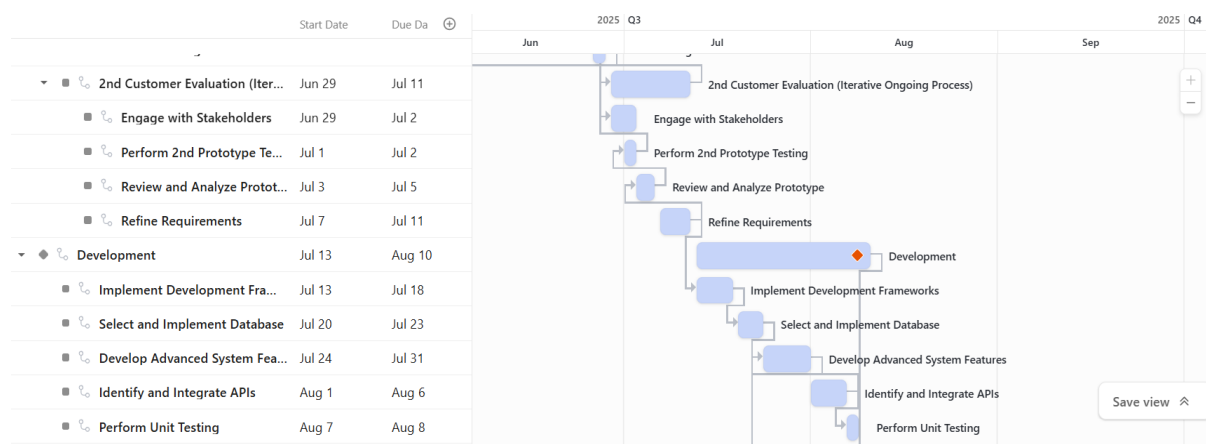


Figure 7 Gantt Chart detailed subtasks- Execution – 2nd Customer Evaluation(Iterative Ongoing Process) & Development

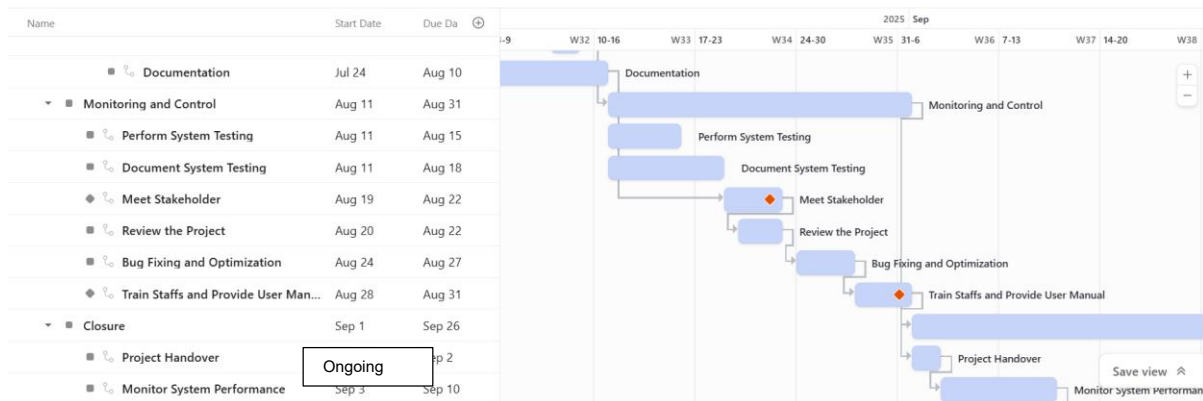


Figure 8 Gantt Chart detailed subtasks- Monitoring and Control

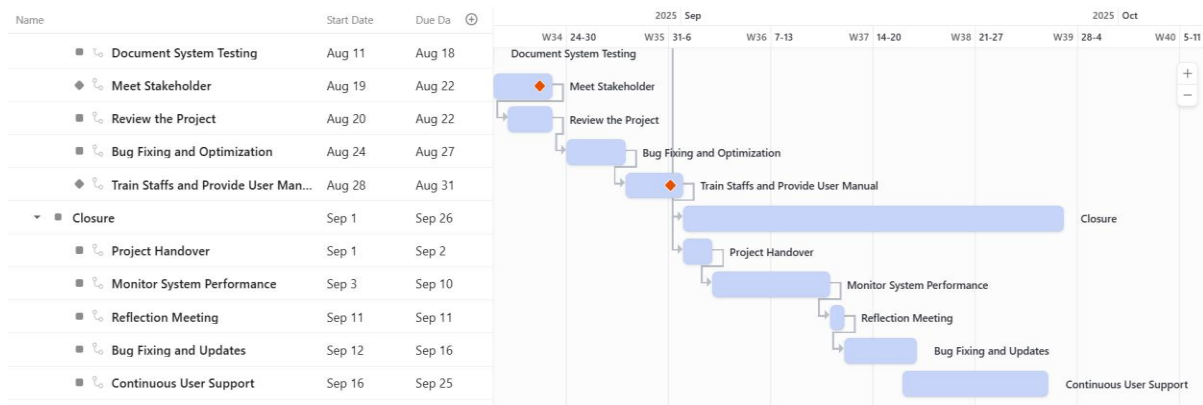


Figure 9 Gantt Chart detailed subtasks- Closure

Figures 8 and 9, the Monitoring and Control, and Closure phases show the steps after successfully developing the system, which involves testing, maintaining, and deploying it to the end users. Milestone is set after training staff and providing a user manual, which indicates a meeting is set before successfully handing over the system. Through testing, review meetings, bug fixes, and training sessions, stakeholders and users are equipped with a well-optimized and documented system. The Closure phase finalizes project execution with a structured handover, post-implementation monitoring, and continuous support to ensure the system operates effectively in real-world scenarios. With continuous user support, the project ends successfully.

