**Inventory Management System**

**Deliverable -4**

**Code Crafters**

**Group List**

1. Sameera Ellanki
2. Varsha Jampala
3. Nikhil Kuchipudi
4. Naga Shivani Pinnamaraju
5. Jaswanth Sai Donepudi
6. Manogna Reddy Anugu
7. Atluri Varun Chowdary
8. Naga Bindu Kondaveeti

Index

[1. Requirements specification for phase one 1](#_Toc163499159)

[1.1 3 Order Processing Requirements 1](#_Toc163499160)

[1.2 9 Invoices and bills 1](#_Toc163499161)

[2 UML Diagrams 3](#_Toc163499162)

[2.1 Class Diagram 3](#_Toc163499163)

[2.2 Sequence Diagram 4](#_Toc163499164)

[2.3 Use case diagrams. 5](#_Toc163499165)

[2.3.1 Normal Case 5](#_Toc163499166)

[2.3.2 Error Case 6](#_Toc163499167)

[3 Test Cases 7](#_Toc163499168)

[4 User Manual (Phase 2) 10](#_Toc163499169)

[5 Compilation Instructions 12](#_Toc163499170)

[6 Code Inspection Feedback 13](#_Toc163499171)

[7 Reflection 14](#_Toc163499172)

[8 Member Contribution Table 15](#_Toc163499173)

# Requirements specification for phase one

## 3 Order Processing Requirements

* **3.1 Sales Order Processing**

The user shall have the ability to Manage the sales order lifecycle from creation to fulfilment, we will take the order as and take it through different stages of fulfilment. Such as pending and completed at for now. We can increase the stages as we need. Each Sales order can contain multiple items that are assigned in the name of a single customer.

* **3.2 Purchase Order Management**

The user should be allowed to handle the purchase orders, including creation, approval, and tracking, the same as the sales order we will take it through the series of fulfilment. Such as pending and completed at for now. We can increase the stages as we need. Each Purchase order can contain multiple items that are assigned in the name of a single vendor.

* **3.3 Returns Management**

The system shall process of returns from customers and to vendors, we will increase the stock and delete the corresponding sales order or purchase order.

The system will manage both purchase orders and sales orders return in a way such that by editing the sales order or purchase order we can remove items or edit the quantity if the quantity is adjusted the stock will adjust automatically in the background.

## 9 Invoices and bills

* **9.1 Bills**

The user should be able to generate bills by using the bills module. It should have the access to the vendors and items data since every bill is assigned to a vendor. Bill are the amount that we pay someone else in our case we will select a vendor and a purchase order that are related and the system will automatically calculate how much is the total amount that we need to pay to the vendor.

* **9.2 Invoices**

The user should be able to generate invoices by using the invoice module. It should have the access to the customers and items data since every invoice is assigned to a customer. Invoices are the amount that we get from the customer in this case we will select a customer and a sales order and the system will automatically calculates the amount that we get from customers.

# UML Diagrams

## Class Diagram

The Phase 2 class diagram of the "Inventory Management System" lists attributes below the rectangles that represent the objects, with class names placed in the upper section. Links are used to show relationships between various objects, and lines are used to indicate the connections between these objects.

A diagram of a company

Description automatically generated

## Sequence Diagram

Sequence diagram for the "Inventory management system" in relation to the requirements for Phase 2. Objects are the entities that converse within the system. Vertical lines are used in the sequence diagram to represent them.

A diagram of a diagram

Description automatically generated**Sales Order Creation sequence diagram**

## Use case diagrams.

### Normal Case

This is the normal case scenario for a user to use inventory management system.

Users can perform various actions such as managing sales orders, purchase orders, returns, generating invoices, generating bills, managing items, customers, and vendors.

A diagram of a diagram

Description automatically generated

### Error Case

* Users may attempt to delete certain things.
* For example system checks if bills are raised on the purchase order.
* If bills are raised, an error message is displayed indicating that the purchase order cannot be deleted due to existing bills.

