SAP Retail Management System

Project Report

Sri Lanka Institute of Information Technology

IT2080 Information Technology Project

Group ITP2023_S2_T20/ Y2.S2.WD.IT.02.01

Student Name with initials	IT Number	Name of the System/Functions
Ahamed M.S.A	IT20012892	Inventory Management System
Mohamed Z.M.N	IT21360114	Human Resource Management System
Weerasekara D.D.R.R	IT21193804	Supply Chain Management System
Shehan H.A	IT21177682	Marketing and sales Management System
Ransika M.R.T	IT21177514	Customer Relationship Management System
Bandara J.M.O.N	IT20655648	Asset and Expense Tracking System
Abeykoon A.M.Y.V.B	IT21197246	Accounting and financial Reporting System
Sindujan.P	IT21158568	Point of sale System

Feb-June 2023

Appendix 2 – Declaration

Declaration

This project report is our original work and the content is not plagiarized from any other resource. References for all the content taken from external resources are correctly cited. To the best of our knowledge, this report does not contain any material published or written by third parties, except as acknowledged in the text.

Authors:

Author SID	Author name	Signature
IT20012892	Ahamed M.S.A	…….
IT21360114	Mohamed Z.M.N	…….
IT21193804	Weerasekara D.D.R.I	…….
IT21177682	Shehan H.A	…….
IT21177514	Ransika M.R.T	…….
IT20655648	Bandara J.M.O.N	…….
IT21197246	Abeykoon A.M.Y.V.I	…….
IT21158568	Sindujan.P	…….

Date : 5/18/2023

CONTENT

Abstract

This project was developed as part of the Information Technology Project (ITP) module in the second year of the second semester. The objective was to design a Retail Management System for S.A.P super, an apparel manufacturing company. The existing inventory system at S.A.P super had limitations, with data on item sales not aligning with payment collections, resulting in discrepancies in sales and cash flow analysis. Additionally, staff performance evaluation and user feedback processes were manual and inefficient. Asset and liability tracking relied on journals, making it time-consuming to locate and view specific information. Furthermore, the company lacked a dedicated marketing and event planning sector. To address these challenges, the Retail Management System was developed with eight main functionalities tailored to the client's requirements. The functionalities of the system include Inventory Management, Marketing & Sales Management, Point of Sale Management, Human Resource Management, Asset & Expenses Tracking, Customer Relationship Management, Accounting and Financial Tracking, and Supply Chain Management. Each functionality was carefully designed and implemented to enhance the overall efficiency of the business operations and facilitate easy and reliable access to relevant information. In conclusion, the Retail Management System developed for S.A.P super effectively streamlines workflows and improves the efficiency of various business functions. It provides a comprehensive solution for managing inventory, sales, human resources, assets, customer relationships, accounting, financial tracking, and supply chain operations. The system offers a robust platform for increasing productivity, optimizing processes, and enabling informed decision-making.

Acknowledgement

We would like to extend our gratitude towards Sri Lanka Institute of Information Technology (SLIIT) for providing us with the opportunity to do this project as a whole module. Special thanks should go to our lecturer Mr. Jeewaka Perrera, the evaluator Mr. Nisal and Ms. Indika who guided us throughout the semester by providing us the knowledge regarding IT project module and who guided us in every evaluation and corrects our mistakes, giving more ideas to complete the project successfully. Special thanks go to our client Mr. Bandara who is the owner of S.A.P super. Thank you for handing over us this opportunity and believing us. Finally, we would like to thank for everyone who supported to make this project a success.

Introduction

Background

The client of our project is “S.A.P Super”, which is a grocery store situated in Bihalpola Kurunegala, that supplies the neighborhood with their daily wants and needs. They have been operating for close to 5 years and are a partnered company with Darley Butler, a company that acts as the distributing agent in the region. It is by Darley Butler that most products that hit the shelves of S.A.P super come from. They sell items ranging from daily essentials like food and beverages to household items such as books and stationary. The company houses a small staff of around 30 individuals who manage and run the businesses. Among them, a managerial team consisting of accountants, HR representatives and sales representatives are tasked with the responsibilities of running both SAP and Darley Butler.

Problem and Motivation

One of the main problems faced by the business is the lack of tracking of its inventory as many items have been misplaced and are to this day unaccounted for. Furthermore, item restocking has been slow and many items have not been restocked in time, which leads to missed opportunities to make a profit. Certain items that are currently missing from the store also need to be added in order to keep up with the growing and changing consumer needs. The owner also wishes to have an understanding of the performance of the supermarket as well as the performance of the staff employed. Information regarding the assets and liabilities of the business is also not readily available and takes quite a long time to be retrieved.

Currently, the S.A.P has a basic inventory system that logs data on item sales via barcodes. The payments are collected for items sold by cash or card and data comparisons on sales and cash flow do not coincide. Furthermore, the staff performance and user feedback is handled manually and so is relatively inefficient. Assets and liabilities are also logged in the form of journals and so, finding and viewing information pertaining an asset is time consuming. Finally, a marketing and event planning sector does not exist for the business as of yet. As evident, the client's most important requirement is to increase the efficiency of the business, as well as, increase the efficiency of viewing information relating to the business.

The main objective that we strive to achieve through this system is to bring a boost to the efficiency of running the business to its staff and owners by giving (varying) access to a multitude of managerial functions. It is designed in a way that will automate many tasks in the business that is currently done manually as well as help the owners and any other staff interested to better identify customer needs and implement them. The system is also designed to help the owner track different aspects of the business from the comfort of his computer without too much technical expertise or difficulty. Benefits of the System to S.A.P:

1) Automated report generation.

2) Manage and track staff and their performance.

3) Notify staff when a restock is required

4) Increase efficiency in viewing information on assets and liabilities.

5) Plan and implement promotions faster.

6) Calculate expenses

7) Track assets and their current value.

8) Identify Customer requirements and automate implementation of them.

Literature Review

We conducted a study on a few supermarket management systems that already exist. We analyzed these systems for their pros and cons. The following is brief introduction and analysis of them.

QuickBooks POS:

QuickBooks POS is a popular retail management system that provides features such as inventory management, point of sale, and customer relationship management. Its pros include:

Pros:

- Ease of use: QuickBooks POS is a user-friendly system, with an intuitive interface that is easy to navigate.
- Compatibility: The software is compatible with other QuickBooks software, making it easy to integrate with other systems.
- Affordability: QuickBooks POS is an affordable option compared to other retail management systems.

However, it also has some limitations or cons, such as:

Cons:

- Limited scalability: The system may not be suitable for larger businesses, as it has limitations on the number of products and transactions it can handle.
- Limited customization options: The system may not offer the level of customization that some businesses require.

Square POS:

Square POS is a cloud-based retail management system that offers features such as inventory management, point of sale, and employee management. Its pros include:

Pros:

- Easy integration: The software can easily integrate with other Square software, making it a suitable choice for businesses already using other Square products.
- Affordability: Square POS is an affordable option for businesses of all sizes.
- Accessibility: The system can be accessed from any device with an internet connection.

However, it also has some limitations or cons, such as:

Cons:

- 19 Limited customization options: The system may not provide the level of customization that some businesses require.
- Occasional software glitches: Some users have reported software glitches or malfunctions, which may impact the reliability of the system.

Retail Pro:

Retail Pro is a comprehensive retail management system that provides features such as inventory management, point of sale, and customer relationship management. Its pros include:

Pros:

- High level of customization: Retail Pro offers a high level of customization, making it a suitable choice for businesses with unique requirements.

- Scalability: The system can handle large numbers of products and transactions, making it a suitable option for larger businesses.
- Integrations: Retail Pro can integrate with other software, such as e-commerce platforms and accounting software.

However, it also has some limitations or cons, such as:

Cons:

- Steep learning curve: The software may have a steep learning curve, requiring extensive training for employees to use effectively.
- High cost: Retail Pro is one of the more expensive retail management systems available, which may not be affordable for smaller businesses.

What makes our system different?

After conducting studies as to other systems that exist in the market, it is evident that similar systems do exist that help business log their sales. However, while some systems offer more functionality than ours, they are often too complicated and cause a steep learning curve when introduced to the staff of said businesses. On the other hand, other systems that are simple in design and usability are very expensive in their subscription based model. This too is just POS systems and further integration of HR and Asset management tools require different software or expansions to be accessible. That leaves us with our system which is customizable to our client's needs while providing many different functionalities that are needed to run the business including our own Human Resource management, Marketing and Asset Management systems. Our system will also be user friendly and relatively easy to understand and use, even for a beginner

Methodology

The proposed Retail Management system is a full-stack web application that employs four technologies, collectively known as MERN, including MongoDB, Express, ReactJS and Node JS. This system is broken down 3 main components being: Front end, Back end and Web Server. 5 React JS is a client-side browser library that facilitates the creation of highly responsive user interfaces, referred to as the Front end. Node Js is a server-side browser library that operates outside of the browser, accepts requests and sends back responses over the internet. It executes server-side logic, interacts with databases and files and is typically used in combination with Express. Express, a Node Js framework, offers utility functions and a structured approach to building application that provide new functionalities. These two technologies are collectively referred to as the Back end. MongoDB, a NoSQL database engine, stores documents in collections instead of records and tables. For security reasons, it is not directly connected to Mongo DB from inside React applications, but rather with help of Node and Express. MongoDB is referred to as the Web Server. Inventory Controller, HR Controller, Supply Controller, Account Controller, Marketing Controller, Sales Controller and Admin are the main roles that will be in our system. There will be a separate login for each user role to access each function of the system. Each role cannot access other functionalities. Further each user can generate an important report using system data. The admin of the system can access each user role and can view all functionalities of the whole system. The system will use an Application Process Interface to communicate between the different subsystems in the Retail management system.

Structure of the report

The project will be discussed under five main sections. Those five sections will be . Introduction, Requirements , Design and Development, Testing, Evaluation and Conclusion. Details of those five sections and componentsof the system will be discussed under each section separately. Figures, diagrams, and tables related to each, and every section and component of the system will be included in the section there themselves and supplementary material will be included in the appendices.

A clickable link to the Git repo

https://github.com/SLIITITP/y2_s2_wd_it_01-itp_wd_b02_g01

Requirements

Stakeholder analysis

Customers:

- Interests: They want a seamless and efficient purchasing experience, accurate payment processing, availability of requested items, and access to discounts and offers.
- Influence: They influence the success of the business through their purchasing decisions and feedback.

Sales and Marketing Team:

- Interests: They aim to drive sales through effective marketing campaigns, validate and apply discounts accurately, and enhance customer satisfaction.
- Influence: They influence the purchasing decisions of customers through marketing efforts and play a role in shaping customer experience.

Staff Members:

- Interests: They require user-friendly systems that enable them to perform their tasks efficiently, such as processing sales, managing inventory, logging cash flow, and handling customer requests.
- Influence: Their performance and adherence to system processes directly impact the smooth operation of the business.

Inventory Management Personnel:

- Interests: They need an inventory management system that accurately tracks stock levels, generates restock notifications, and ensures availability of items.
- Influence: Their efficient management of inventory helps meet customer demand and prevents stockouts.

Owner/Accountant:

- Interests: They seek an asset and expense tracking system that provides accurate calculations, generates financial reports, and aids in decision-making.
- Influence: Their oversight of financial management and asset tracking is crucial for the business's financial health and planning.

Human Resources Department:

- Interests: They require a reliable HR management system to add and manage staff members, track performance, and maintain personnel records.
- Influence: They contribute to building a capable and productive workforce through effective management and monitoring of staff.

System Administrator:

- Interests: They need a well-maintained system, proper access controls, and regular updates to ensure smooth functioning.
- Influence: They play a vital role in system maintenance, security, and user management, impacting the overall system performance.

Suppliers:

- Interests: They are interested in timely and accurate orders, clear communication regarding stock requirements, and a smooth ordering and delivery process.
- Influence: Suppliers play a crucial role in ensuring the availability of products, timely deliveries, and maintaining a good relationship with the business.

Requirements Analysis & Modelling

Accounting and Finance Management System

(IT21197246 – A.M.Y.V.B.Abeykoon)

The main objective of this function is to compute accounting calculations using data from the other systems. To do this, it accessed the APIs of the other systems, processes said data and displays it on a dashboard. The finance manager is also able to add, edit, delete and view other incomes and expenses as well as view purchases and sales. The manager can also filter through incomes, expenses, purchases and sales via a search bar. Finally, reports pertaining to the data can also be downloaded in pdf form.

System Actor

Finance Manager

The Finance manager is responsible for overseeing the financial aspect of the business. The manager can add, edit, view and delete bonus incomes and expenses that pertain to the business that are not of the sales, purchase and assets & liability category. Furthermore, the finance manager has access to the financial details of the business that are calculated via the information the system is fed through the other management systems.

Functional and non-functional requirements of Accounting and Finance

Functional Requirements

- Finance manager should be able to add bonus incomes and expenses.
- Finance manager should be able to edit bonus incomes and expenses.
- Finance manager should be able to delete bonus incomes and expenses.
- When incomes and expenses are recorded, the system records:
    - Title
    - Amount
    - Category
    - Description
    - Type
    - Date
- The title, category, description, type and date must not be left empty.
- The amount entered must not be negative, empty or zero.
- Information relating to incomes, expenses, sales and purchases on the system can be downloaded in PDF form.
- The system can compute :
    - Total cash leaving the business
    - Total cash entering the business
    - Gross profit
    - Closing stock

    - Opening Stock
    - Total Sales
    - Total Purchases
    - Total Incomes
    - Total Expenses
- The above mentioned data is displayed on the main Finance dashboard page.