4 OOAD Analysis

4.1 Use Case Diagram:

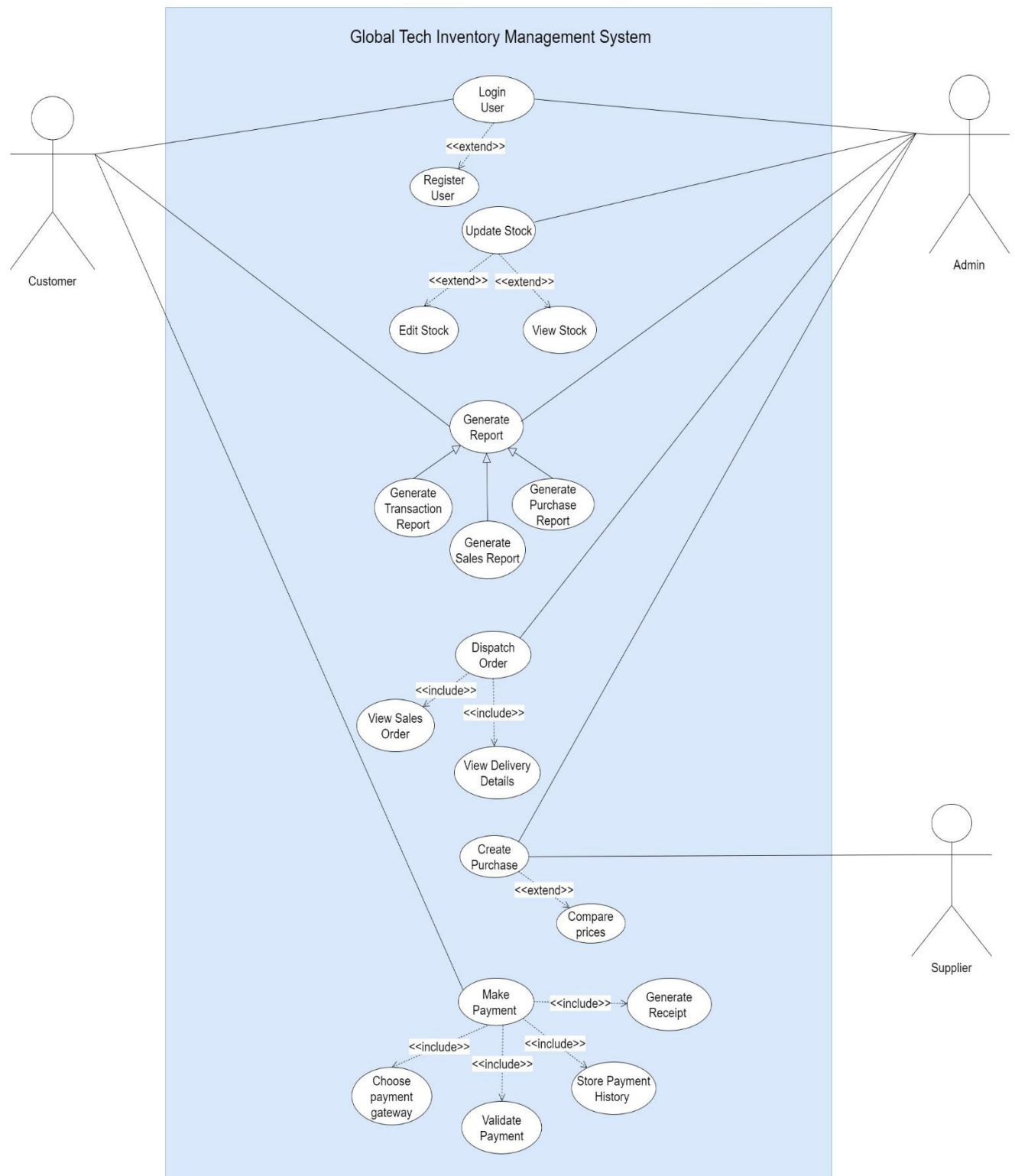


Figure 10 OOAD Analysis- Use Case Diagram

Fig 10, in the figure is a use case diagram for the Global Tech Inventory Management System. The use case describes specific functionalities or tasks that a system provides to its users. The use case consists of 6 major features within the system boundary. In a Use Case Diagram, the naming of use cases follows a verb-noun format to clearly describe the action being performed.

The system begins with 'Login User', where users can register if they haven't registered yet. Thus, forming an extended relation with the 'Login User'. The admin can access the 'Update Stock' where the admin can choose to edit and view stock or both. Since, the system has role-based user access, there is a generalized feature 'Generate Report'. Under the Generate Report are the three types of report being generated, Purchase Report and Sales Report, which can be accessed by the admin, and the other Transaction Report which can be accessed by the customer.

The 'Dispatch Order' and 'Create Purchase' are the use cases that are associated with the admin. The customer can access the 'Make Payment' feature, which includes the customer choosing a payment gateway and then validating the payment. The payment is then stored for the customer to access the payment history. Make Payment also includes generating the receipt after successfully making the payment.

The high-level description of use case and extended level use case description are given below.

4.2 High-Level Use Case Description

1. Use Case: Register User

Actors: New Customer (Initiator), Admin(Initiator)

Description: A new user provides valid personal details such as name, email, and password to register in to the system and create an account.

2. Use Case: Login User

Actors: Customer(Initiator), Admin (Initiator)

Description: The User provides login credentials, which the system verifies and grants them access to system functionalities based on their assigned role.

3. Use Case: Create Purchase Order

Actors: Admin(Initiator), Supplier

Description: The admin can create a purchase order including the details of the product and quantity for the supplier. This use case extends Compare Prices.

4. Use Case: Generate Report

Actors: Admin, Customer

Description: The system generates different types of reports based on user roles:

- **Admin:** Can generate Purchase and Sales reports for decision-making.
- **Customer:** Can generate their Transaction report showing completed payments and order details.

5. Use Case: Dispatch Order

Actors: Admin (Initiator)

Description: Admin manages order dispatch by reviewing payment status and delivery details before initiating shipment to the customer. This use case includes viewing both sales order and delivery details.

6. Use Case: Real-Time Stock Update

Actors: Admin

Description: The system automatically updates stock levels when new purchase orders are made and added to the inventory, and reduces stock when sales happen.

7. Use Case: Make Payment

Actors: Customer(Initiator)

Description: A customer makes the payment for their purchase using a preferred payment method or gateway. If the payment is successful, the system sends a receipt via email/SMS. This use case includes:

- Choose Payment Gateway
- Validate Payment
- Generate Receipt
- Store Payment History

4.3 Expanded Use Case Descriptions

4.3.1 Use Case: Register User

Actors: New User (Initiator)

Description: A new user registers with the system by providing validated personal details and creating an account. If the account is successfully registered, the user receives login credentials.

Typical Course of Events:

Table 1 Use Case: Register User

Actor Action	System Response
1. The user accesses the registration form.	2. Displays a registration form with required fields.
3. The new user provides personal details for registration.	4. The system validates the input fields.
5. Clicks the "Register" button.	6. Checks for duplicate accounts.
	7. Validated details to register and create the account.
	8. Displays a success message and sends login credentials.

Alternative Course:

Line 5: If any input field is missing, system prompts an error message. Use Case ends.

Line 7: If the email is already registered, system notifies the user. Use Case ends.

4.3.2 Use Case: Make Payment

Actors: Customer (Initiator)

Description: A customer makes payment for their purchase using a preferred payment method or gateway. If the payment is successful, the system sends a receipt via email/SMS.

Typical Course of Events:

Table 2 Use Case: Make Payment

Actor Action	System Response
1. The customer proceeds to make payment after finalizing the product.	2. Displays the total amount and available payment gateways.
3. Selects a payment gateway.	4. The system requests payment details.
5. Enter payment details and confirm purchase.	6. Validates and processes the payment.
7. Receives payment confirmation.	8. Sends a receipt via email/SMS.

Alternative Course:

Line 5: If payment details are not valid, prompt an error. the Use Case ends.

Line 6: If the payment process fails, the system allows retrying with another payment gateway. Use Case ends.

5 Collaboration / Communication Diagram

Collaboration Diagrams are used to show how objects interact to perform the behavior of a particular use case. It is used to define and clarify the roles of the objects that perform a particular flow of events of a use case.

From the above Expanded Use Case Description of MakePayment, the following Communication Diagram is produced. The typical course of events of the expanded level descriptions is taken into consideration when producing the communication diagram. The nouns from the Actor and System Actions description are identified as objects. The main name of the UI and control object is the name of the use case we decided to make the diagram for.

In the following figure, the MakePayment is the control object, and the name of the control object followed with UI, MakePaymentUI, is the boundary object. The domain objects here are Order to fetch order details to the UI, PaymentGateway to list the gateways and initiate transaction, PaymentProcess to process the payment and sends the error or confirmation message, Payment to store confirmed payment details and Email/smsService to send the receipt to the users. The customer is the actor on the left of the Boundary object is the initiator who interacts with the Boundary Object.

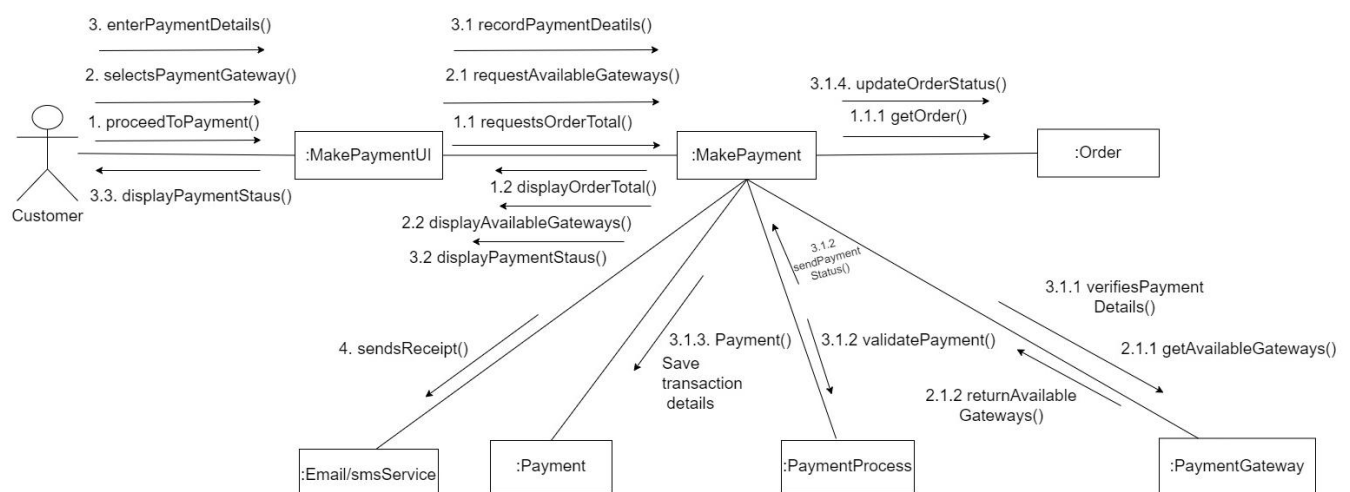


Figure 11 Collaboration / Communication Diagram: MakePayment

Figure 11, depicts the Communication Diagram for the "Make Payment" use case. The process starts with the Customer initiating the payment through the MakePaymentUI, which then relays the request to MakePayment. The control object fetches order details from Order object and displays it to the customer. Then, the customer selects a payment gateway, after PaymentGateway fetches the available options. Upon entering payment details, the transaction moves to PaymentProcess for validation and processing the payment. If successful, the Payment object stores the confirmed transaction, and the Email/smsService sends a receipt. It also updates the order object as payment done for the particular order. The control object then communicates back to the UI, displaying a confirmation message for the customer. The diagram ensures clear interactions and communication between objects of the use case.