A diagram of a customer order

Description automatically generated

# Test Cases

There is no explicit way to execute test cases for now we must do the test cases manually. You can follow the following things to do the actions for input and output results. **NOTE: Few sections might not be able to delete because thy are used elsewhere.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Description** | **Input** | **Output** | **Test Result** |
| Check if the signup works | Name: name  Email: [email@email.com](mailto:email@email.com)  Password: 12345678  Org Name: Org | Navigates to login page | pass |
| Check if the signup works, if given a same email adress | Name: name  Email: [email@email.com](mailto:email@email.com)  Password: 12345678  Org Name: Org | Shows error | pass |
| Check if the Login works | Email: [email@email.com](mailto:email@email.com)  Password: 12345678 | Navigates to dashboard | pass |
| Check if the Login works if given wrong password | Email: [email@email.com](mailto:email@email.com)  Password: 123456 | Nothing happens | pass |
| Item add | Navigate to items and click on add new item, fill the form and create. | Item creates | pass |
| Item update | Click on the three dots on an item and select the edit icon, update the form and submit | Item updates | pass |
| Item delete | Click on the three dots on an item and select the button. | Item delete | pass |
| Customers add | Navigate to Customers and click on add new Customer, fill the form and create. | Customers creates | pass |
| Customers update | Click on the three dots on an Customers and select the edit icon, update the form and submit | Customers updates | pass |
| Customers delete | Click on the three dots on an Customer and select the button. | Customers delete | pass |
| Vendors add | Navigate to Vendors and click on add new Vendor, fill the form and create. | Vendors creates | pass |
| Vendors update | Click on the three dots on an Vendor and select the edit icon, update the form and submit | Vendors updates | pass |
| Vendors delete | Click on the three dots on an Vendor and select the button. | Vendors delete | pass |
| Create a Sales Order | Click on create sales order button and fill the form. | SO should be created and stock of the selected items should be reduced. | pass |
| Edit a Sales Order | Click on three dots select Edit change the records as needed. (some are restricted from editing) | SO should be edited accordingly. Stock is adjusted accordingly | pass |
| View a Sales Order | Click on 3 dots select the view button. | SO opens with all disabled columns. | pass |
| Delete a Sales Order | Click on 3 dots and select delete. | SO should be deleted and stock is put pack. | pass |
| Create a Purchase Order | Click on create purchase order button and fill the form. | PO should be created and stock of the selected items should be updated. | Pass |
| Edit a Purchase Order | Click on three dots, select Edit, and change the records as needed. (some are restricted from editing) | PO should be edited accordingly. Stock is adjusted accordingly. | Pass |
| View a Purchase Order | Click on three dots, select the view button. | PO opens with all disabled columns. | Pass |
| Delete a Purchase Order | Click on three dots and select delete. | PO should be deleted and stock is updated accordingly. | Pass |
| Create Invoice | Navigate to Invoice module click on create Invoice and select a customer and a sales order and hit create. | A invoice with calculated amount due should be displayed. | Pass |
| Update Invoice | If the corresponding SO is updated we can update the invoice by selecting 3 dots update button. | Invoice amount due should be updated | Pass |
| Delete Inoice | Select 3 dots hit delete button | Invoice should be deleted. | Pass |
| Create a Bill | Navigate to the Bills module, click on create Bill, select a vendor and a purchase order, then hit create. | A bill with calculated amount due should be displayed. | Pass |
| Update Bill | If the corresponding purchase order is updated, select the bill, then click on three dots and select the update button. | Bill amount due should be updated accordingly. | Pass |
| Delete Bill | Select the bill, then click on three dots and hit delete. | The bill should be deleted. | Pass |

# User Manual (Phase 2)

**System Access:**

Requirements: Internet connection, modern web browser.

URL: Access the system by entering the provided one of the URL after running the UI repository.

**Create Account:**

Click “Sign Up” on the login page, enter your details, and submit.

Make sure the email wasn’t used before.

**Logging In:**

Login: Enter your email and password on the login page and click “Login.”

**Managing Items:**

Add: Go to “Items” > “Add New Item,” fill in details, and save.

Update/Delete: Find the item under Items action by clicking on the three dots, select “Edit” to update or “Delete” to remove.

**Managing Vendors:**

Add: Navigate to “Vendors” > “Add New Vendor,” enter details, and save.

Update/Delete: Choose a vendor action by clicking on the three dots, click “Edit” to update or “Delete” to remove.

**Managing Customers:**

Add: Go to “Customers” > “Add New Customer,” fill out the form, and click “Save.”

Update/Delete: Select a customer action by clicking on the three dots, “Edit” to update their information, or “Delete” to remove them.

**Managing Sales Orders:**

Create: Click on the “Create Sales Order” button, fill out the form with the necessary details, and submit to create a new sales order. This action will reduce the stock of selected items.

Edit: Select the sales order you wish to edit, click on the three dots, and choose “Edit” to make changes to the sales order records. Note that some fields may be restricted from editing.

View: Click on the three dots of a sales order and select “View” to open the sales order with all columns disabled for viewing purposes.

Delete: To delete a sales order, click on the three dots of the respective sales order and select “Delete.” This action will put the stock back to its previous state.