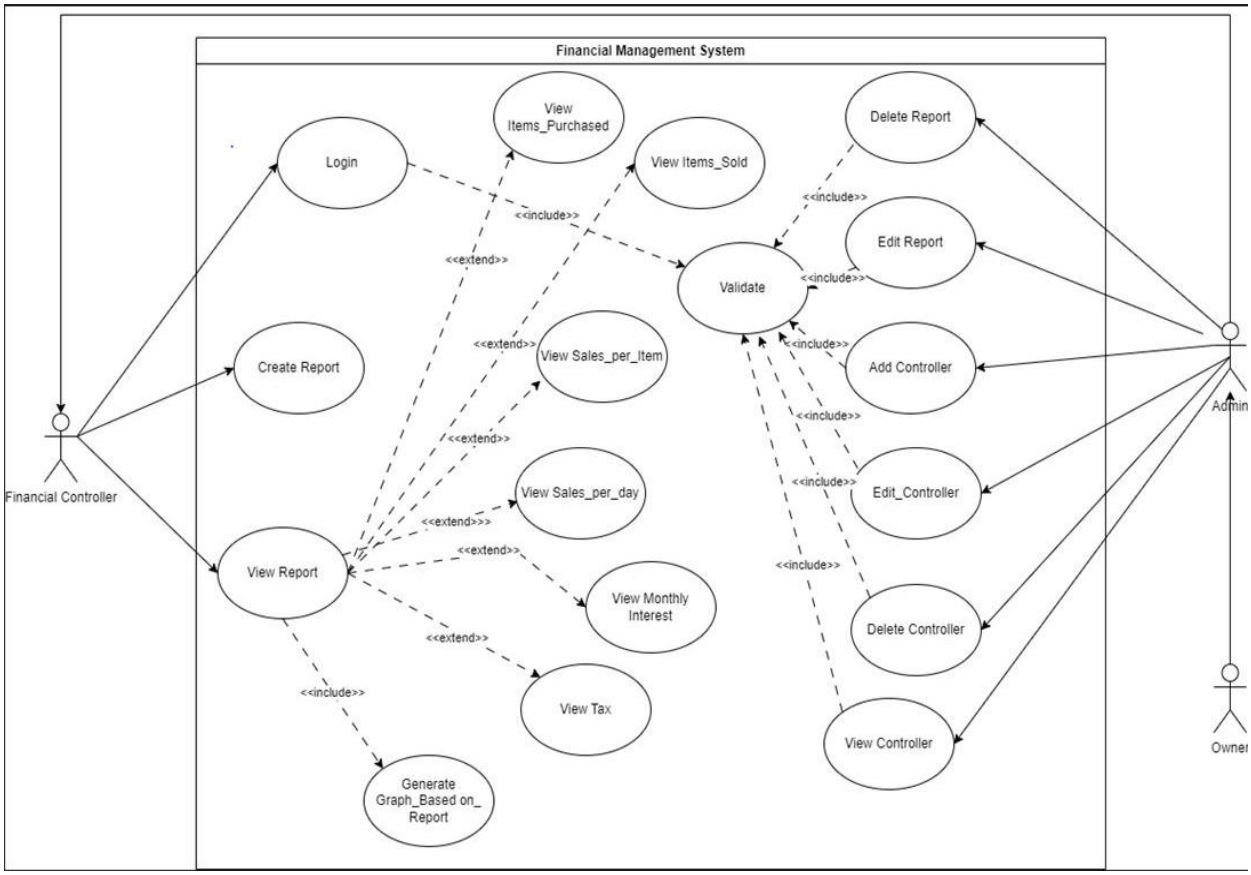
Non-Functional Requirements

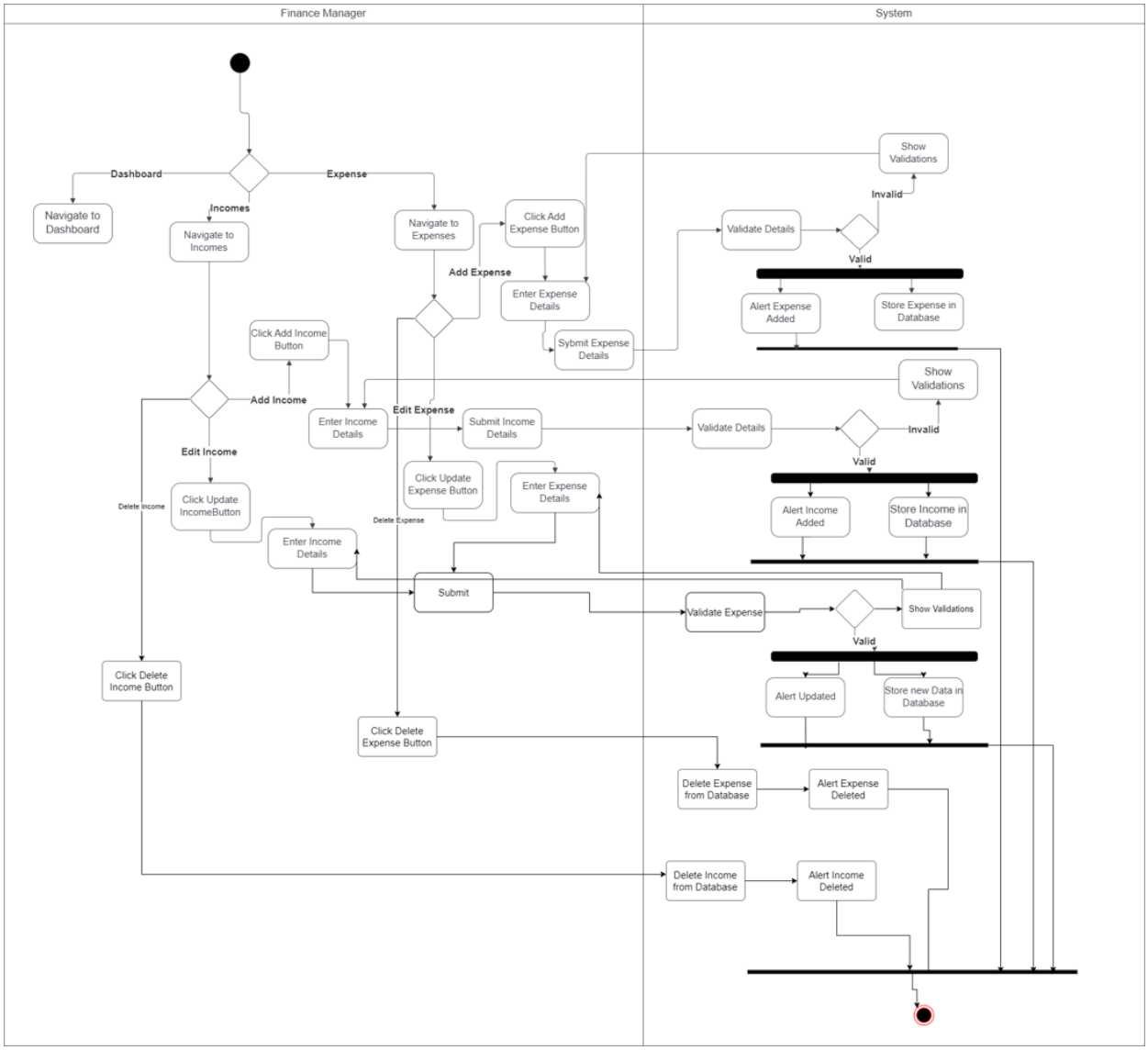
1) Performance
    • The response and processing time of each interface are within a few minutes.
    • Website must load quickly as possible.
2) Security
    • Only admin can generate the report.
    • If customer wishes website and system must allow option to logout.
    • Registered customers have to login to the website by providing their username and password.
3) Safety
    • Each user has unique username and password to login and access their accounts on the website.
    • If customers forget their account login password, he can contact the administration through the message provided in the homepage. Accessible link will be resent to the user so by clicking on it the password can be reset.

4) Software Quality Attributes

- Availability - The system is available for all authorized customers and staffs to access when needed (24x7).
- System Maintenance - If any faults are found within the system it should be noted by the staff members and forwarded to the development team.
- Reliability - Development team continues to monitor how the system is performing and continue to make improvements to the system through updates.
- Usability – User can easily understand the system and functions and operate them. Interfaces should be user friendly.
- Accuracy - All the managements function with expected response

Diagram


Financial Management System
Login
View Items_Purchased
View Items_Sold
Delete Report
Edit Report
Validate
Create Report
View Sales_per_Item
Add Controller
Admin
Financial Controller
View Sales_per_day
Edit_Controller
View Report
View Monthly Interest
Delete Controller
View Tax
View Controller
Generate Graph_Based on_ Report
Owner
<<include>>
<<extend>>


Finance Manager
System
Dashboard
Expense
Incomes
Navigate to Dashboard
Navigate to Incomes
Navigate to Expenses
Click Add Expense Button
Add Expense
Enter Expense Details
Sybmit Expense Details
Click Add Income Button
Add Income
Enter Income Details
Edit Expense
Submit Income Details
Edit Income
Click Update IncomeButton
Delete Income
Click Update Expense Button
Enter Expense Details
Delete Expense
Enter Income Details
Submit
Click Delete Income Button
Click Delete Expense Button
Show Validations
Invalid
Validate Details
Valid
Alert Expense Added
Store Expense in Database
Show Validations
Invalid
Validate Details
Valid
Alert Income Added
Store Income in Database
Validate Expense
Show Validations
Valid
Alert Updated
Store new Data in Database
Delete Expense from Database
Alert Expense Deleted
Delete Income from Database
Alert Income Deleted

Supply Chain management System

IT21193804 - Weerasekara D.D.R.R

The supply chain management function is responsible for managing the replenishment of inventory and ensuring the availability of necessary supplies. there are two main categories in the resupply chain management: Adding supplier details to the system and emergency making resupply orders with those details.

The supply chain manager oversees the entire process and collaborates with other managers, such as inventory managers and financial managers, to ensure smooth operations. They have access to the resupply chain management function and can initiate and monitor resupply activities.

The supply chain manager maintains a list supplier details and a list of each resupply order. They review the inventory levels, consumption patterns, and demand forecasts to determine the appropriate resupply quantities and timings. After assessing the information, the resupply chain manager places the resupply orders with the designated suppliers.

All financial transactions related to resupply chain management, such as purchase orders, invoices, and payments, are handled by the account management function. The resupply chain manager works closely with the account management team to ensure accurate financial records and timely payments to suppliers.

Overall, the resupply chain management function plays a crucial role in maintaining the availability of supplies, ensuring smooth operations, and meeting the organization's demands in an efficient and cost-effective manner.

Functional and non-functional requirements of Supply Chain Management

Functional Requirements

• Supply manager should be able to create Supplier accounts and supply orders.

• And Supply manager can edit and delete those details.

• When adding a Supplier following records are saved in the database.

- Supplier ID
- Supplier Name
- Phone number
- Item Type
- Payment Details

• And all the field has proper validation.

• phone number ten-digit number unless it generate error message.

• all field must be filled unless it generates error message.

• ones generate report button pressed system will generate PDF with complete details of all supplier details.

• When adding a new Supply Order following records are saved in the database.

- Order ID
- SID
- Supplier Name
- Item
- Amount of the items
- Price
- Discount
- Delivery Date

• And all the field has proper validation.

• Delivery date should be in valid format unless it generates error message.

•Amount and Price must be a number and it should be valid positive number to save supply order details to the database.

• ones generate report button pressed system will generate PDF with complete details of Supply orders.

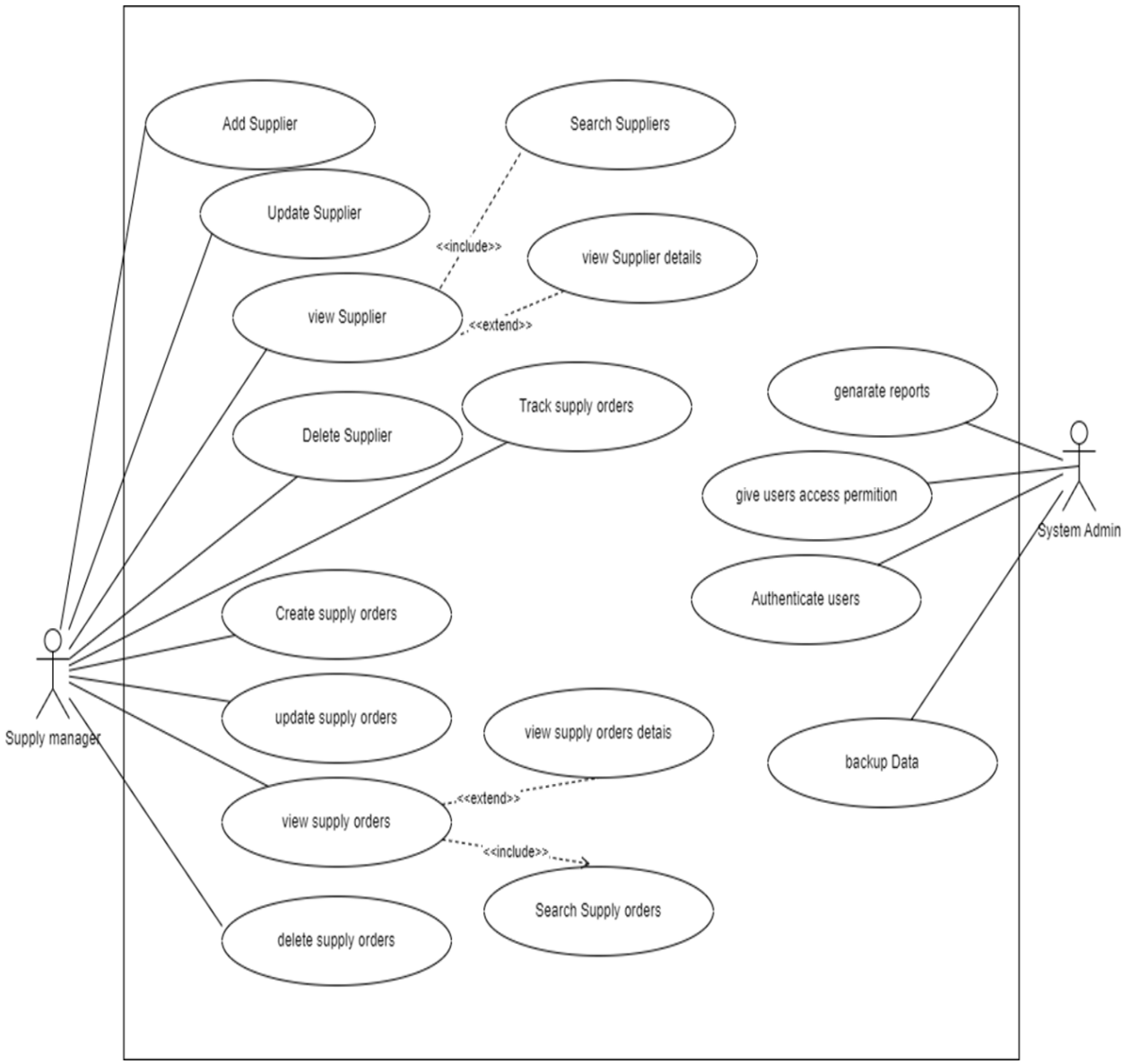
Non-Functional Requirements

1) Performance
    - The response and processing time of each interface are within a few minutes.
    - Website must load quickly as possible.
2) Security
    - Only admin can generate the report.
    - If customer wishes website and system must allow option to logout.
    - Registered customers have to login to the website by providing their username and password.
3) Safety
    - Each user has unique username and password to login and access their accounts on the website.
    - If customers forget their account login password, he can contact the administration through the message provided in the homepage. Accessible link will be resent to the user so by clicking on it the password can be reset.

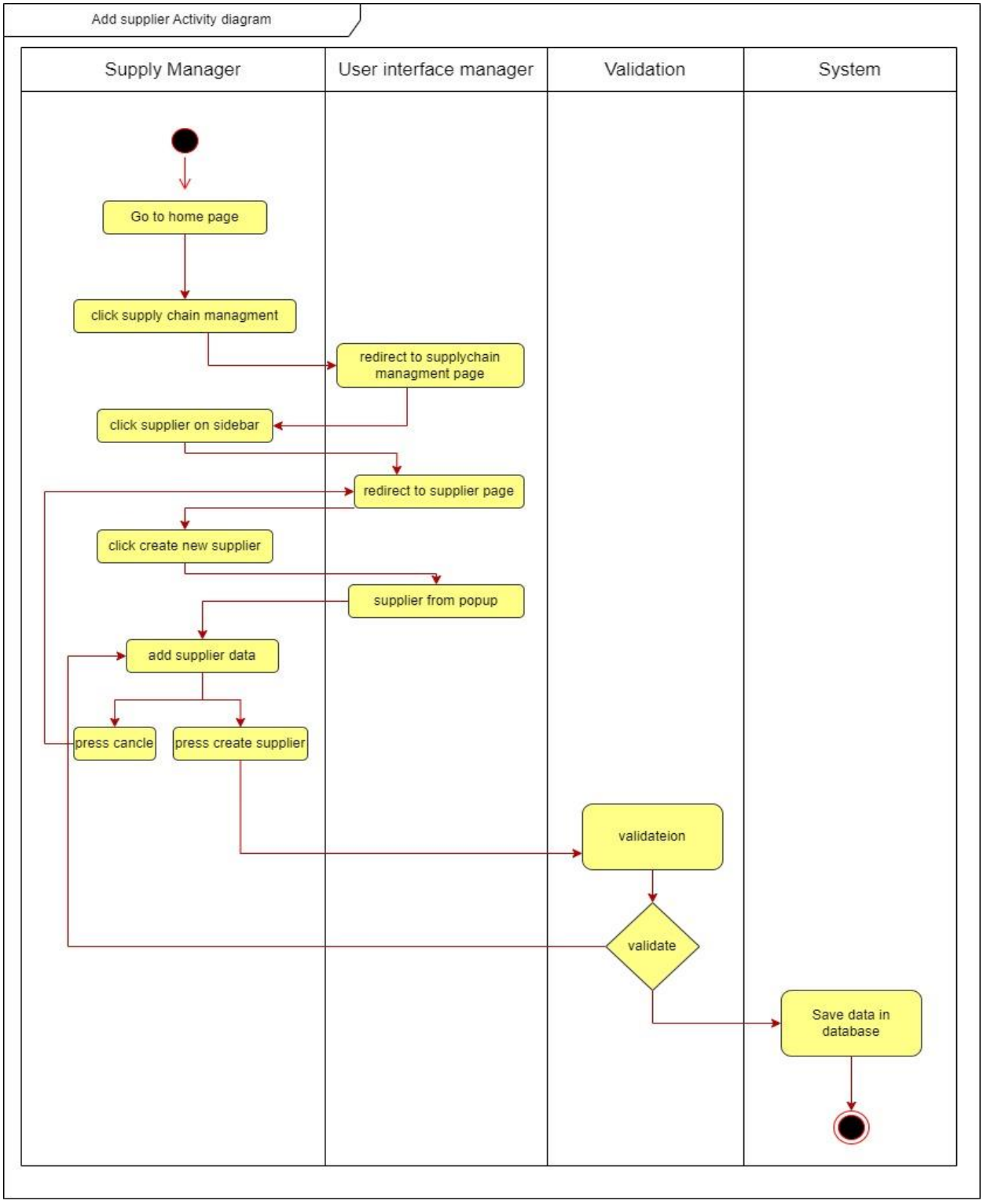
4) Software Quality Attributes
    - Availability - The system is available for all authorized customers and staffs to access when needed (24x7).

- System Maintenance - If any faults are found within the system it should be noted by the staff members and forwarded to the development team.
- Reliability - Development team continues to monitor how the system is performing and continue to make improvements to the system through updates.
- Usability – User can easily understand the system and functions and operate them. Interfaces should be user friendly.
- Accuracy - All the managements function with expected response

Supply chain management User Case diagram



Supplier add activity Diagram



Customer Relationship Management System.