6 Activity Diagram

An activity diagram is a type of UML used to model the workflow of a system or process. The activity diagram, given below in fig 12, uses symbols to define the overall workflow of user registration. It is composed of activities, decisions, and paths (flows).

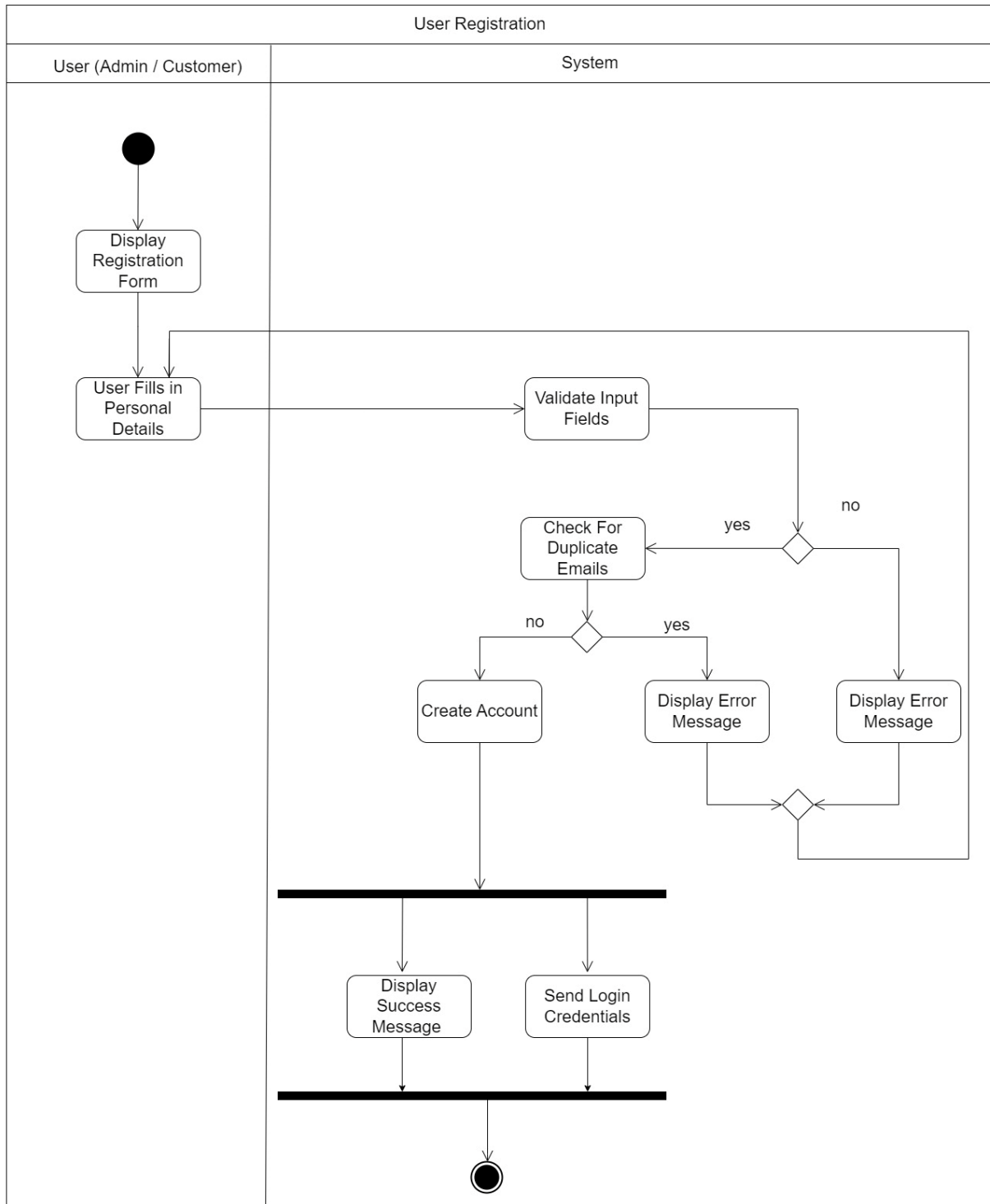


Figure 12 Activity Diagram: Register User

Fig 12 represents the activity diagram of the Register User use case. It starts with an Initial Node, where the user accesses the registration form, and ends at the Final Node. The user fills in personal details, and the system performs input validation. If any required fields are missing, an error message is displayed, and the process loops back to the input form. Similarly, if the email entered already exists in the system, another error message is shown, prompting re-entry. Thus, both the error messages are merged, as the both lead to the iteration of the "input fields" process till the entries are all valid. Upon successful validation, the system proceeds to create the user account. This activity diagram successfully shows dynamic behavior and sequential flow of User Registration Process using UML notations.

6.1 Swimlane Roles and Diagram Component

- **User Swimlane:** Represents activities done by the user (Admin or Customer).
- **System Swimlane:** Represents actions automatically handled by the system.
- **Activities:** Represented using rounded rectangles, showing each step like "Fill Details", "Validate Input", and "Create Account".
- **Decision Nodes:** Represented using diamonds, used to check conditions like field validity and duplicate emails.
- **Merge Node:** Combines error handling paths into a single flow for re-entry of input.
- **Action Flow Arrows:** Show the direction of process flow.
- **Fork Node:** Used to perform parallel tasks – displaying a success message and sending login credentials.
- **Join Node:** Combines both parallel actions before ending the process.
- **Final Node:** Indicates the end of the registration process.

7 Class Diagram

Below is the list of all possible domain classes based on the use cases:

Table 3 Domain Classes based on Use Cases

S.N.	Use Cases	Domain Classes
1.	Register User	User, Customer
2.	Login User	User, Customer, Admin
3.	Create Purchase	PurchaseOrder, Admin, Supplier, Inventory
4.	Generate Report	Report, PurchaseReport, SalesReport, TransactionReport, Admin, Customer
5.	Dispatch Order	Order, DeliveryService, Payment, Customer
6.	Real-Time Stock Update	Inventory, Admin, PurchaseOrder, Order
7.	Make Payment	Payment, PaymentGateway, Receipt, Customer, Order

Class Diagram:

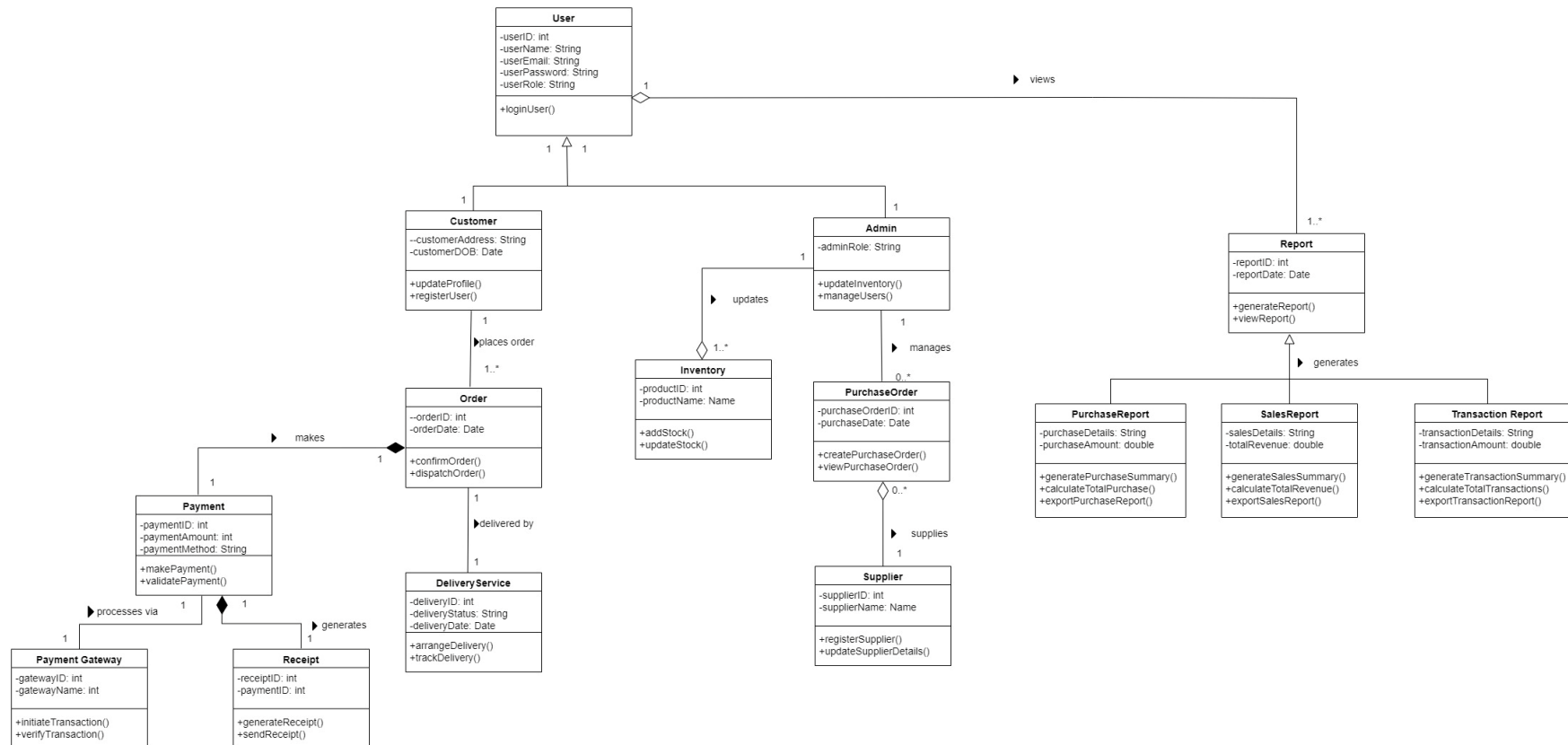


Figure 13 Class Diagram

The class diagram shows the main structural parts of the Inventory Management System. The User class is the main parent class and is divided into Customer and Admin to show role-based access. Customers can place orders and make payments, while Admins manage inventory and generate reports.

The Payment class is connected with both Receipt and PaymentGateway using composition because they are directly part of the payment process and cannot exist without it. PurchaseOrder is linked to Supplier using aggregation to show that the supplier exists separately but is related. Inventory is also connected to Admin through aggregation.

For reporting, a general Report class is used as the parent, and it has three subclasses: PurchaseReport, SalesReport, and TransactionReport. These reports are shown based on roles. Customers can only view their TransactionReport, and Admins can view the SalesReport and PurchaseReport.

The diagram includes all required object-oriented relationships such as inheritance between User and its subclasses, composition between Payment and Receipt, aggregation between Supplier and PurchaseOrder, and associations between classes like Order, Customer, and Inventory.

The Inventory class supports real-time updates. When a PurchaseOrder is created or an Order is dispatched, the stock levels are updated accordingly.

Multiplicity is shown in the diagram. For example, a Customer can place many Orders (1 to many), and each Order is linked with a DeliveryService. This class diagram helps to separate roles clearly, manage dependencies, and supports future updates to the system.

8 Further Development

With the analysis and design phases being completed, the project will progress into the implementation and deployment phases. The prototype model helped visualize and validate the system's initial core features through feedback and iteration. However, for the remaining phases, Agile will allow me to handle changing requirements more effectively, deliver usable software in shorter cycles, and enable continuous improvement through iterative sprints.

The chosen Agile Scrum methodology supports flexibility, faster delivery, and consistent stakeholder involvement. The work will be broken down into manageable tasks and delivered in multiple sprints. Each sprint will aim to deliver a working and testable increment of the system, allowing quicker feedback, better collaboration, and more adaptive planning throughout the development cycle.

8.1 Architectural Choice

The architectural choice of the software defines the basic characteristics of an application while addressing concerns like performance limitation, high availability and minimizing business risk.

The system will be developed using a Layered Architecture (n- Tier Architecture). There are 4 distinct layers in this Architecture, they are:

1. Presentation Layer: UI/UX (User Interfaces)

The presentation layers handle all user interface elements and interaction. It handles user input and displays information (GeeksforGeeks, 2023).

2. Business Logic Layer: Main code and logic (Core Functionalities)

This layer contains the core logic of the application, such as processing user requests and managing business rules (GeeksforGeeks, 2023).

3. Persistence Layer: Bridge between Business Logic layer and Data Access Layer (Controller)

It acts as an intermediary between the business logic and the database, allowing for changes in the database system without affecting the rest of the application (Liferay DXP 7.0 , 2023).

4. Data Access Layer: Database store and retrieval (Database Operation)

Data is retrieved and stored. It interacts with the database, ensuring separation of data concerns (GeeksforGeeks, 2023).

This architecture allows for modular development, easier troubleshooting, and future scalability as the changes made in one layer does not directly affect others reducing development risks and improving code quality.

8.2 Design Pattern

The Model-View-Controller (MVC) design pattern will be implemented throughout the development phase. MVC is one of the most widely used architectural design patterns in modern web application development because it clearly separates concerns and improves maintainability, testing, and scalability.

The MVC pattern divides the application into three interconnected components:

1. **Model:** Model manages the data, logic, and rules of the application. In this project, the Model will represent core entities such as Inventory, Orders, Customers, Payments, and Suppliers, handling all data-related operations and business rules. It ensures that data is cleanly managed, retrieved, and updated without being affected by the UI or user actions.
2. **View:** View handles the presentation layer and is responsible for displaying the data to the user. This includes all user interface components such as the Admin Dashboard, Customer Pages, Product Listings, and Order History views. Views will be designed to be responsive and interactive, ensuring a good user experience.
3. **Controller:** Controller acts as an intermediary between Model and View, handling incoming user inputs (like clicks, forms, requests) and updating the Model or View accordingly. It manages the flow of data and control logic, ensuring that changes in the Model reflect in the View and vice versa.

Justification for Choosing MVC

- **Separation of concerns:** Each layer has a defined responsibility which reduces code duplication and confusion.
- **Scalability:** As the project grows, more features or views can be added without affecting the existing structure.
- **Ease of Testing and Maintenance:** Business logic can be tested independently of UI, and UI changes won't break core logic.
- **Better Collaboration:** Developers working on different parts (logic, UI, DB) can work simultaneously without conflict.

MVC supports incremental and iterative development, allowing functionalities to be developed, tested independently, and integrated progressively, aligning perfectly with Agile Sprint cycles. Different developers can simultaneously work on the View, Controller, and Model without causing system inconsistencies. Thus, MVC was found to be advantageous in our system.

8.3 Development Plan

To successfully implement the system with the chosen architecture and design patterns, the selected tech stack ensures each layer of the layered architecture is properly supported while also aligning with MVC principles.

The technology stack and tools planned to use for further development are:

- **Backend:** Java, JSP (JavaServer Pages), Servlets, and PHP
- **Frontend:** HTML, CSS, JavaScript
- **Database:** MySQL / SQLite, JDBC (Java Database Connectivity)
- **Development Environment:** Eclipse IDE, Apache Tomcat
- **Version Control:** Git and GitHub
- **Project Management:** ClickUp/ TeamGantt
- **Prototyping and Wireframing:** Figma

Priority Order of Features:

Phase 1: User Registration, Login System (Role-based Access Control)

Phase 2: Inventory Management (Product Add/Update/Delete)

Phase 3: Purchase Order and Supplier Management

Phase 4: Customer Order Placement and Dispatch Management

Phase 5: Payment System Integration with Receipt Generation

Phase 6: Report Generation (Sales, Purchase, Transaction)

Phase 7: Admin Dashboard Polishing and UI Enhancements

8.4 Testing Plan

Testing is a critical phase to ensure the system meets functional and non-functional requirements. As the development continues under the Agile methodology, I plan to implement incremental and continuous testing throughout the lifecycle. The various tests to be performed in this system are as follows:

- Unit Testing will be performed for individual modules (Inventory Update, Payment Validation). For example, validating methods like `calculateTotal()`, `updateStock()`, or `validatePayment()` using JUnit.
- Integration Testing after combining modules (e.g., Order Placement + Payment). This helps detect errors in how components communicate.
- System Testing on the fully integrated system will be done before the user demo.
- User Acceptance Testing (UAT) with stakeholders after each iterative delivery, i.e. prototype evaluation session, to check whether the system meets actual user expectations and use cases.
- Regression Testing will be done after updates or bug fixes to ensure that the new changes have not broken any existing functionalities.