**Managing Purchase Orders:**

Create: Navigate to the Purchase Orders module, click on “Create Purchase Order,” fill out the form with the necessary details, and submit to create a new purchase order. This action will update the stock of selected items accordingly.

Edit: Select the purchase order you wish to edit, click on the three dots, choose “Edit,” and make the required changes. Stock adjustments will be made accordingly.

View: Click on the three dots of a purchase order and select “View” to open the purchase order with all columns disabled for viewing.

Delete: To delete a purchase order, click on the three dots of the respective purchase order and select “Delete.” This action will update the stock accordingly.

**Managing Invoices:**

Create: Navigate to the Invoice module, click on “Create Invoice,” select a customer and a sales order, then click “Create” to generate an invoice with the calculated amount due.

Update: If the corresponding sales order is updated, select the invoice, click on the three dots, and choose “Update” to reflect the changes in the invoice amount due.

Delete: To delete an invoice, select the invoice, click on the three dots, and hit “Delete.” This action will remove the invoice from the system.

**Managing Bills:**

Create: Access the Bills module, click on “Create Bill,” select a vendor and a purchase order, then click “Create” to generate a bill with the calculated amount due.

Update: If the corresponding purchase order is updated, select the bill, click on the three dots, and select “Update” to ensure the bill amount due is updated accordingly.

Delete: To delete a bill, select the bill, click on the three dots, and hit “Delete.” This action will remove the bill from the system.

**Logging Out:**

Logout: Click the red button on the side panel called Logout.

# Compilation Instructions

**Deploying the server**

* Navigate to the server files.
* Install Node Package Manager (npm is bundled with node) by installing node. (<https://nodejs.org/en/download>)
* Run the command “npm i”.
* Next Run “npm start”.

**Deploying the UI**

* Open a new terminal but don’t close the server one.
* Navigate to the UI files.
* Install yarn using “npm install --global yarn”.
* Run the command “yarn”.
* Next run “yarn dev”.
* Open one of the links that pops up in the terminal.

The test cases should be tested manually as per the given instructions above in the test cases section.

This can be done only after running the application successfully.

# Code Inspection Feedback

**Suggested to implement Error feedback:**

Previous we didn’t have any errors showing up when we login if it was unsuccessful.

Now we have added an error notification across the app wherever necessary and updated how the errors are picked up so we can now get 2 separate errors for if login failed due to invalid credentials and one for not user found.

**Suggested to add module name on top of each corresponding page**

Previously it was shows as a blank with no heading, now that we have added the heading it is a much better improvement.

**Suggested to add a sort feature**

The items can be sorted in various forms now hover over the table heading if you see a pointer it is a sortable column.

Not only the items few other eligible modules also now can be sorted.

**Suggested to add a search feature**

The inventory management system now supports a search feature on every module if the module has a name it will search the name if the module does not have a name it will search with the id.

# Reflection

During Phase 2 of our project on developing an Inventory Management System, our team made significant progress in implementing key features such as purchase and sales order management, bills, and invoices. This phase demanded a collaborative effort from all team members, each contributing their skills to different aspects of the project.

One of the most notable achievements was the successful implementation of various modules, including Sales Order Processing, Purchase Order Management, Returns Management, Invoices, and Bills. These modules were designed to streamline the process of handling orders, payments, and inventory tracking, ultimately enhancing the efficiency of inventory management for businesses.

In terms of technical implementation, our team focused on building APIs, developing UI components, and ensuring smooth interaction between different system modules. This involved writing clean and efficient code, as well as conducting thorough testing to identify and address any potential issues.

Additionally, we incorporated user feedback and suggestions to enhance the user experience. For instance, we implemented error feedback notifications to provide users with clearer guidance in case of login failures or other errors. We also added sorting and search features to improve data organization and accessibility within the system.

# Member Contribution Table

|  |  |  |
| --- | --- | --- |
| **Member Name** | **Contribution description** | **Overall Contribution (%)** |
| Sameera | Worked on sales orders API, repository maintenance and cleanup of code. | 12.5 |
| Varsha | Worked on Invoices API, contributed to documentation. | 12.5 |
| Nikhil | Worked on purchase orders API, Drawn the class diagram. | 12.5 |
| Jaswanth | Worked on Bills API, written test cases. | 12.5 |
| Varun | Worked on sales orders UI, updated manual | 12.5 |
| Shivani | Worked on Invoices UI, Drawn the sequence diagram and use case diagrams. | 12.5 |
| Bindu | Worked on purchase orders UI, wrote reflection and code inspection feedback. | 12.5 |
| Manogna | Worked on Bills UI, wrote meeting minutes. | 12.5 |