IT21177514 - Ransika M.R.T.

In the process of doing business customers may have to face different problems like Problems with purchased goods and services, Invoicing problems, Late deliveries, Problems with refund policies, and Ignorance about services and goods provided by the business. There should be a productive communication method to use customers to inform their business-related inquiries as well as business should have a productive response method to help customers with different problems.

For a business to develop day by day, it must implement new ideas and methods. There, the views of the parties directly related to the business (Business Owners, Management units, Other internal employees) as well as indirectly related parties like customers are very important. Therefore, there should be a productive method for customers to publish their own constructive feedback related to business.

It is important to offer discounts, and packages and inform new business updates to the customers to increase their attraction towards the business. There should be a method to inform customers about those offers and new updates related to business.

The Customer Relationship Management System of the SAP Retail Management System will manage customer inquiries and customer feedback and inform new business updates and offers to the customers.

Functional and non-functional requirements Customer Relationship Management System

Functional Requirements

1. Registering customers and maintaining a collection of customer information.
2. Viewing, updating, and deleting customer registration credentials.
3. Maintaining a portal to collect problems and feedback from customers.
4. Viewing, updating, deleting, and analyzing customer problems and feedback.
5. Maintain a FAQ section.
6. Respond to problems faced by customers.
7. Communicating seasonal discounts and packages to registered customers.

Non-Functional Requirements

1)Performance

- •The response and processing time of each interface are within a few minutes.
- •Website must load quickly as possible.

2)Security

- •Only admin can generate the report.

•If customer wishes website and system must allow option to logout.

•Registered customers have to login to the website by providing their username and password.

3)Safety

•Each user has unique username and password to login and access their accounts on the website.

•If customers forget their account login password, he can contact the administration through the message provided in the homepage. Accessible link will be resent to the user so by clicking on it the password can be reset.

4)Software Quality Attributes

•Availability - The system is available for all authorized customers and staffs to access when needed (24x7).

•System Maintenance - If any faults are found within the system it should be noted by the staff members and forwarded to the development team.

•Reliability - Development team continues to monitor how the system is performing and continue to make improvements to the system through updates.

•Usability – User can easily understand the system and functions and operate them. Interfaces should be user friendly.

•Accuracy - All the managements function with expected response

Use Case Diagram

Assets and Expenses Tracking System

Bandara J.M.O.N - IT20655648

An Asset and Expense Tracking system is designed to help businesses manage their assets, loans, and expenses. The system must be able to track the acquisition and depreciation of assets, as well as keep track of loans and their interest rates. The system will store information such as the date of acquisition, initial value of the asset or loan, interest rates, and annual depreciation or interest expense.

A system for tracking assets and expenses is employed by companies to manage all of their assets throughout the organization, whether they are tangible or intangible. This can encompass personnel, buildings, software and hardware, inventory, monetary assets, and other essential components required for the smooth operation of the business on a daily basis.

This information can be used to calculate the current value of the asset, and to generate reports for financial analysis and planning. The system must also be able to calculate the depreciation of assets over time, based on the useful life of the asset.

Customer can view the more detail about assets and special view real value of assets.

Functional and non-functional requirements of Assets and Expenses Tracking System

Functional Requirements

- Need to calculate depreciation and interest.
- User can update and delete those details.

- When adding a user following records are saved in the database.
    - Item code
    - Item Name
    - Date
    - Amount
    - Residual value
    - Item Keep years in company

    - All the field has proper validation.
    - All field must be filled generates error message.
- When adding a Expenses following records are saved in the database.
    - Item code
    - Name
    - Date
    - Amount

- Ratio
- Item Keep years in company

• All the field has proper validation.

• Amount must be a number and it should be valid positive number to save details to the database.

Non-Functional Requirements

1)Performance

•The response and processing time of each interface are within a few minutes.

•Website must load quickly as possible.

2)Security

•Only admin can generate the report.

•If customer wishes website and system must allow option to logout.

•Registered customers have to login to the website by providing their username and password.

3)Safety

•Each user has unique username and password to login and access their accounts on the website.

•If customers forget their account login password, he can contact the administration through the message provided in the homepage. Accessible link will be resent to the user so by clicking on it the password can be reset.

4)Software Quality Attributes

•Availability - The system is available for all authorized customers and staffs to access when needed (24x7).

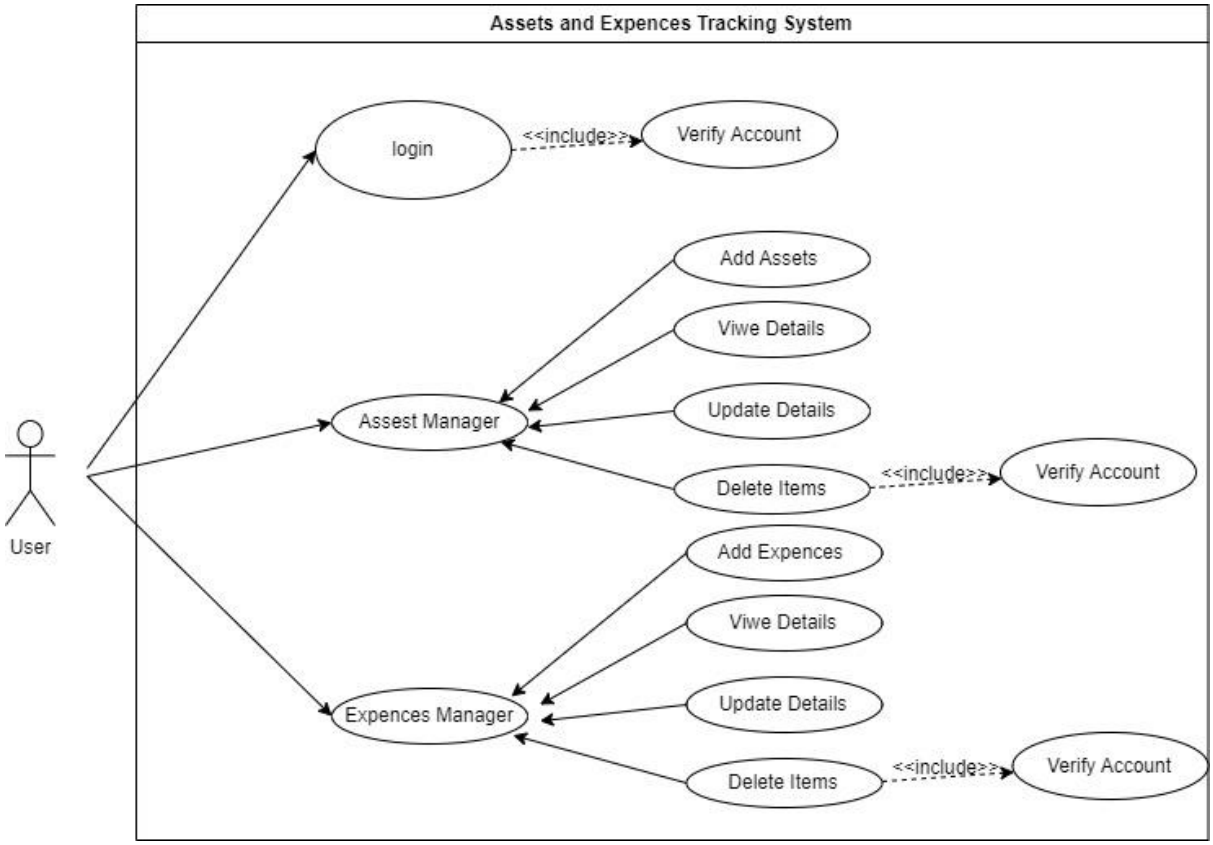
•System Maintenance - If any faults are found within the system it should be noted by the staff members and forwarded to the development team.

•Reliability - Development team continues to monitor how the system is performing and continue to make improvements to the system through updates.

•Usability – User can easily understand the system and functions and operate them. Interfaces should be user friendly.

•Accuracy - All the managements function with expected response

User Case diagram - Assets and Expenses Tracking System



Marketing and sales management system

Shehan H.A - IT21177682

Marketing and sales management system helps businesses manage their marketing activities more efficiency and effectively. The system should allow the marketing team to create and manage marketing projects, set deadlines, assign tasks to team members, and track progress. The system should allow the team to create and manage marketing campaigns, including email marketing, social media marketing, and paid advertising. The team should be able to track the performance of each campaign and make adjustments as needed. The system should provide analytics and reporting tools that will allow the team to track the performance of their marketing efforts, identify areas for improvement, and make data-driven decisions.

Functional and non-functional requirements of Production Management

Functional Requirements ·

Marketing Manager should be able to add new Campaign from to the system. ·

When adding a new campaign to the system, it should be record following details in to the database.

- Campaign Id
- Item code
- Item name
- Media type
- Target Audience
- Start Date
- End Date
- Price
- Discount
- Quantity
- Note
- Status

Quantity and Price have input types in number, and it can't be a negative number also. ·

The Marketing Manager should be able to view all the Campaign Details.

The Marketing Manager should be able to edit the Campaign details.

The Marketing Manager should be able to delete Campaign details. ·

The Marketing Manager should be able to generate a report of campaign details.

- As a pdf format
- As a excel format ·

Marketing Manager should be able to search any campaign detail By Item code and item name.

And all the fields have proper validation.

All the campaign status categories and view the count of card view.

Non-Functional Requirements

1)Performance

- •The response and processing time of each interface are within a few minutes.
- •Website must load quickly as possible.

2)Security

- •Only admin can generate the report.
- •If customer wishes website and system must allow option to logout.
- •Registered customers have to login to the website by providing their username and password.

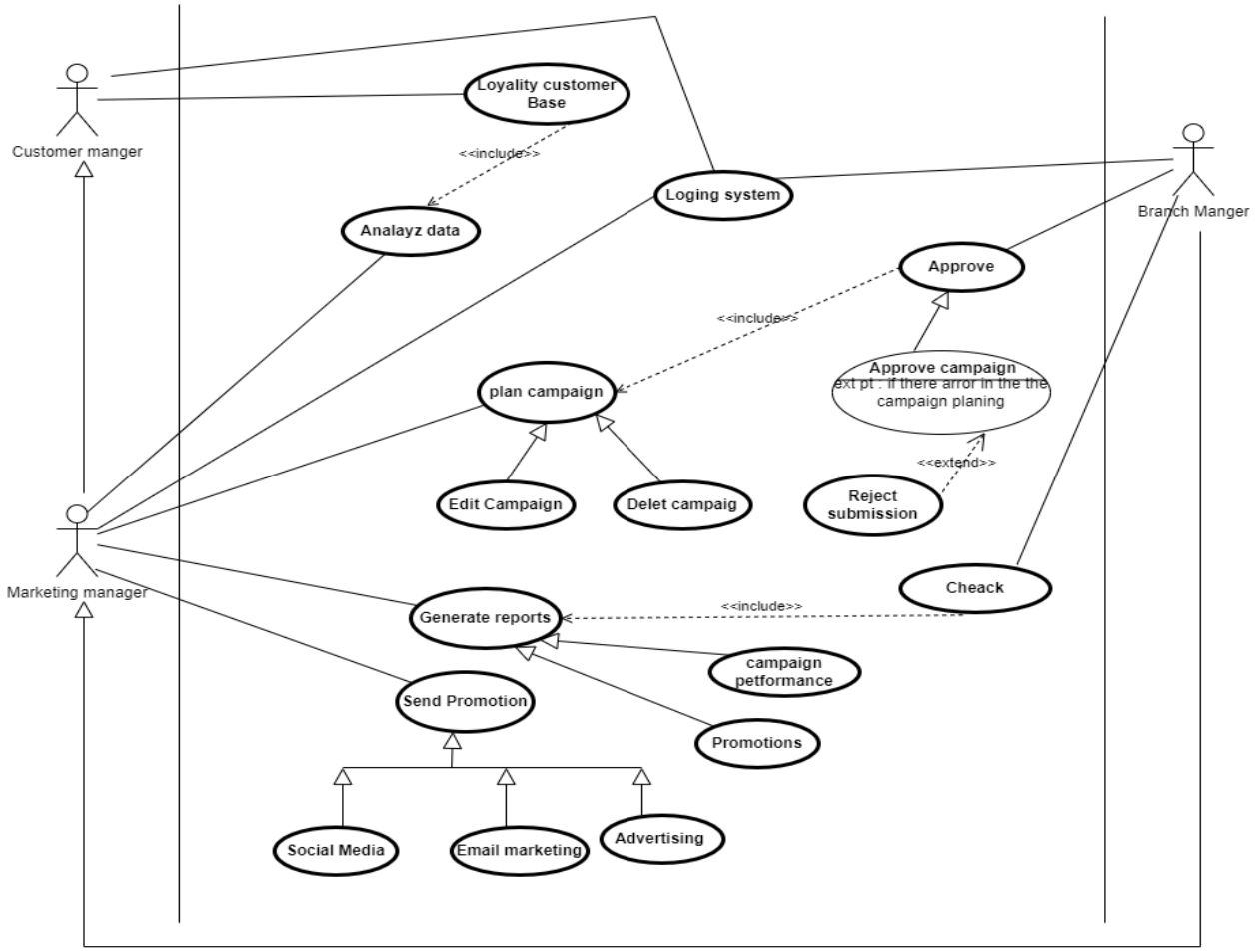
3)Safety

- •Each user has unique username and password to login and access their accounts on the website.
- •If customers forget their account login password, he can contact the administration through the message provided in the homepage. Accessible link will be resent to the user so by clicking on it the password can be reset.

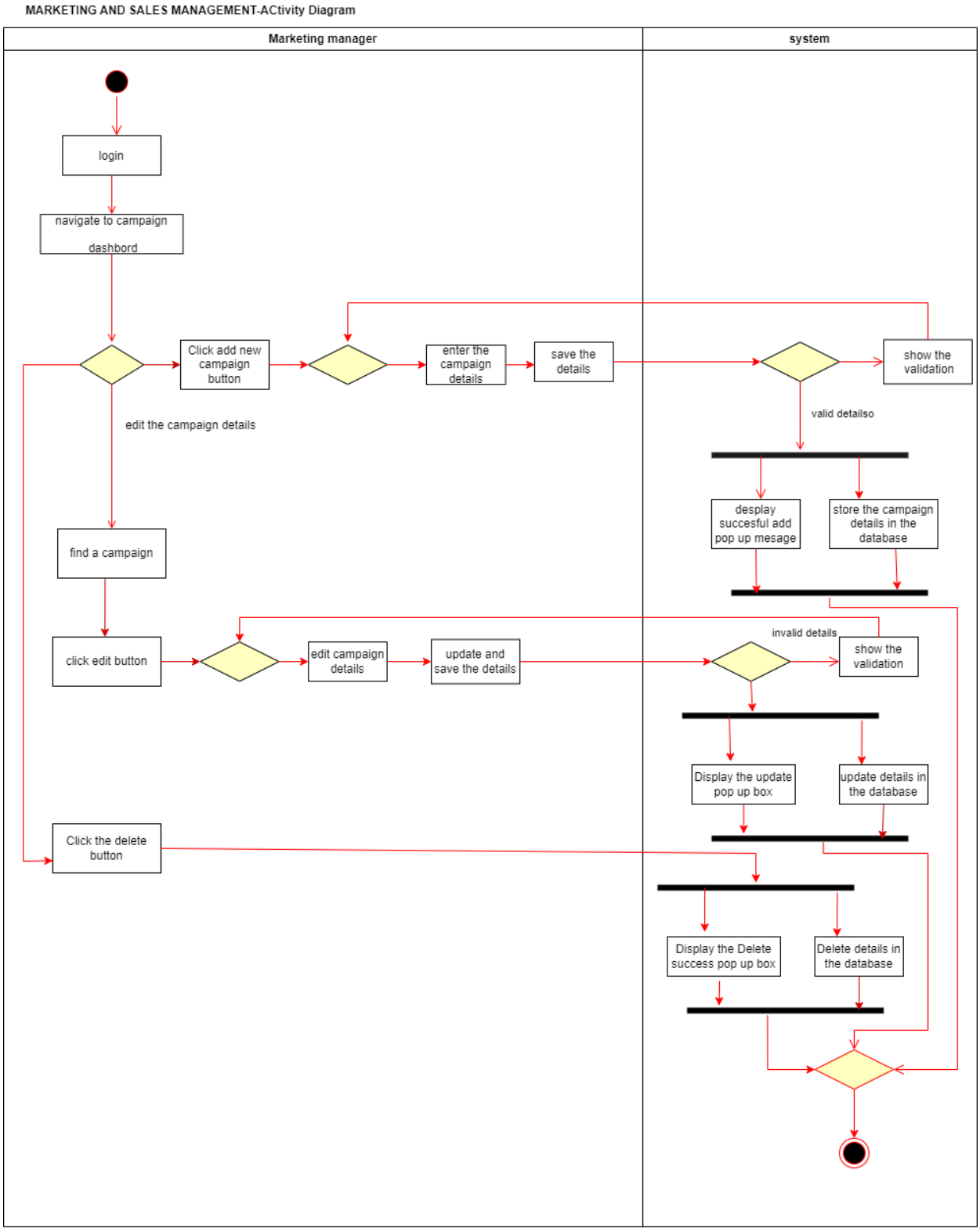
4)Software Quality Attributes

- •Availability - The system is available for all authorized customers and staffs to access when needed (24x7).
- •System Maintenance - If any faults are found within the system it should be noted by the staff members and forwarded to the development team.
- •Reliability - Development team continues to monitor how the system is performing and continue to make improvements to the system through updates.
- •Usability – User can easily understand the system and functions and operate them. Interfaces should be user friendly.
- •Accuracy - All the managements function with expected response

User case diagram for marketing and sales management system



Activity diagram for marketing and sales management system



Point of Sales system

Sindujan.P - IT21158568

A point of sale (POS) system, which makes it easier to manage sales transactions and speed up the checkout process, is an essential part of an SAP store website. It has several features, such as creating bills and adding, editing, and deleting things. Here is a quick summary of each feature.