Types of Testing Involved:

1. Black Box Testing: Used during System Testing, UAT where testers validate functionalities without knowledge of internal code.
2. White Box Testing: Used during Unit Testing where developers check the internal logic of code modules.
3. Smoke Testing: Quick initial testing after deployments to check major functionalities are working.

Tools for Testing:

1. JUnit – For Java-based unit testing.
2. Manual Testing – For GUI interactions and acceptance testing.
3. Browser DevTools – For inspecting frontend behavior and debugging client-side scripts.

A test case document will be maintained with scenarios, expected outcomes, and status. Bugs will be logged and tracked using a shared document or GitHub issues to ensure a reliable, stable and user-friendly system before final deployment.

8.5 Maintenance and Deployment

Once the system is fully developed and tested, it will be deployed and monitored regularly to ensure stability and performance.

Deployment Tools:

- Document for the system setup, codebase, and user instructions will be made to assist with future maintenance.
- Initial deployment will be done in a staging environment, using XAMPP locally.
- Apache Tomcat Server for running Java web applications.
- After successful testing and validation, the system will move forward with live deployment on a public shared or cloud server.

Database Hosting:

- MySQL will be hosted on a secure server to ensure reliable data access.
- Initial data will be seeded for testing and demonstration purposes.

Maintenance Plan:

System maintenance will include regular updates, bug fixes, and small enhancements based on user feedback.

- Corrective Maintenance:

Fixing issues or bugs found after deployment.

- Adaptive Maintenance:

Modifying the system to accommodate future platform changes or software updates.

- Perfective Maintenance:

Improving system performance, UI/UX, or adding minor feature upgrades after receiving feedback from users.

- Preventive Maintenance:

Optimizing code, updating libraries, and checking for security vulnerabilities to avoid future issues.

9 Prototype

A prototype is a simulation of a final product which design teams use for testing before committing resources to building the actual thing. The goal of a prototype is to test and validate ideas before sharing them with stakeholders and eventually passing the final designs to engineering teams for the development process (UXPin, 2024). The following are the prototypes that serve as the visual representation of the system’s use cases.

9.1 Admin Dashboard:

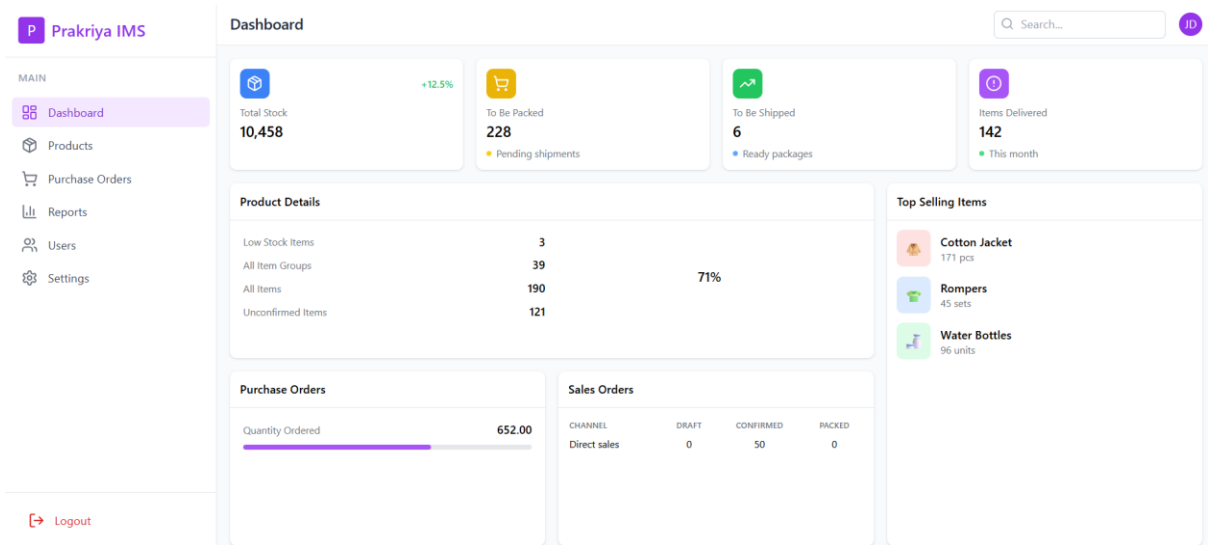


Figure 14 Admin Dashboard

9.2 Purchase Orders:

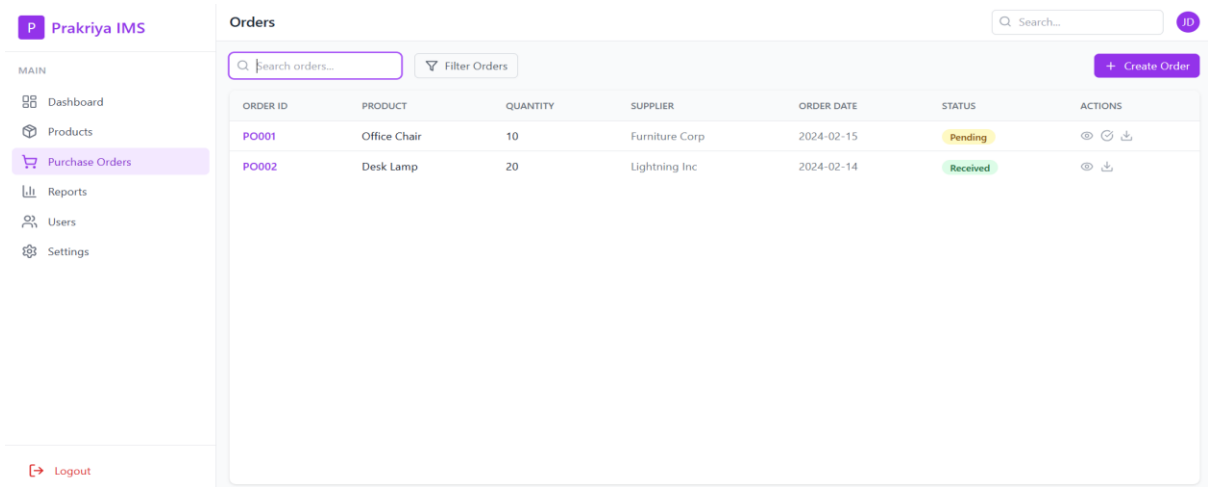


Figure 15 Purchase Order (Admin)

9.3 Create Purchase Order UI:

The screenshot shows the 'Create Purchase Order' modal in the Prakriya IMS application. The modal is titled 'Create Purchase Order' and has a subtitle 'Fill in the order details'. It contains the following fields:

- Product ***: A dropdown menu with the placeholder text 'Select a product'.
- Supplier ***: A dropdown menu with the placeholder text 'Select a supplier'.
- Quantity ***: A text input field with a placeholder '1' and a small up/down arrow icon.
- Expected Delivery ***: A date input field with a placeholder 'mm/dd/yyyy' and a calendar icon.
- Order Notes**: A text area with the placeholder text 'Add any special instructions...'.

Below the fields, there is a yellow note box with a warning icon and the text: 'Note: Orders cannot be modified once created. Delivery date must be at least 2 days from today.' At the bottom right of the modal are two buttons: 'Cancel' and 'Create Order'.

The background shows the 'Orders' section of the application with a table of existing orders. The table has columns for 'ORDER ID', 'STATUS', and 'ACTIONS'. The visible rows are:

ORDER ID	STATUS	ACTIONS
PO001	Pending	View, Refresh, Download
PO002	Received	View, Download
PO003	Ordered	View, Refresh, Download

Figure 16 Create Purchase Order (Admin to Supplier)

9.4 Add Product UI:

The screenshot shows the 'Add Product' modal in the Prakriya IMS application. The modal is titled 'Add Product' and has a subtitle 'Fill in the information below'. It contains the following fields:

- Name ***: A text input field.
- SKU ***: A text input field.
- Category ***: A dropdown menu with 'Furniture' selected.
- Description**: A text area.
- Image**: A button with a camera icon and the text 'Upload'.
- Quantity ***: A text input field.
- Price ***: A text input field with a '\$' symbol.
- Supplier ***: A dropdown menu with 'Furniture Corp' selected.

At the bottom right of the modal are two buttons: 'Cancel' and 'Add Product'.