Adding things: Staff can add new items to their inventory using the POS system. Entering pertinent information such the item name, description, price, and any applicable fees or discounts is required. These goods can be chosen from during the checkout process once they have been added.

Updating inventory: Companies frequently need to make changes to the things already in their stock. Users of the POS system can edit item details including price, description, stock level, and any other pertinent qualities. For make invoice Turley

Items can be deleted from the inventory using the POS system when they are no longer offered or are being phased out. This helps keep an accurate record of the products.

Bill generation: The POS system creates a bill or receipt summarizing the transactional information when a customer completes their purchase.

In general, a strong POS system that is incorporated with a company website gives firms the ability to effectively manage their sales activities. The process of adding, updating, and deleting things from the inventory is made simpler, and it also makes it possible to generate bills quickly and easily, assuring efficient transactions and precise record-keeping of customers.

System Actor

Cashier

The Cashier responsible for point of sales system. The casher can add items to cart , update , delete and view sales and make invoice and get customer name.

Functional and non-functional requirements of Point of Sales system

Functional requirements

Item management: The system should enable the addition, updating, and deletion of items from the inventory, together with their associated information (such as name, description, price, quantity in stock, and any related discounts or taxes).

Sales processing: The system should make it easier to choose items, compute totals, apply discounts.

Billing and receipt generation: For each transaction, the system should provide correct bills or receipts that include itemized information, totals, any applicable taxes, discounts, and payment details. Additionally, it must enable customers to print or get digital receipts.

The system can compute total sales

When items adding new bills, it should be take

Customer name,
Contact number,
Total Amount,
Payment method.

When items adding new item it should be take

Item name,
Price ,
Image URL,
Category.

Non-Functional Requirements

1)Performance

•The response and processing time of each interface are within a few minutes.

•Website must load quickly as possible.

2)Security

•Only admin can generate the report.

•If customer wishes website and system must allow option to logout.

•Registered customers have to login to the website by providing their username and password.

3)Safety

•Each user has unique username and password to login and access their accounts on the website.

•If customers forget their account login password, he can contact the administration through the message provided in the homepage. Accessible link will be resent to the user so by clicking on it the password can be reset.

4)Software Quality Attributes

•Availability - The system is available for all authorized customers and staffs to access when needed (24x7).

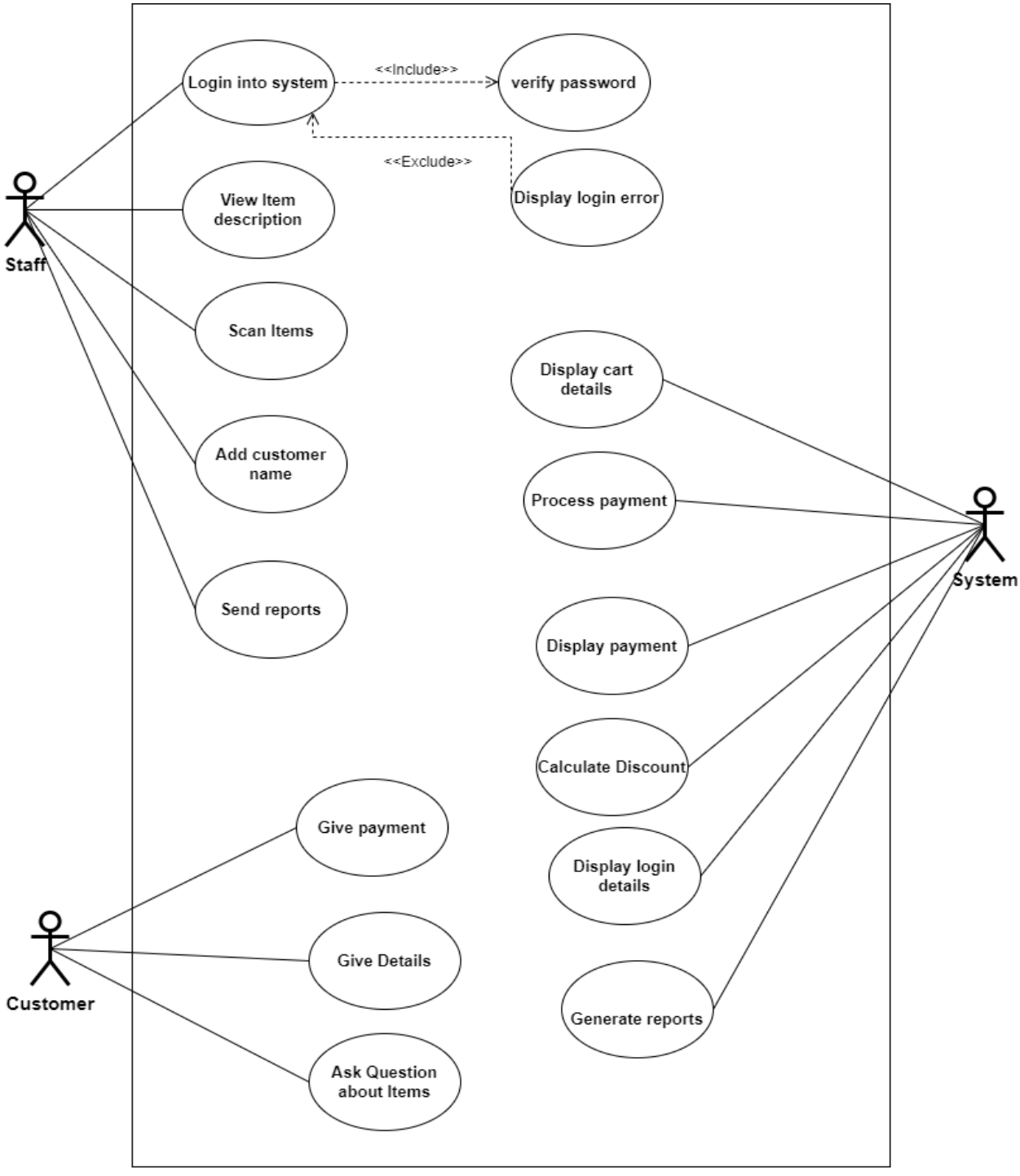
•System Maintenance - If any faults are found within the system it should be noted by the staff members and forwarded to the development team.

•Reliability - Development team continues to monitor how the system is performing and continue to make improvements to the system through updates.

•Usability – User can easily understand the system and functions and operate them. Interfaces should be user friendly.

•Accuracy - All the managements function with expected response

User case diagram for Point of sales system



Human Resource Management System

IT21360114 – Mohamed Z.M.N

System Actors

HR Manager

HR is the main leader of the this subsystem. He can view the employer's details, show the details, read the details, and update the details. He can view details anywhere, anytime given from employers.

All the privacy details and workplace handled by the HR manager. The employer can responsible for the work place if have any doughts about the system works or not then complain to the HR and resolve them. HR manager can do anything from system. He can add new employers, view their details, read their details, update and delete details if no longer use for the system. They generate and ex HR management system is an individual and independent system compare to other 6 systems. So finally when enter the details in DB they can generate a pdf or excel sheets and view another folder from this system. HR Manager can also view their data s. The working salary create by how they working hours counting.

Functional and non – functional requirements of Human resource management system.

Functional Requirements

- HR manager can Add the employee's details.
- When adding new employee details system should recorded these details in database
    - Emp_id
    - Emp_name
    - Age
    - Salary
    - Working hours
    - Job category
    - Description.
- When you enter id and name they should input Emp_ word both, if don't entered dose not work.
- Age shouldn't lower than 18.
- Working hours should minimum 5 hours.
- HR manager should able to see employee details.
- HR manager should able to Delete employee details.
- HR manager should able to Update employee details.
- HR manager should able to View employee details.
- HR manager should able to create employee details.
- HR manager should able to generate the report of employee details.
    - PDF
    - Excel sheet
    - HTML form
- HR manager should able to see different kind of job category details.(My system include it, accountant, supply, financial, POS managers)
- HR manager should able to see employee details by the columns.

Non-Functional Requirements

1)Performance

•The response and processing time of each interface are within a few minutes.

•Website must load quickly as possible.

2)Security

•Only admin can generate the report.

•If customer wishes website and system must allow option to logout.

•Registered customers have to login to the website by providing their username and password.

3)Safety

•Each user has unique username and password to login and access their accounts on the website.

•If customers forget their account login password, he can contact the administration through the message provided in the homepage. Accessible link will be resent to the user so by clicking on it the password can be reset.

4)Software Quality Attributes

•Availability - The system is available for all authorized customers and staffs to access when needed (24x7).

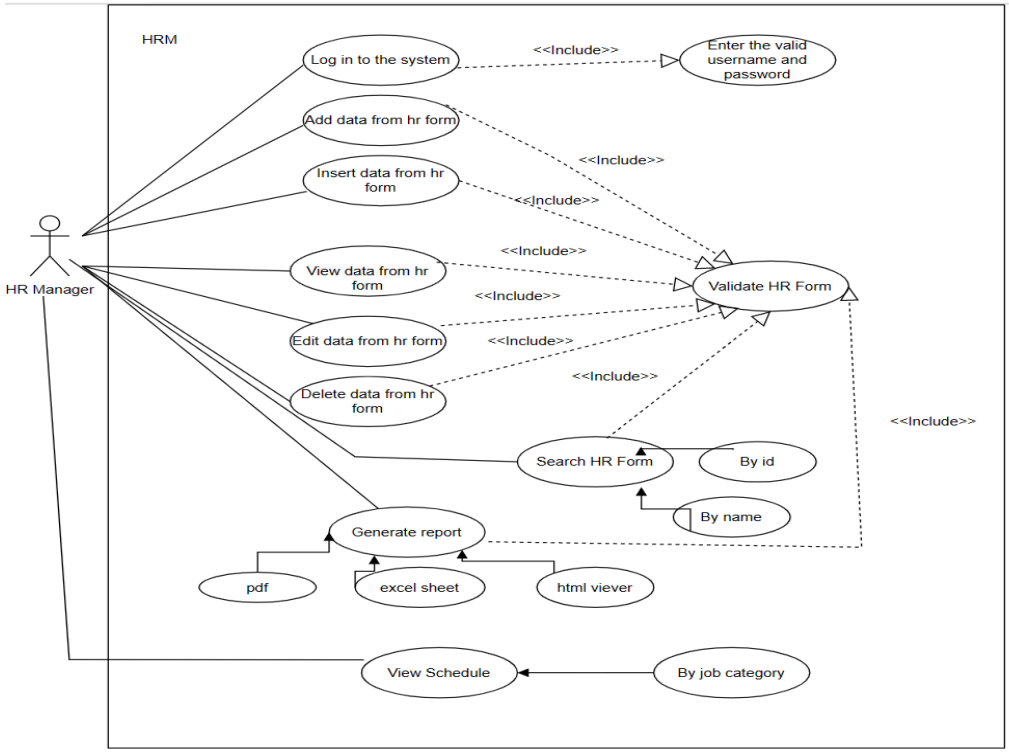
•System Maintenance - If any faults are found within the system it should be noted by the staff members and forwarded to the development team.

•Reliability - Development team continues to monitor how the system is performing and continue to make improvements to the system through updates.

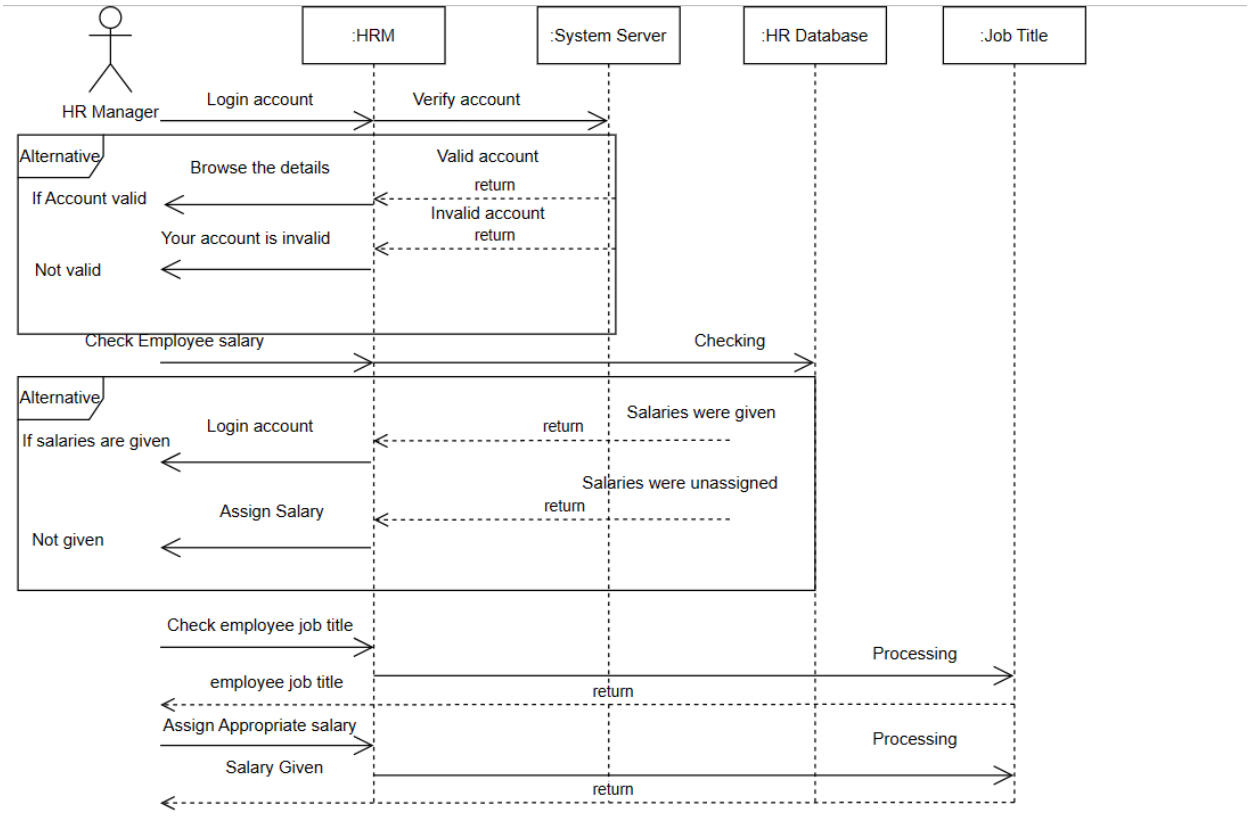
•Usability – User can easily understand the system and functions and operate them. Interfaces should be user friendly.

•Accuracy - All the managements function with expected response

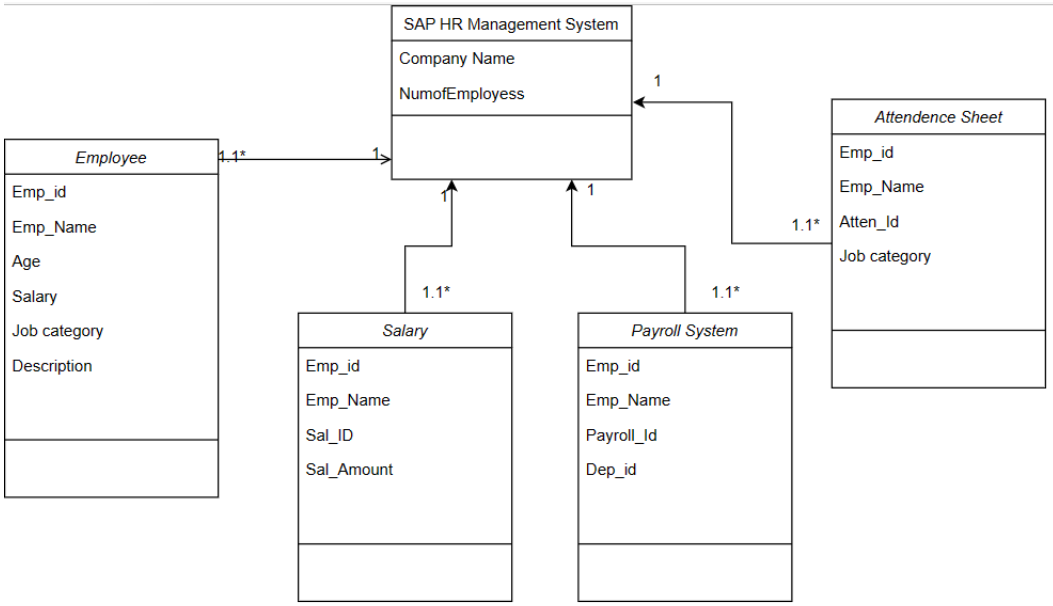
Use case Diagram of Human Resource Management System



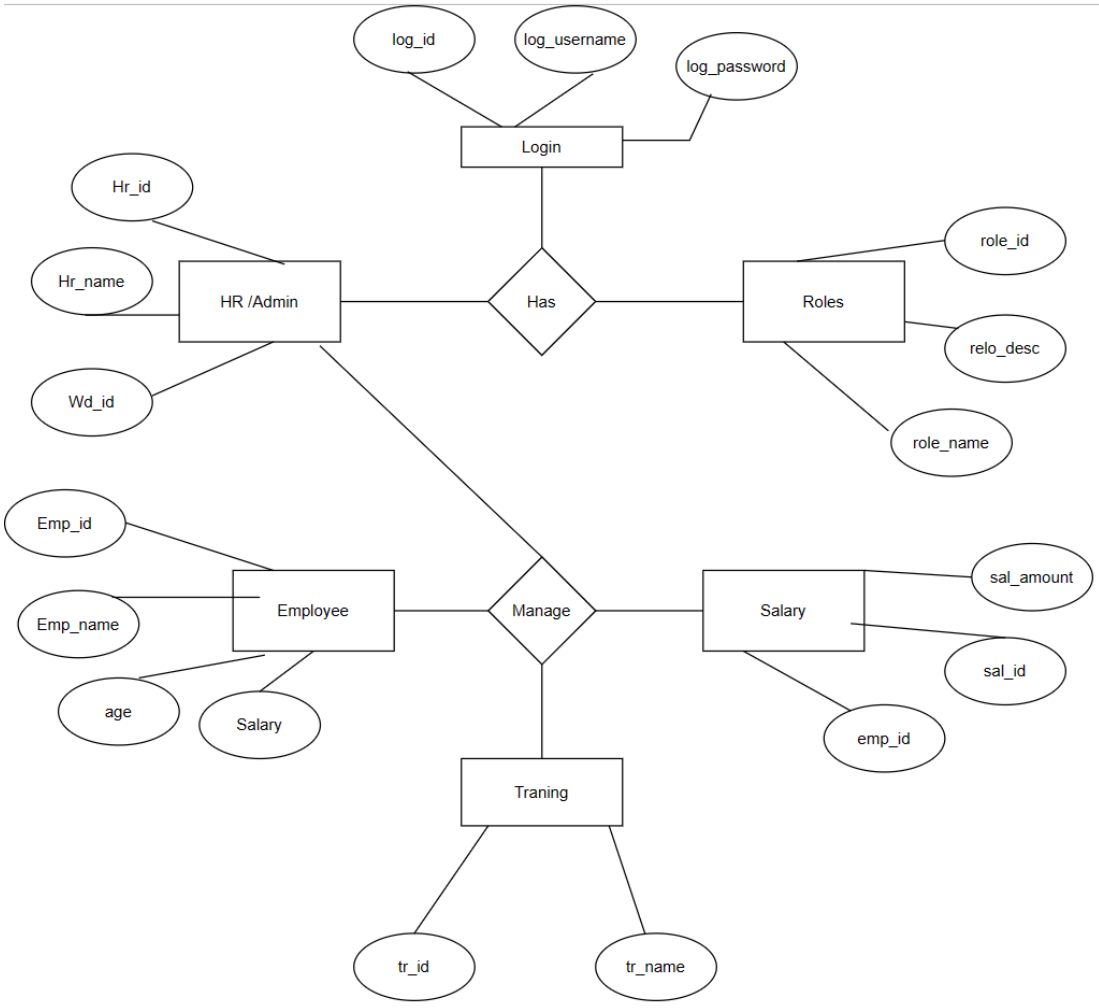
Sequence Diagram of Human Resource Management System



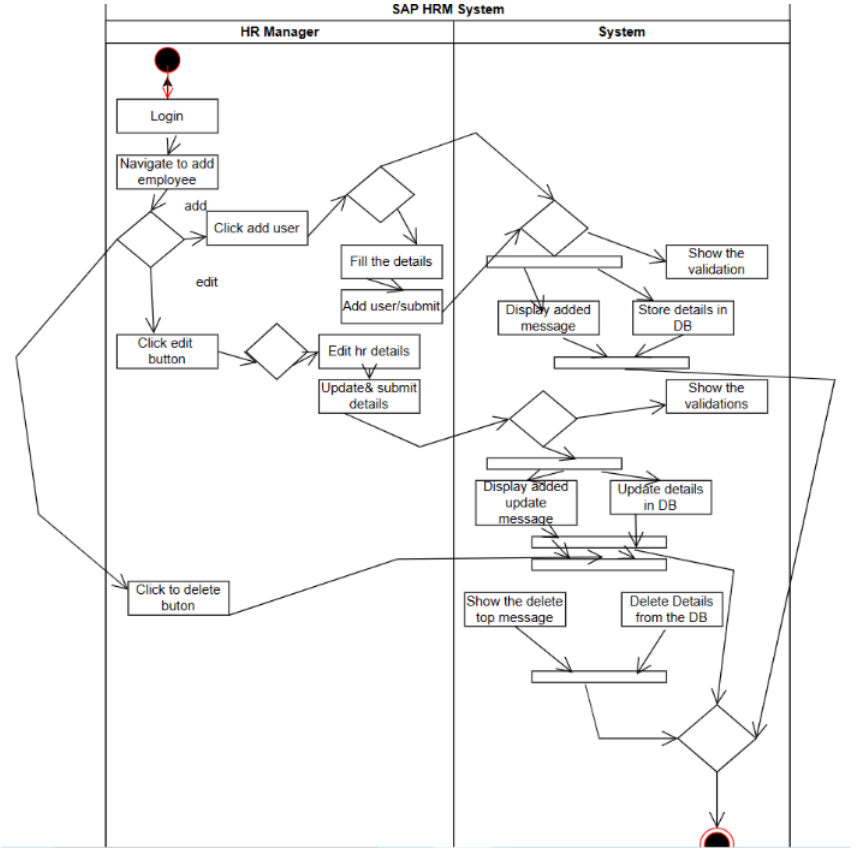
Class Diagram of Human Resource Management System



ER Diagram of Human Resource Management System

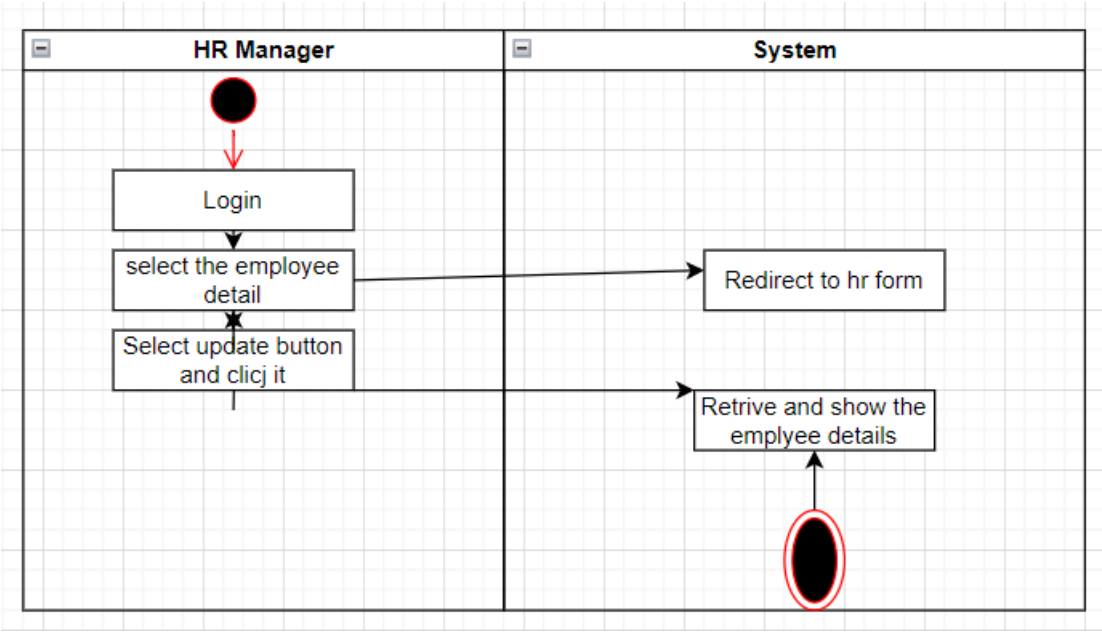


Activity Diagram of Human Resource Management System

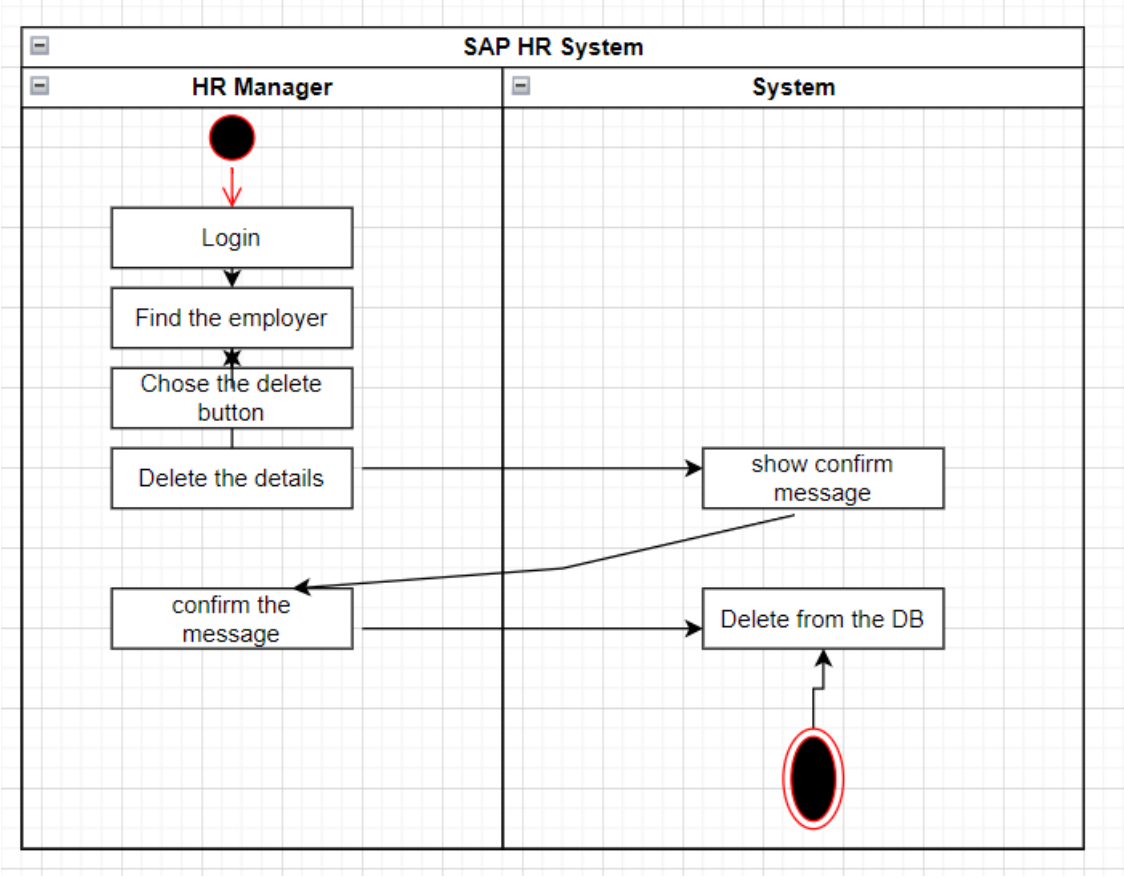


Activity Diagram with CRUD Actions

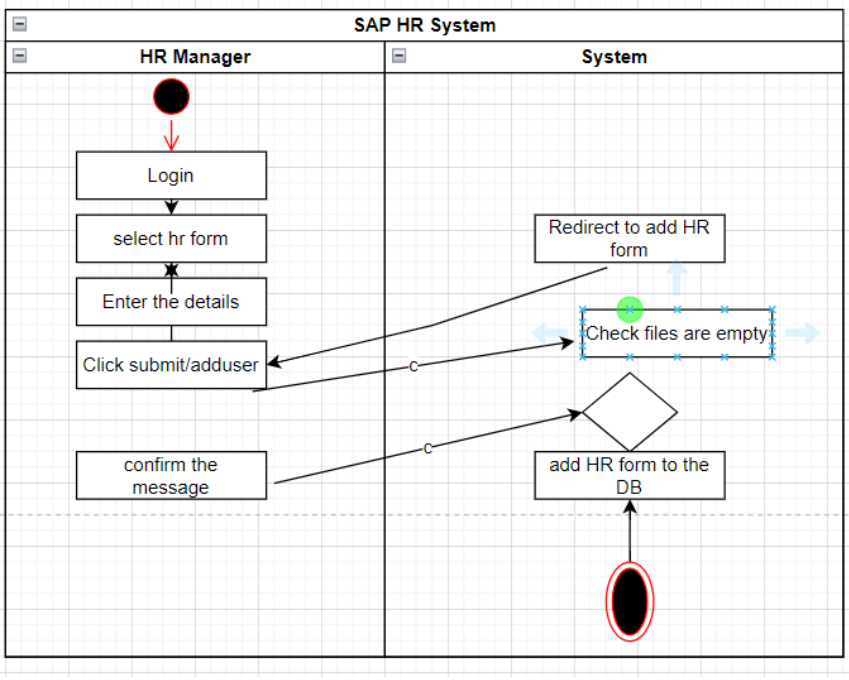
Reterive



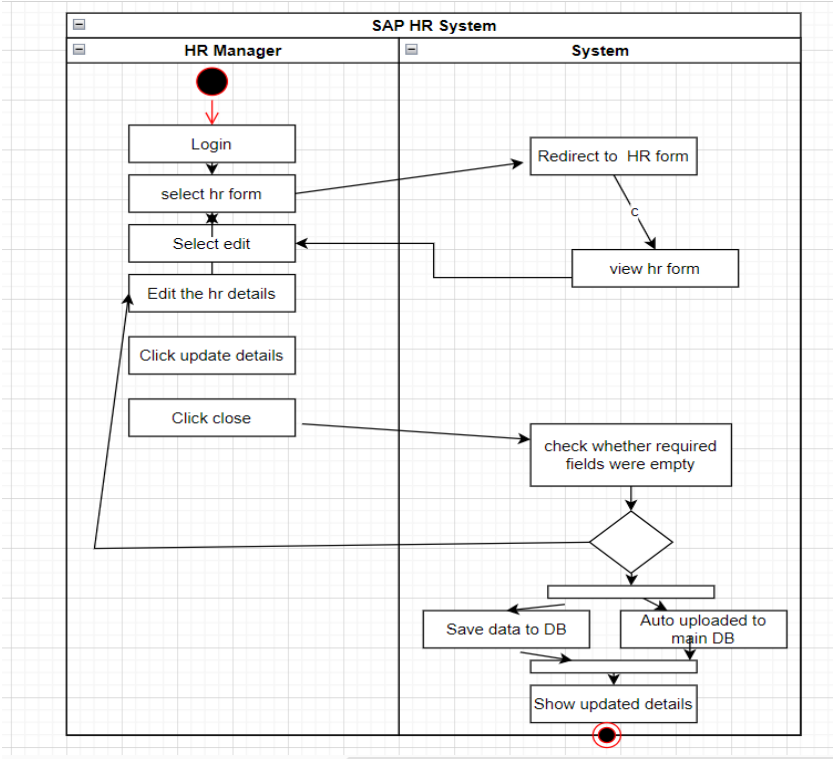
Delete


SAP HR System
HR Manager
System
Login
Find the employer
Chose the delete button
Delete the details
show confirm message
confirm the message
Delete from the DB

Add


SAP HR System
HR Manager
System
Login
select hr form
Enter the details
Click submit/adduser
Redirect to add HR form
Check files are empty
c
confirm the message
c
add HR form to the DB

Edit


SAP HR System
HR Manager
System
Login
select hr form
Select edit
Edit the hr details
Click update details
Click close
Redirect to HR form
c
view hr form
check whether required fields were empty
Save data to DB
Auto uploaded to main DB
Show updated details

Inventory Management System

(IT20012892 – Ahamed M.S.A)

This function manages inventory through a software application that enhances inventory management for businesses. It enables users to add, view, update, and remove product information, with unique names for easy tracking. When stock levels reach zero, the system generates email alerts for administrators to notify users. Additionally, it includes a search bar for filtering through inventory, allowing for quick and targeted searches. The function also offers the capability to download reports in Excel format, providing valuable data insights for informed decision-making.