The background shows the 'Products' section of the application with a table of existing products. The table has columns for 'PRICE', 'STATUS', and 'ACTIONS'. The visible rows are:

PRICE	STATUS	ACTIONS
\$199.99	Low Stock	Edit, Delete
\$49.99	In Stock	Edit, Delete
\$29.99	Out of Stock	Edit, Delete

Figure 17 Add Product (Admin to Inventory)

9.5 Home Page (Landing Page)

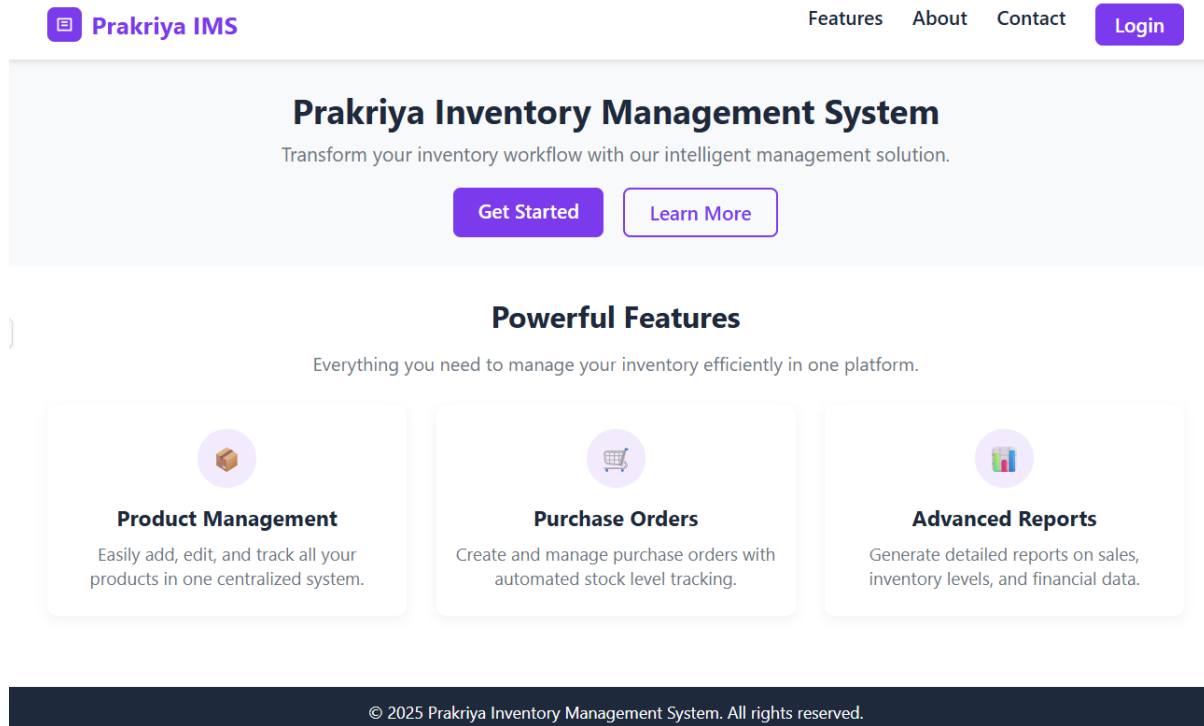


Figure 18 Landing Page (User)

9.6 Log In Page

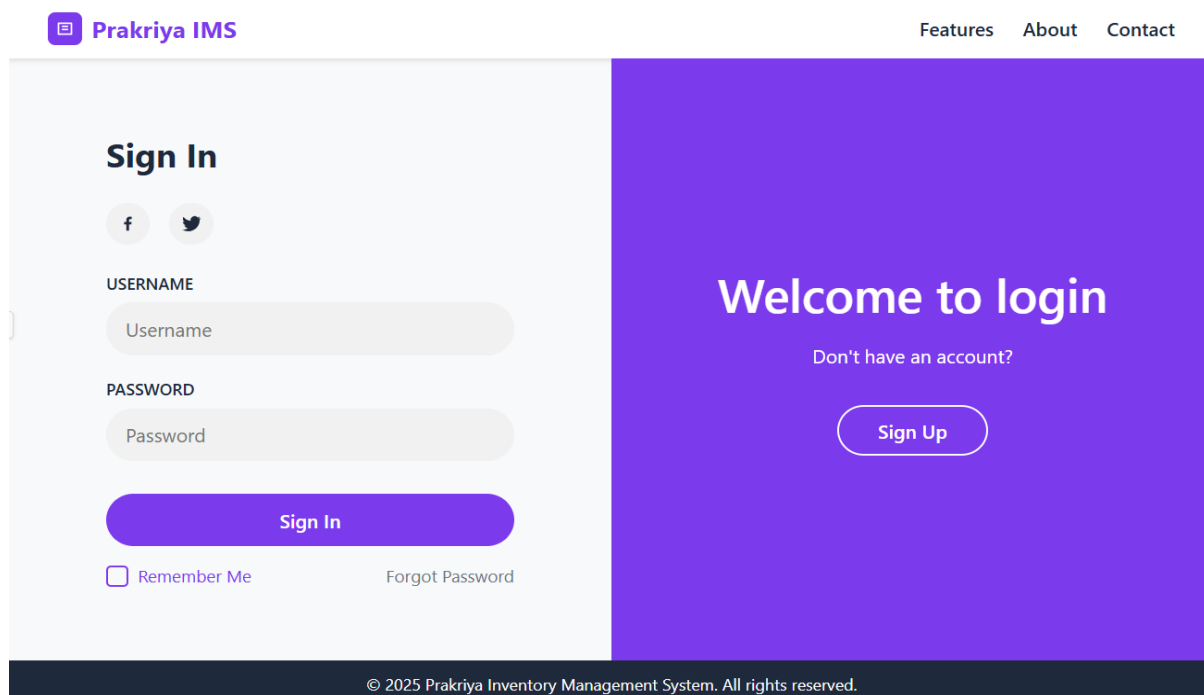


Figure 19 Login Page (User)

9.7 Sign Up Page

Prakriya IMS Features About Contact

Welcome back

Already have an account?

[Sign In](#)

Create Account

f t

USERNAME
Username

EMAIL
your@email.com

ROLE
Select Role

PASSWORD
Password

[Create Account](#)

☐ I agree to the Terms & Conditions

Figure 20 Sign Up page(User)