System Actor

Inventory Manager

The Inventory manager plays a vital role in overseeing all inventory-related operations within the business. With the ability to add, edit, view, and delete inventory, they have full control over maintaining accurate and up-to-date inventory records. The manager also benefits from accessing comprehensive inventory details that are precisely calculated by the system, ensuring accurate stock levels and efficient inventory management. By effectively utilizing the system's features, the Inventory manager can make informed decisions, optimize stock levels, and ensure smooth inventory operations for the business.

Functional and non-functional requirements of the Inventory Management

Functional Requirements

- Inventory manager should be able to add inventory.
- Inventory manager should be able to edit inventory.
- Inventory manager should be able to delete inventory.
- When Inventory are recorded, the system records:
    - Name
    - Category
    - Price
    - Quantity
    - Description
    - Date
- The title, category, description, type and date must not be left empty.
- The quantity entered must not be negative, empty
- The Price entered must not be negative, empty, zero.
- Information relating to inventory on the system can be downloaded in PDF form.
- The system can compute :
    - Total no of products.
    - Total no of categories.
    - Total no of products out of stock.
    - Total item value.

The above mentioned data is displayed on the inventory dashboard page

Non-Functional Requirements

1)Performance

•The response and processing time of each interface are within a few minutes.

•Website must load quickly as possible.

2)Security

•Only admin can generate the report.

•If customer wishes website and system must allow option to logout.

•Registered customers have to login to the website by providing their username and password.

3)Safety

•Each user has unique username and password to login and access their accounts on the website.

•If customers forget their account login password, he can contact the administration through the message provided in the homepage. Accessible link will be resent to the user so by clicking on it the password can be reset.

4)Software Quality Attributes

•Availability - The system is available for all authorized customers and staffs to access when needed (24x7).

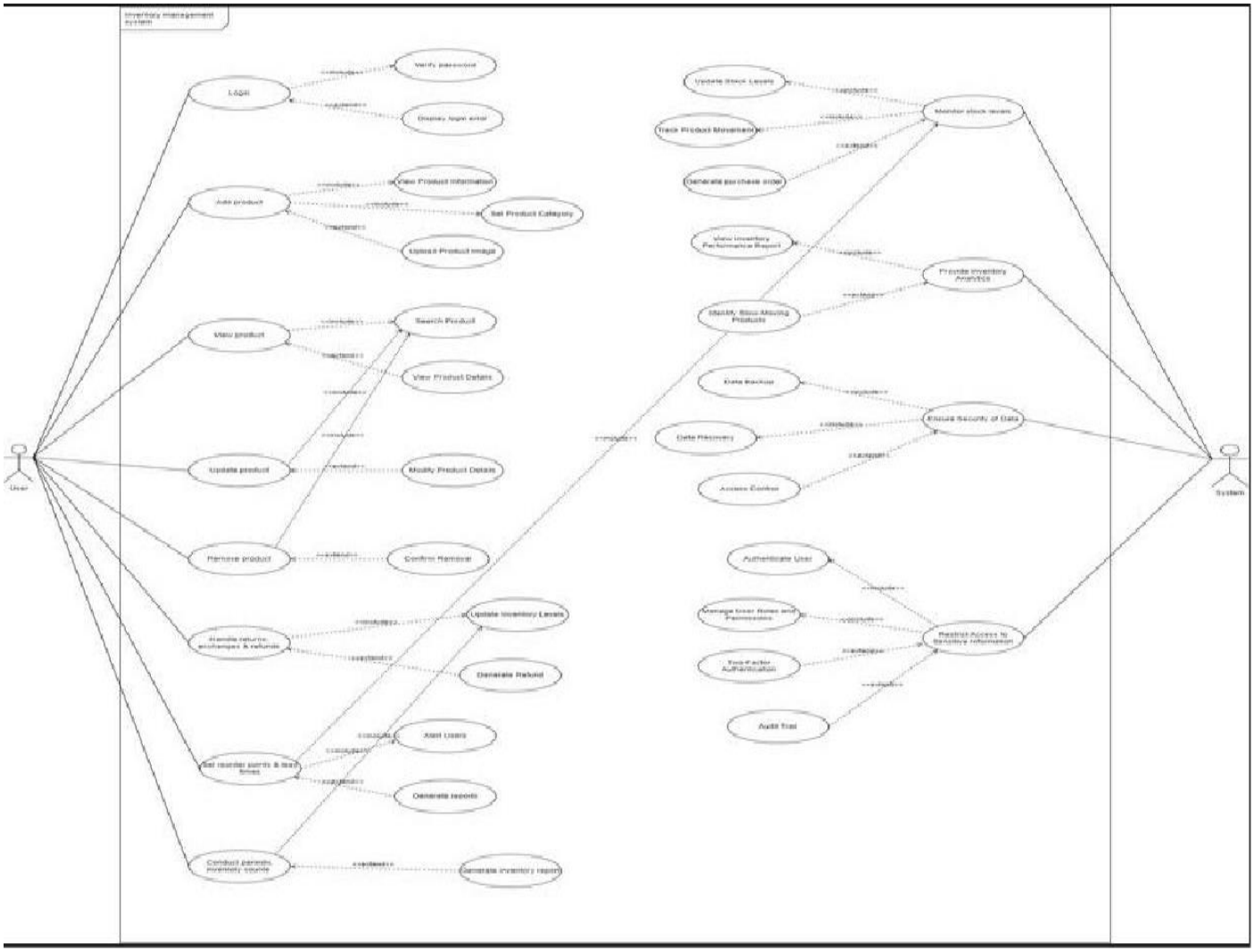
•System Maintenance - If any faults are found within the system it should be noted by the staff members and forwarded to the development team.

•Reliability - Development team continues to monitor how the system is performing and continue to make improvements to the system through updates.

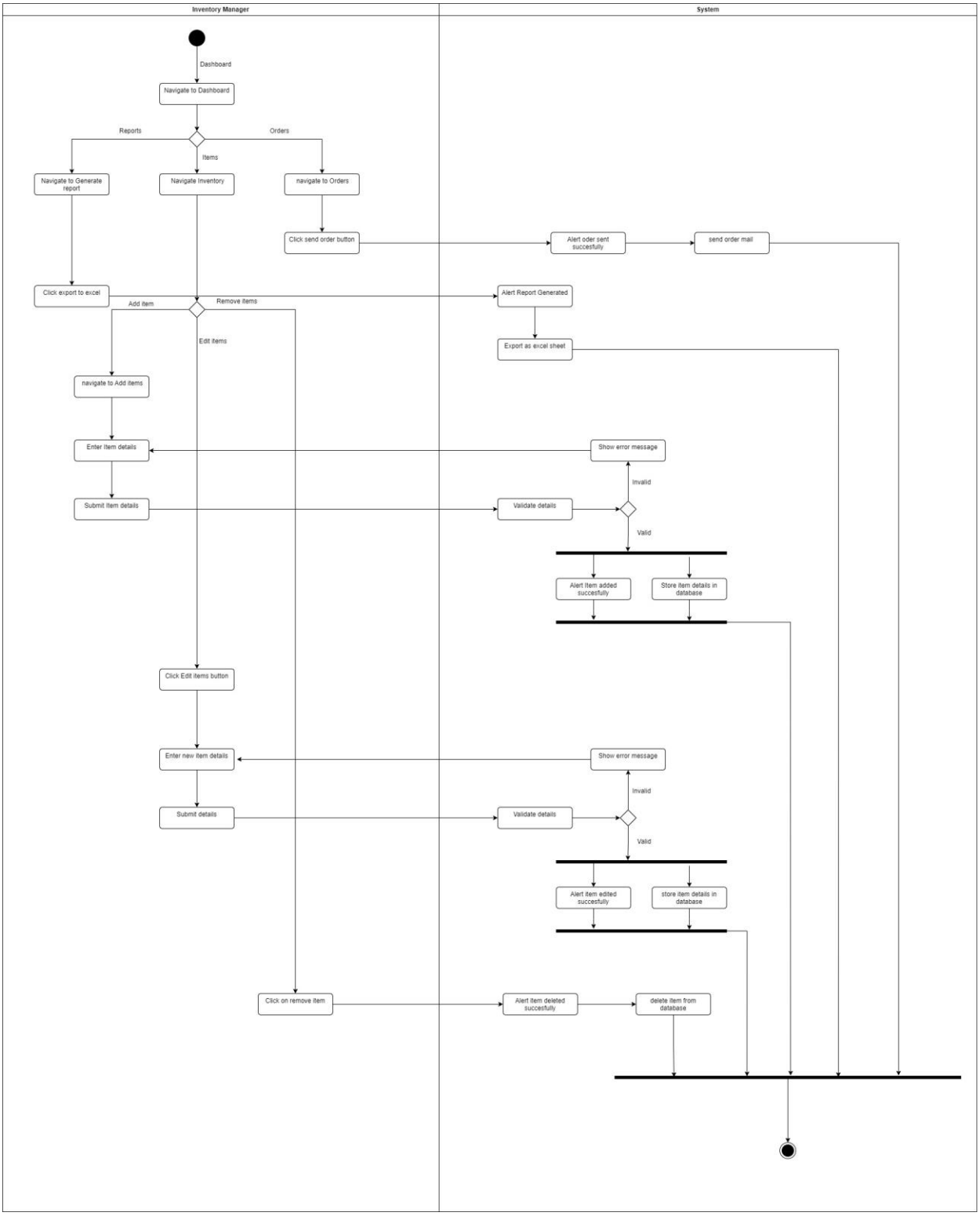
•Usability – User can easily understand the system and functions and operate them. Interfaces should be user friendly.

•Accuracy - All the managements function with expected response

User case Diagram

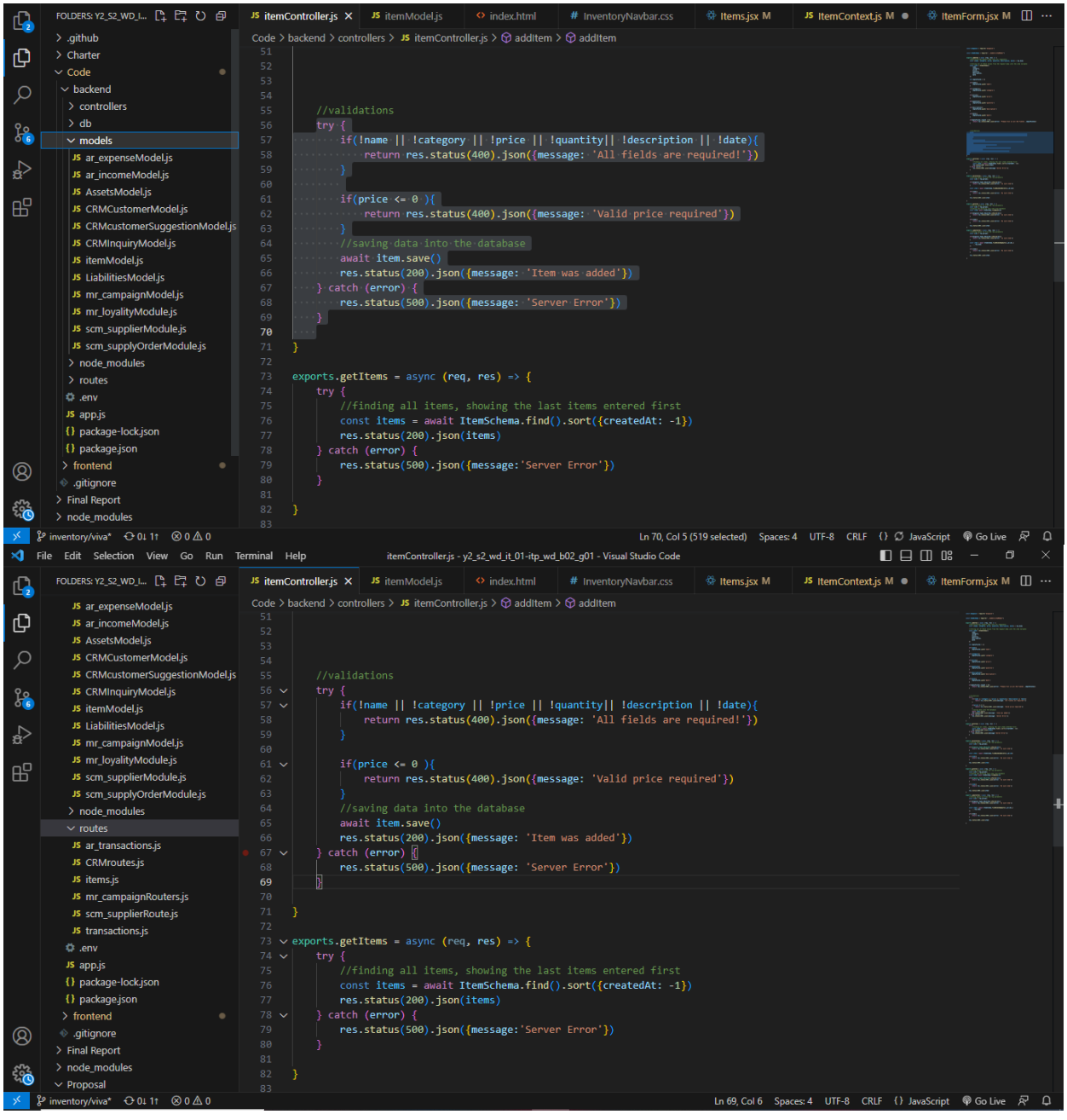


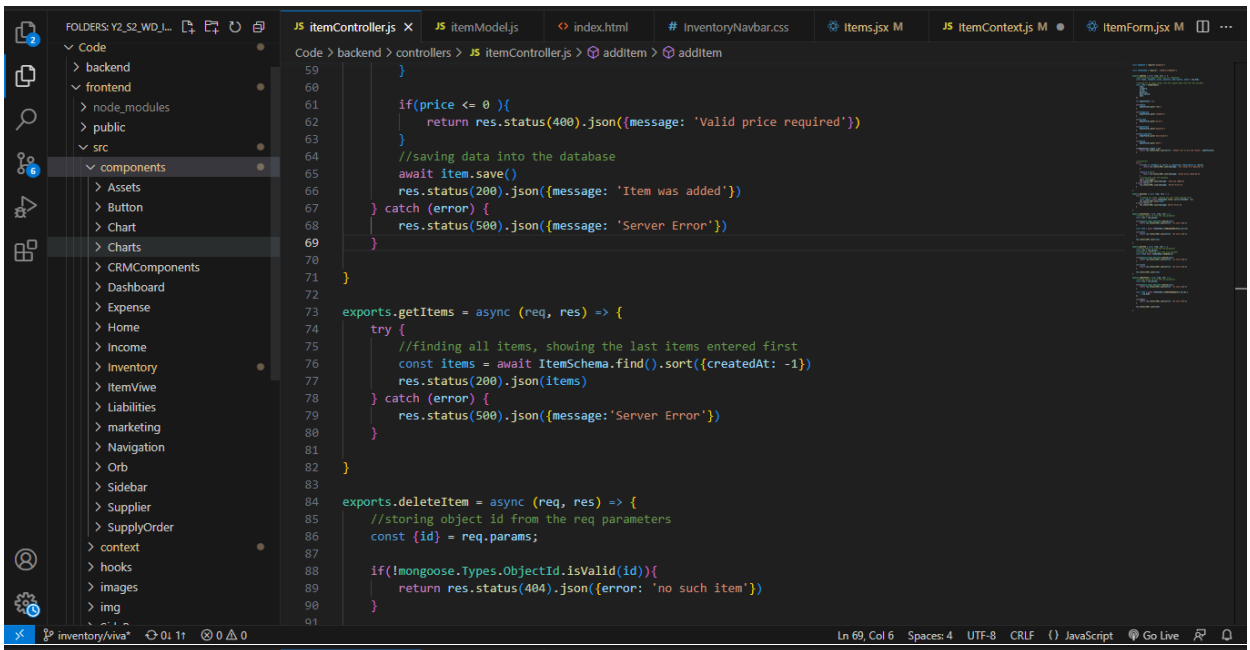
Activity Diagram

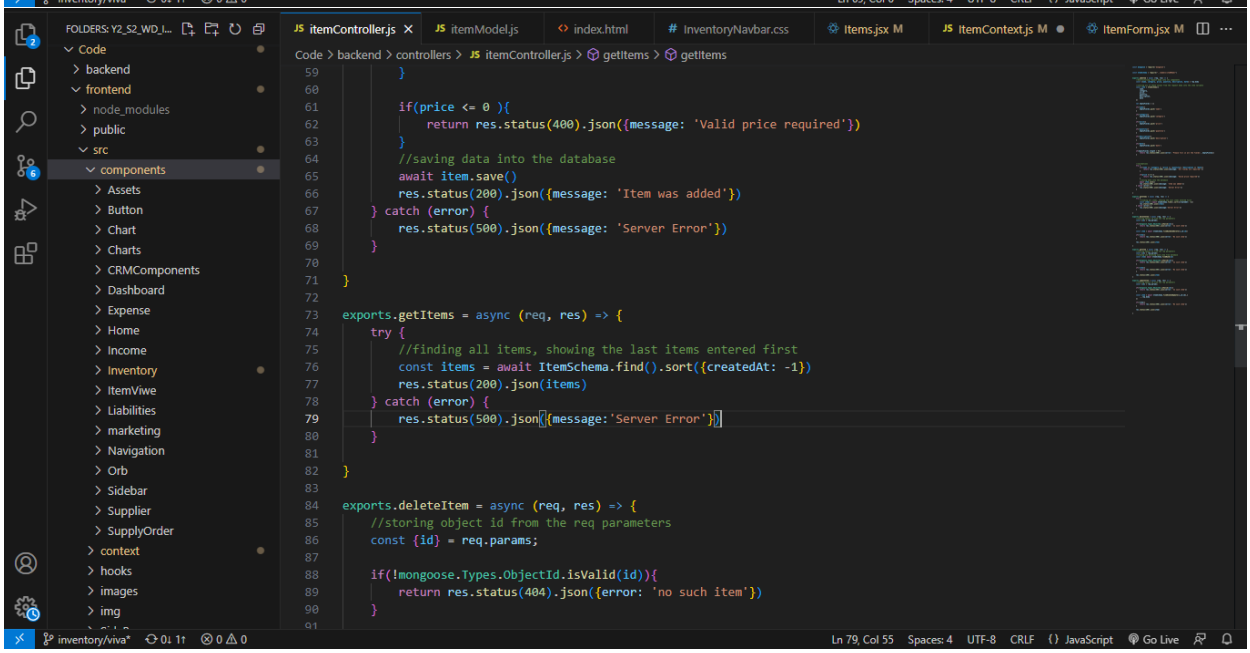


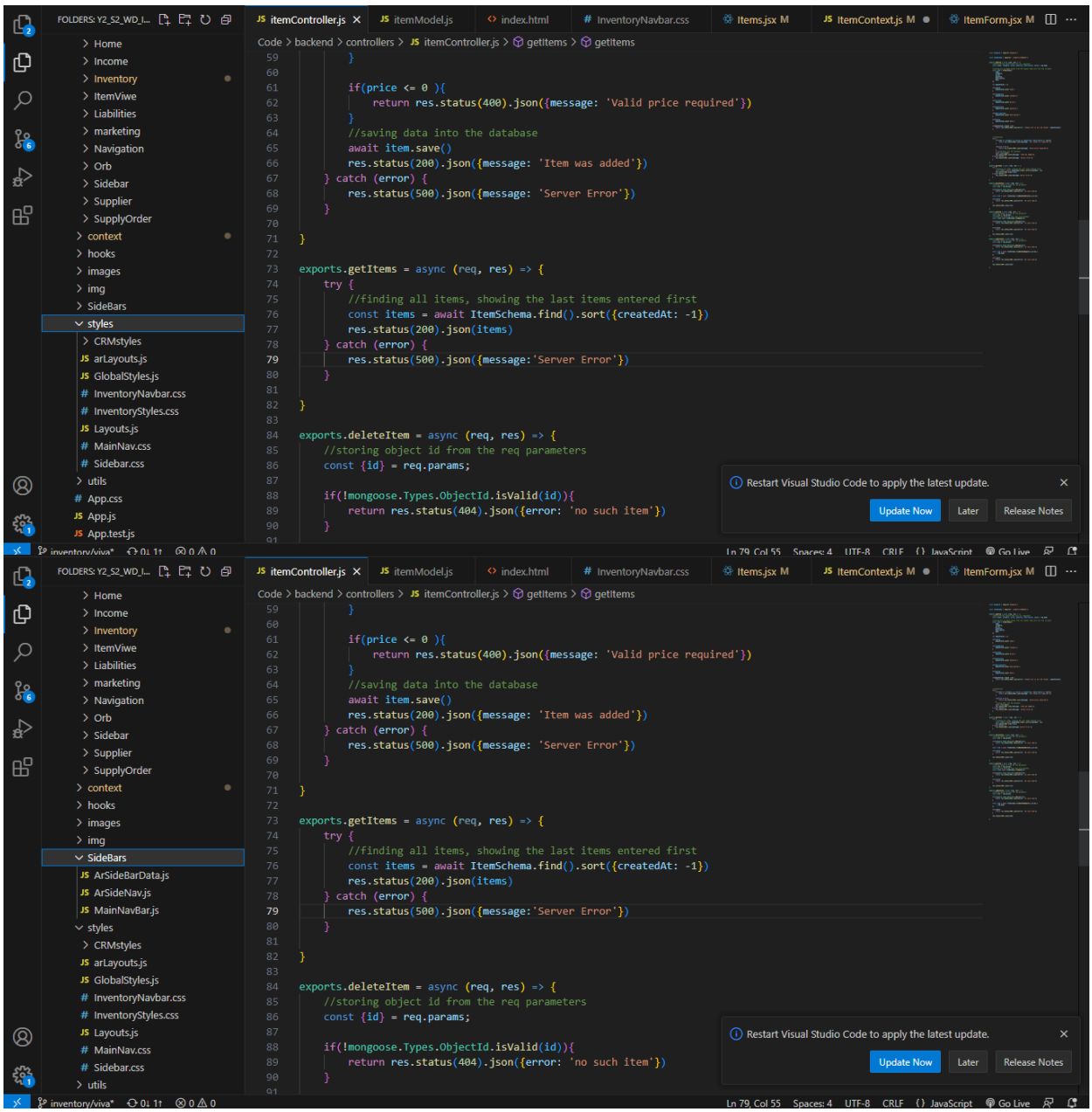
Design and Development

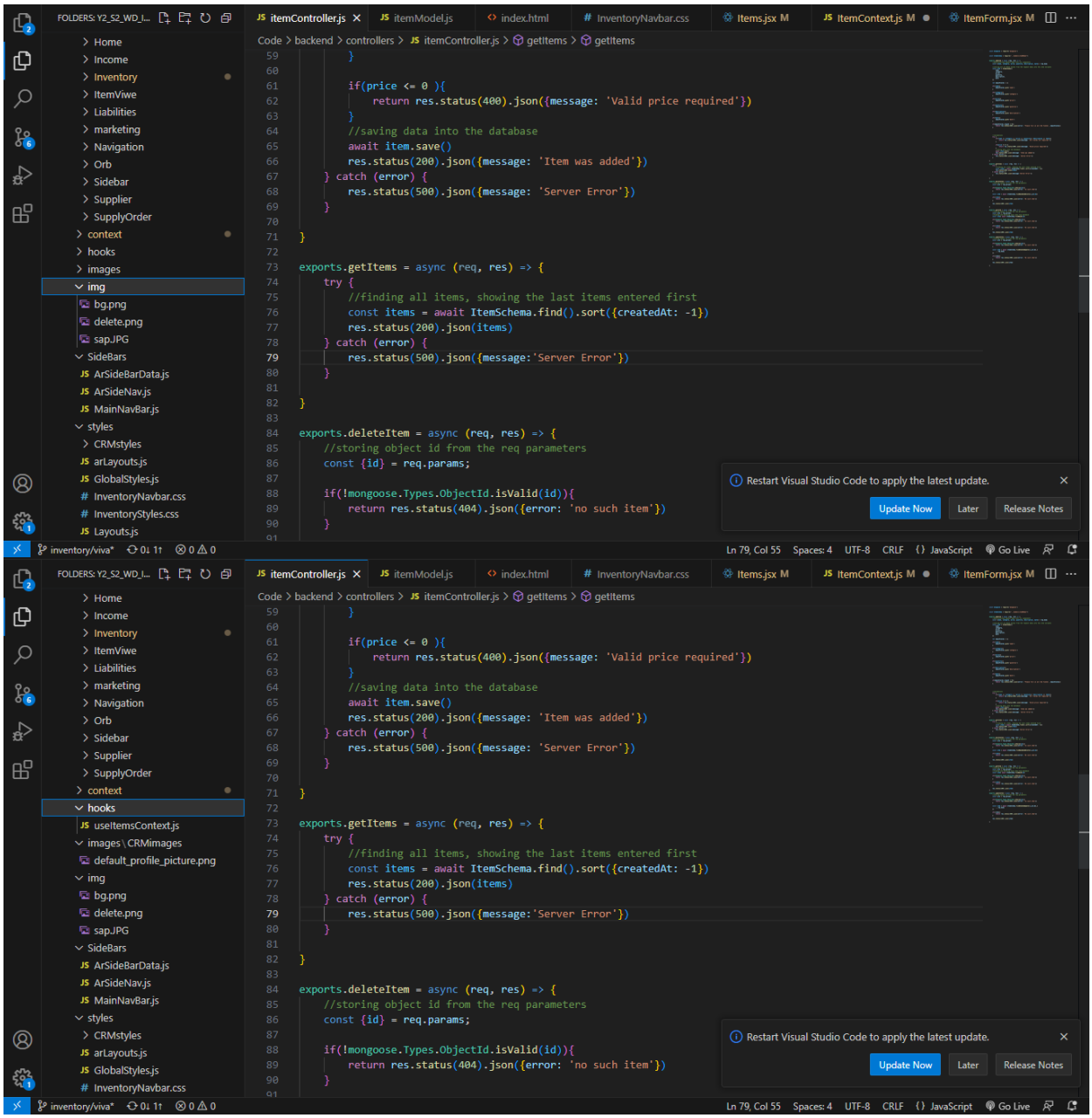
Implementation

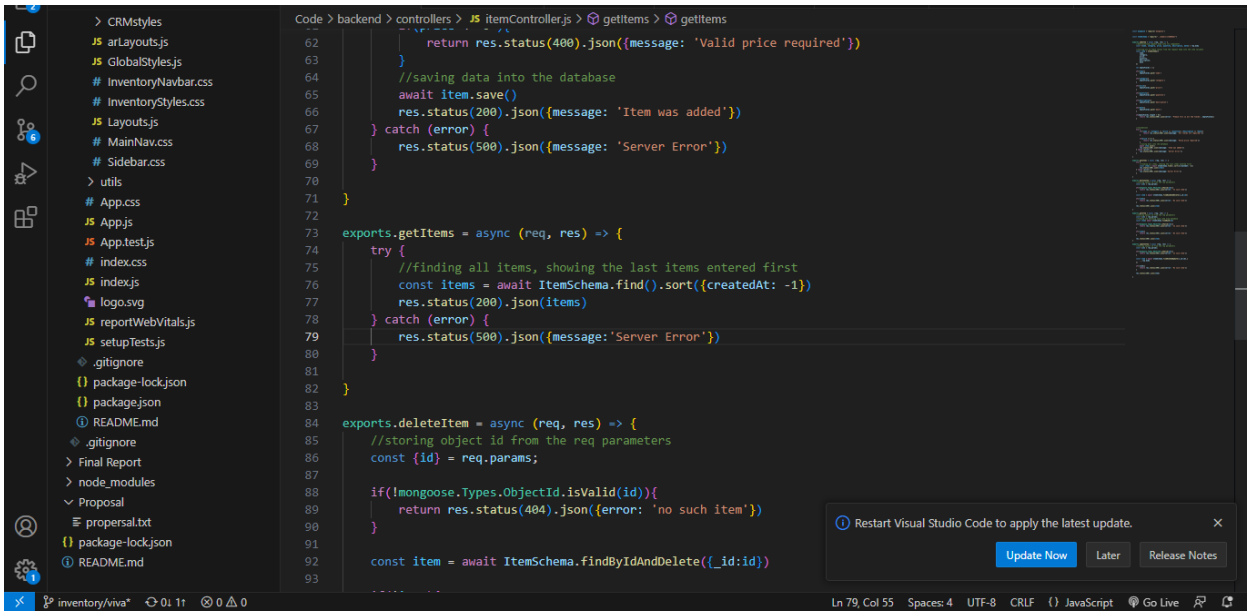



itemController.js
itemModel.js
index.html
InventoryNavbar.css
Items.jsx M
ItemContext.js M
ItemForm.jsx M
Code > backend > controllers > itemController.js > addItem > addItem
FOLDERS: Y2_S2_WD_I...
Code
backend
frontend
node_modules
public
src
components
Assets
Button
Chart
Charts
CRMComponents
Dashboard
Expense
Home
Income
Inventory
ItemViwe
Liabilities
marketing
Navigation
Orb
Sidebar
Supplier
SupplyOrder
context
hooks
images
img
}
if(price <= 0){
return res.status(400).json({message: 'Valid price required'})
}
//saving data into the database
await item.save()
res.status(200).json({message: 'Item was added'})
} catch (error) {
res.status(500).json({message: 'Server Error'})
}
}
exports.getItems = async (req, res) => {
try {
//finding all items, showing the last items entered first
const items = await ItemSchema.find().sort({createdAt: -1})
res.status(200).json(items)
} catch (error) {
res.status(500).json({message:'Server Error'})
}
}
exports.deleteItem = async (req, res) => {
//storing object id from the req parameters
const {id} = req.params;
if(!mongoose.Types.ObjectId.isValid(id)){
return res.status(404).json({error: 'no such item'})
}
inventory/viva*
Ln 69, Col 6
Spaces: 4
UTF-8
CRLF
JavaScript
Go Live


itemController.js
itemModel.js
index.html
InventoryNavbar.css
Items.jsx M
ItemContext.js M
ItemForm.jsx M
Code > backend > controllers > itemController.js > getItems > getItems
FOLDERS: Y2_S2_WD_I...
Code
backend
frontend
node_modules
public
src
components
Assets
Button
Chart
Charts
CRMComponents
Dashboard
Expense
Home
Income
Inventory
ItemViwe
Liabilities
marketing
Navigation
Orb
Sidebar
Supplier
SupplyOrder
context
hooks
images
img
}
if(price <= 0){
return res.status(400).json({message: 'Valid price required'})
}
//saving data into the database
await item.save()
res.status(200).json({message: 'Item was added'})
} catch (error) {
res.status(500).json({message: 'Server Error'})
}
}
exports.getItems = async (req, res) => {
try {
//finding all items, showing the last items entered first
const items = await ItemSchema.find().sort({createdAt: -1})
res.status(200).json(items)
} catch (error) {
res.status(500).json({message:'Server Error'})
}
}
exports.deleteItem = async (req, res) => {
//storing object id from the req parameters
const {id} = req.params;
if(!mongoose.Types.ObjectId.isValid(id)){
return res.status(404).json({error: 'no such item'})
}
inventory/viva*
Ln 79, Col 55
Spaces: 4
UTF-8
CRLF
JavaScript
Go Live


itemController.js
itemModel.js
index.html
InventoryNavbar.css
Items.jsx M
ItemContext.js M
ItemForm.jsx M
Code > backend > controllers > itemController.js > getItems > getItems
Restart Visual Studio Code to apply the latest update.
Update Now
Later
Release Notes


itemController.js
itemModel.js
index.html
InventoryNavbar.css
Items.jsx M
ItemContext.js M
ItemForm.jsx M
Code > backend > controllers > itemController.js > getItems > getItems
Restart Visual Studio Code to apply the latest update.
Update Now
Later
Release Notes
Ln 79, Col 55
Spaces: 4
UTF-8
CRLF
JavaScript
Go Live