9.8 Sales Report:

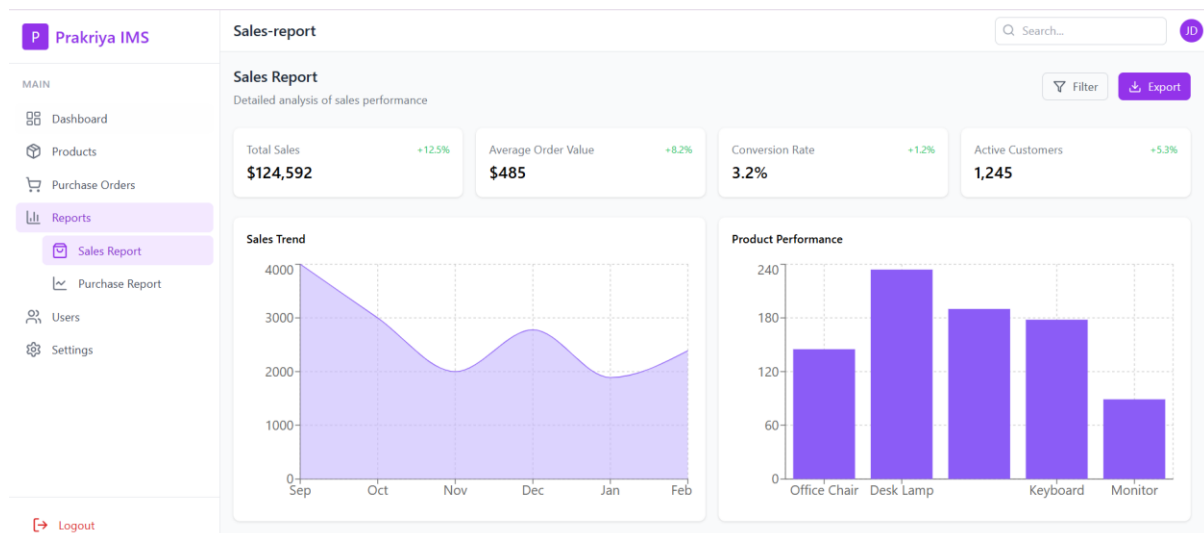


Figure 21 Sales Report (Admin)- 1

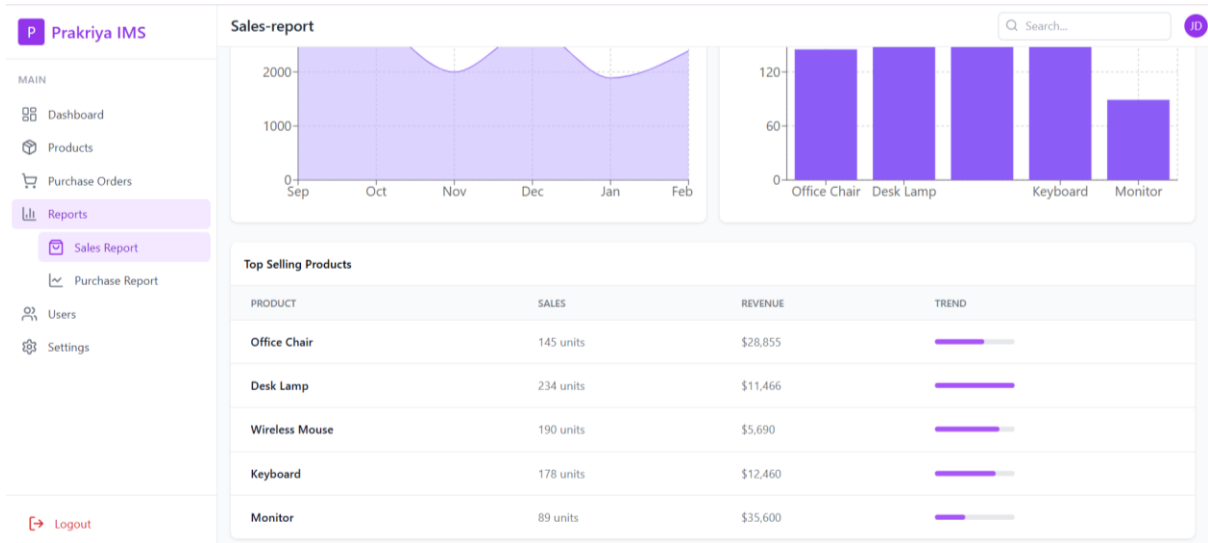


Figure 22 Sales Report (Admin)- 2

9.9 Purchase Report

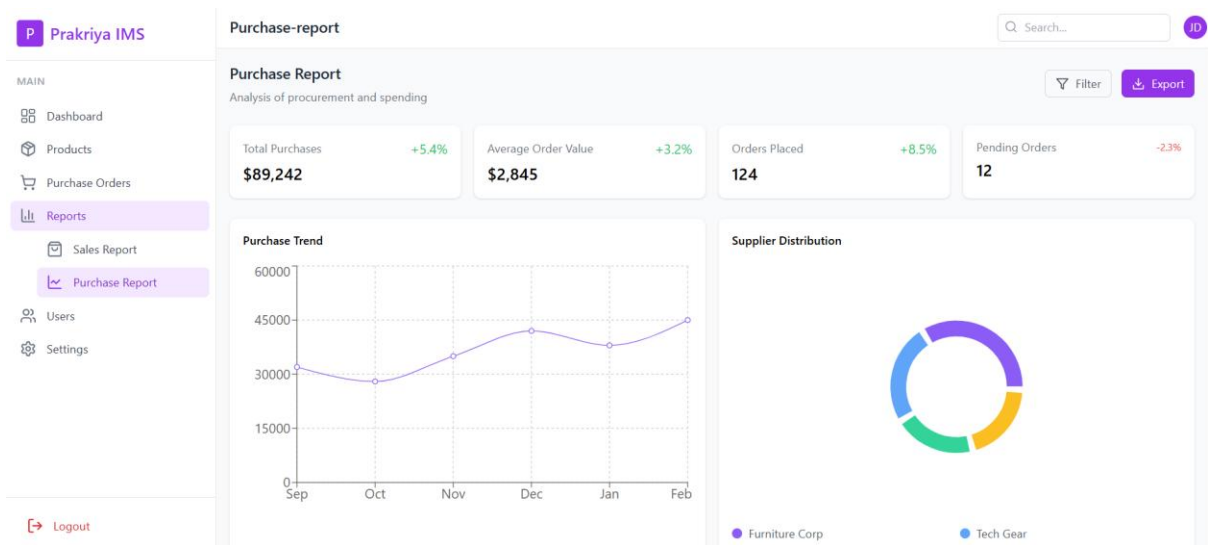


Figure 23 Purchase Report (Admin)- 1

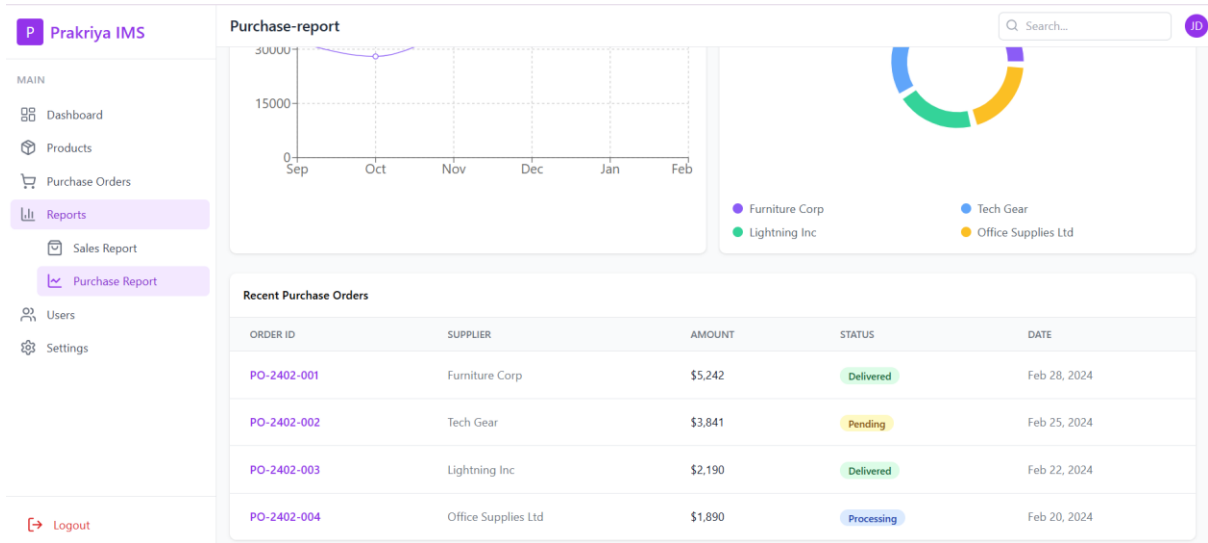


Figure 24 Purchase Report (Admin)- 2

9.10 Make Payment:

← Checkout

Billing Information

First Name Last Name

Email Address


Phone Number


Address


City State ZIP Code

Figure 25 Make Payment 1

Payment Method


Credit Card


Digital Wallet



Pay on Delivery

Card Number

Expiry Date

CVV

Pay \$509.97

 Your payment is secure & encrypted






Figure 26 Make Payment 2

9.11 Dispatch Order

←

Order #A23D4587

Date: 08/02/2023

SHIPPING

✓

Confirmed

📦

Shipping

🚚

To Deliver

Courier

Prakriya Express

Tracking Number

S6A4569AEF4592 [🔗](#)

Items Ordered

🖼️

Office Chair

Qty: 2

\$399.98

🖼️

Desk Lamp

Qty: 1

\$49.99

Total

\$449.97

Figure 27 Dispatch Order Details

9.12 Compare Products

Compare Products ×			
	<div>🖼️</div> <div>Office Chair</div> <div>×</div>	<div>🖼️</div> <div>Gaming Chair</div> <div>×</div>	<div>🖼️</div> <div>Executive Chair</div> <div>×</div>
🏷️ Category	Furniture	Furniture	Furniture
💰 Price	\$199.99	\$299.99	\$399.99
📦 Stock	4 in stock	12 in stock	8 in stock
🚚 Supplier	Furniture Corp	Tech Gear	Office Pro
★ Rating	★ 4.5	★ 4.8	★ 4.2
Description	Ergonomic office chair with lumbar support	Premium gaming chair with adjustable features	Luxurious leather executive chair
	🛒 Add to Cart	🛒 Add to Cart	🛒 Add to Cart

Figure 28 Compare Products

9.13 Transaction Report

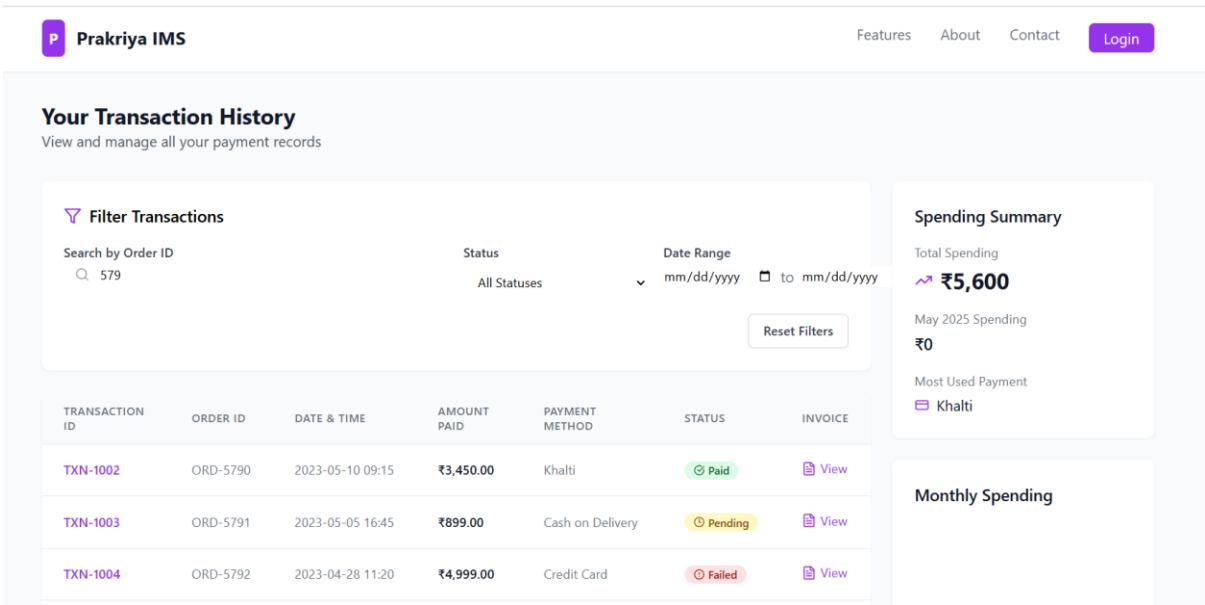


Figure 29 Transaction Report

9.14 Detailed Transaction View

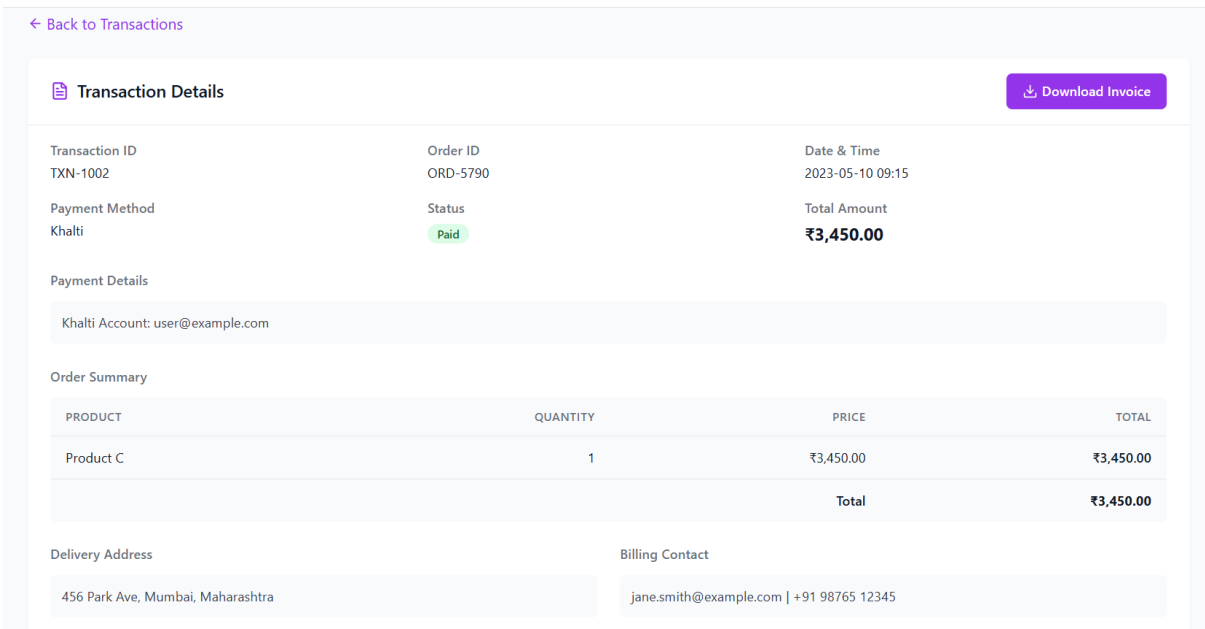



Figure 30 Detailed Transaction Report(Individual page)

9.15 User Profile Page



Riya Basnet

Admin

Joined March 15, 2023

Profile Completion

100%

Personal Information

Email Address

riya.b@example.com

Phone Number

+977 9812345678

Address

Kathmandu, Nepal

Account Information

Username

riya.b

Role

Admin

Last Login

2023-05-20 14:30

Actions

Edit Profile


Change Password

Notifications

☒ Email notifications

☐ Push notifications

☒ SMS notifications

 **Danger Zone**

Delete Account

9.16 Contact Us Page

Contact Us

Have questions? We'd love to hear from you. Send us a message and we'll respond as soon as possible.

Contact Information

Address
123 Business Avenue
Kathmandu, Nepal

Phone
+977 1-4123456

Email
contact@prakriya.com

Office Hours
Monday - Friday: 9:00 AM - 5:00 PM
Saturday: 9:00 AM - 1:00 PM
Sunday: Closed

Follow Us

Send us a Message

Your Name Email Address

Subject

Message

[Send Message](#)

Figure 31 Contact Us Page

10 Conclusion

This report outlines the system planning, analysis, design, and future development approach for the Inventory Management System. It follows a structured flow from identifying use cases to developing behavioral diagrams and finalizing a working prototype. The prototype helped me visualize the system at an early stage and gave me hands-on experience with UI/UX design and functional layout planning.

Through this approach, I aim to deliver a system that meets the client's expectations while also improving my ability to apply real-world software engineering methods in future academic or industry-level projects.

This coursework also helped me gain practical insights into how a system is built from the scratch, starting from detailed analysis and design to planning for implementation and eventual development. It strengthened my skills in technical writing, behavioral diagramming, decision-making, and overall project management, giving me a more confident to develop real software systems as a future developer.

11 References

codecademy, 2023. *General | Software Development Life Cycle | Prototype Model | Codecademy*. [Online]

Available at: <https://www.codecademy.com/resources/docs/general/software-development-life-cycle/prototype-model>

[Accessed 29 3 2025].

GeeksforGeeks, 2023. *Software Architectural Patterns in System Design*. [Online]

Available at: <https://www.geeksforgeeks.org/design-patterns-architecture/>

[Accessed 11 5 2025].

Liferay DXP 7.0 , 2023. *Generating Model, Service, and Persistence Layers*. [Online]

Available at: <https://help.liferay.com/hc/en-us/articles/360017878572-Generating-Model-Service-and-Persistence-Layers#:~:text=The%20persistence%20layer%20saves%20and,calls%20in%20the%20service%20layer.>

[Accessed 11 5 2025].

Microsoft, 2024. *Create a UML activity diagram - Microsoft Support*. [Online]

Available at: <https://support.microsoft.com/en-us/office/create-a-uml-activity-diagram-19745dae-2872-4455-a906-13b736f01685>

[Accessed 31 3 2025].

Nym, 2022. *Activity Diagram for Login and Registration | UML*. [Online]

Available at: <https://itsourcecode.com/uml/activity-diagram-for-login-and-registration/>

[Accessed 31 3 2025].

ProjectManager, 2025. *A Gantt Chart Guide with Definitions & Examples*. [Online]

Available at: <https://www.projectmanager.com/guides/gantt-chart>

[Accessed 29 3 2025].

Trovato, S., 2020. *What is a Gantt chart? Examples + easy to use template [2025]*. [Online]

Available at: <https://monday.com/blog/project-management/everything-you-want-to-know-about-gantt-charts/>

[Accessed 39 3 2025].

UXPin, 2024. *What is a Prototype? A Guide to Functional UX*. [Online]

Available at: <https://www.uxpin.com/studio/blog/what-is-a-prototype-a-guide-to-functional-ux/>

[Accessed 11 5 2025].