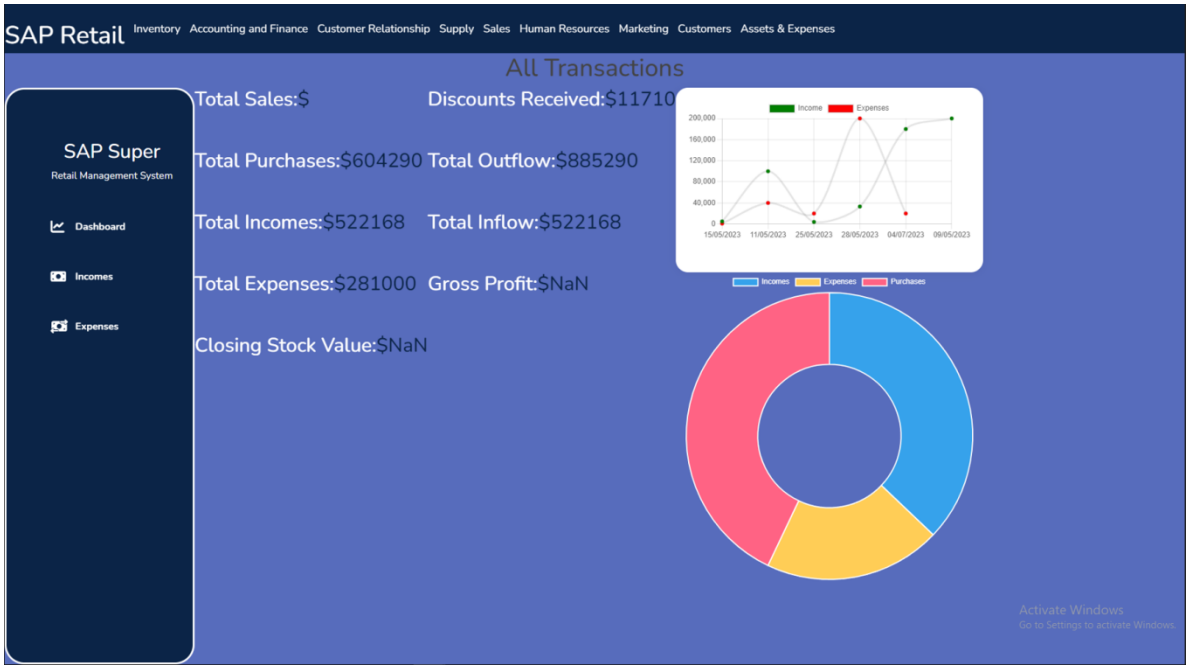


Dependancies

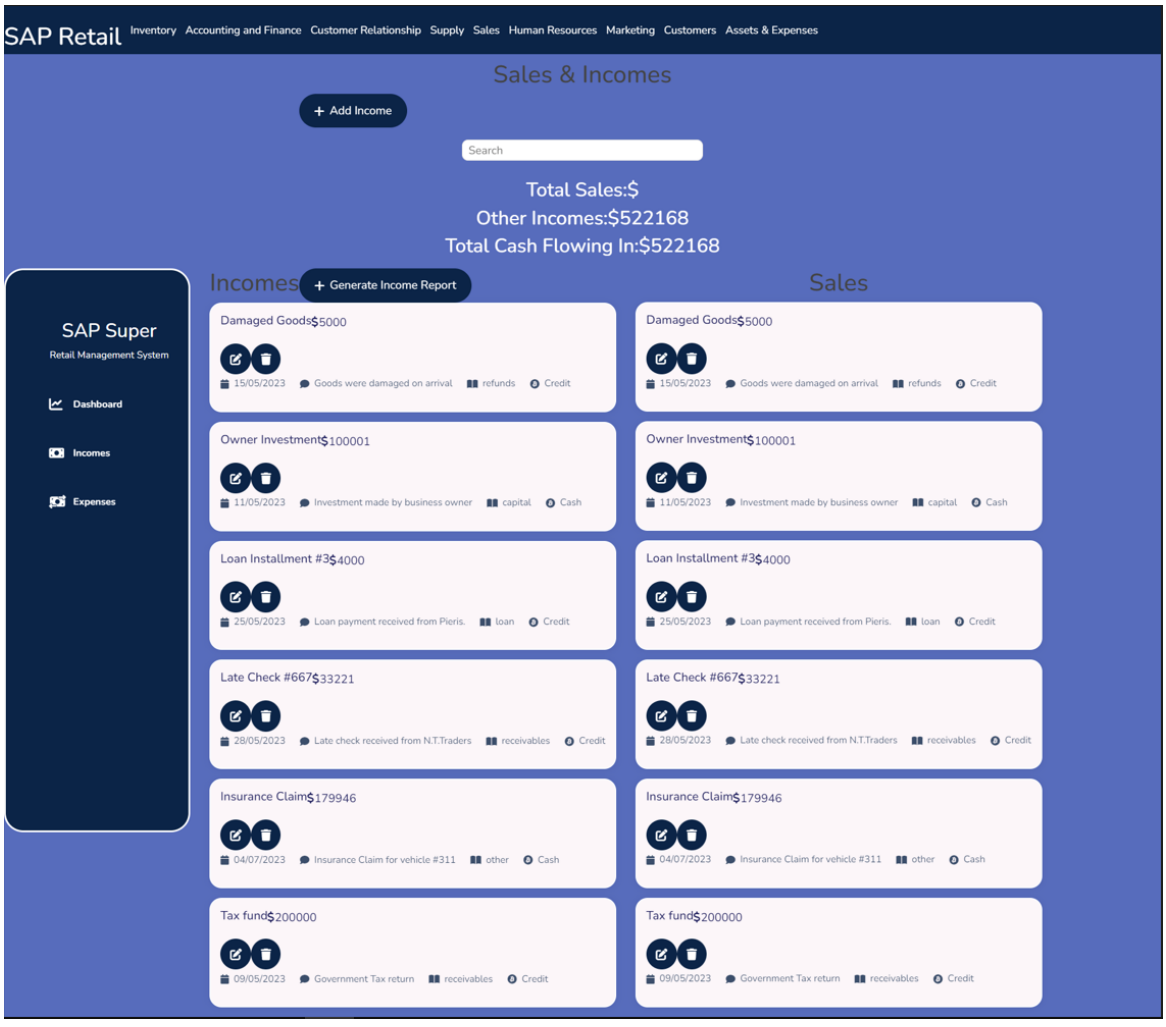
```
{
  "name": "backend",
  "version": "1.0.0",
  "description": "Backend for SAP Retail Management System",
  "main": "app.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "nodemon app.js"
  },
  "repository": {
    "type": "git",
    "url": "git+https://github.com/SLIITITP/y2_s2_wd_it_01-itp_wd_b02_g01.git"
  },
  "author": "Veynitha Bandara",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/SLIITITP/y2_s2_wd_it_01-itp_wd_b02_g01/issues"
  },
  "homepage": "https://github.com/SLIITITP/y2_s2_wd_it_01-itp_wd_b02_g01#readme",
  "dependencies": {
    "cors": "^2.8.5",
    "dotenv": "^16.0.3",
    "express": "^4.18.2",
    "fast-csv": "^4.3.6",
    "json2csv": "^6.0.0-alpha.2",
    "mongoose": "^7.0.3",
    "multer": "^1.4.5-lts.1",
    "nodemailer": "^6.9.1",
    "nodemailer-sendgrid-transport": "^0.2.0",
    "nodemon": "^2.0.22",
    "uuid": "^9.0.0"
  },
  "keywords": []
}
```

Accounting and Finance Management System

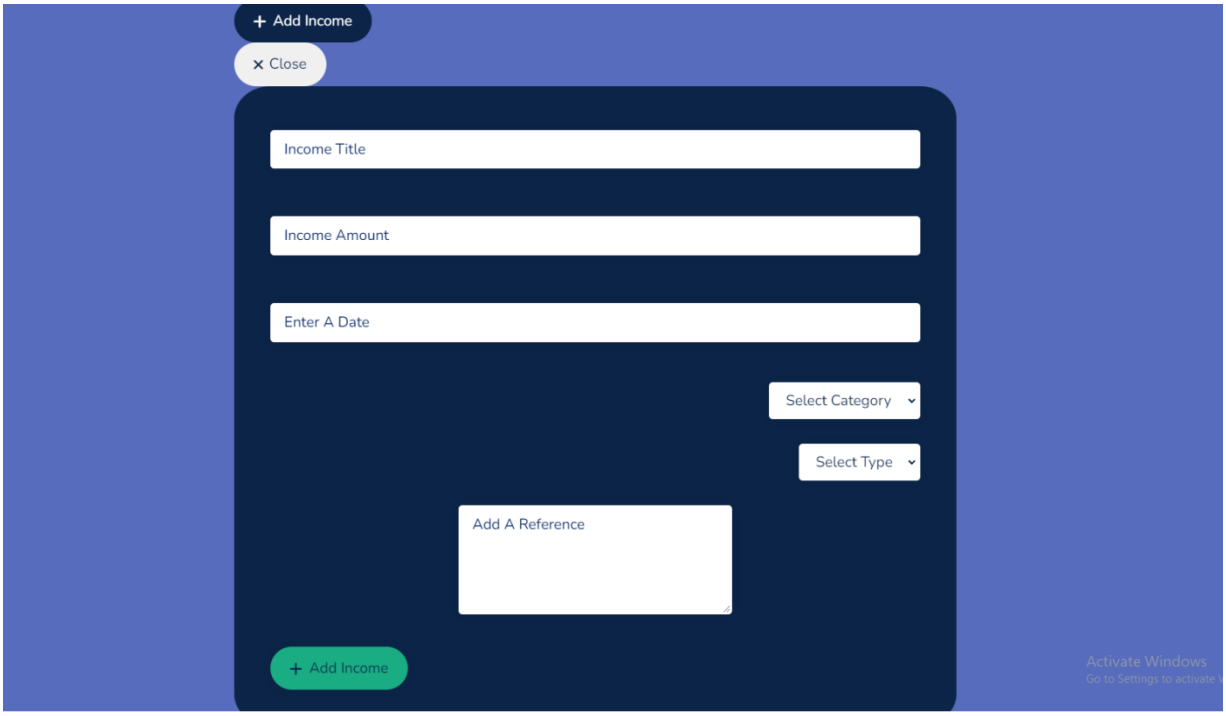
Accounting and Finance Dashboard



Sales and Income UI

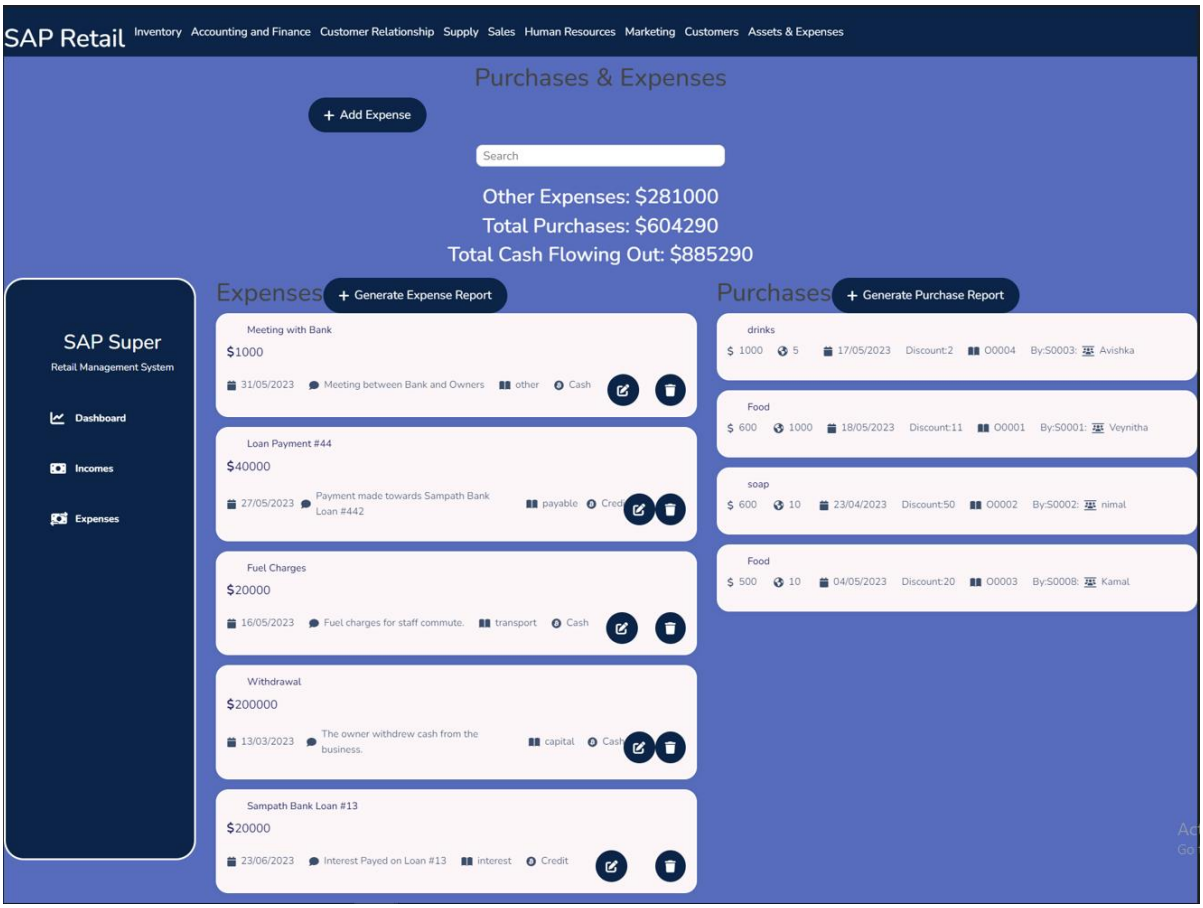


Add Income Form

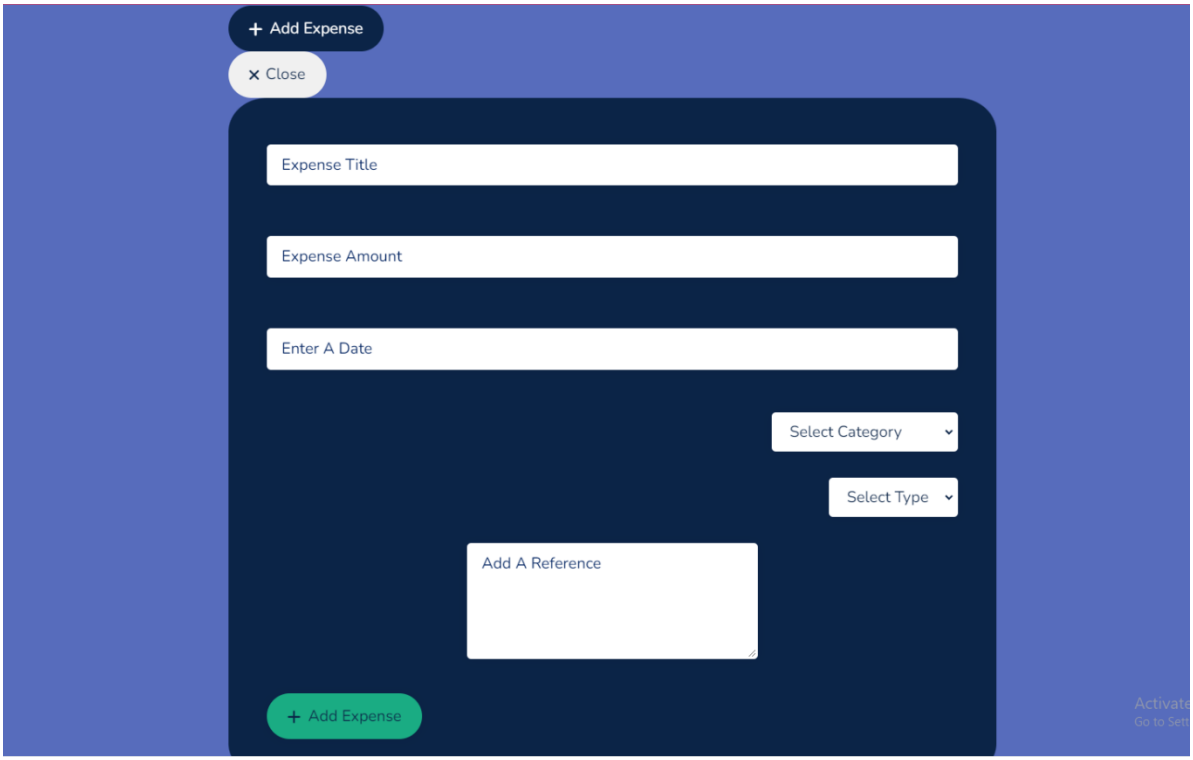


Edit Income Form

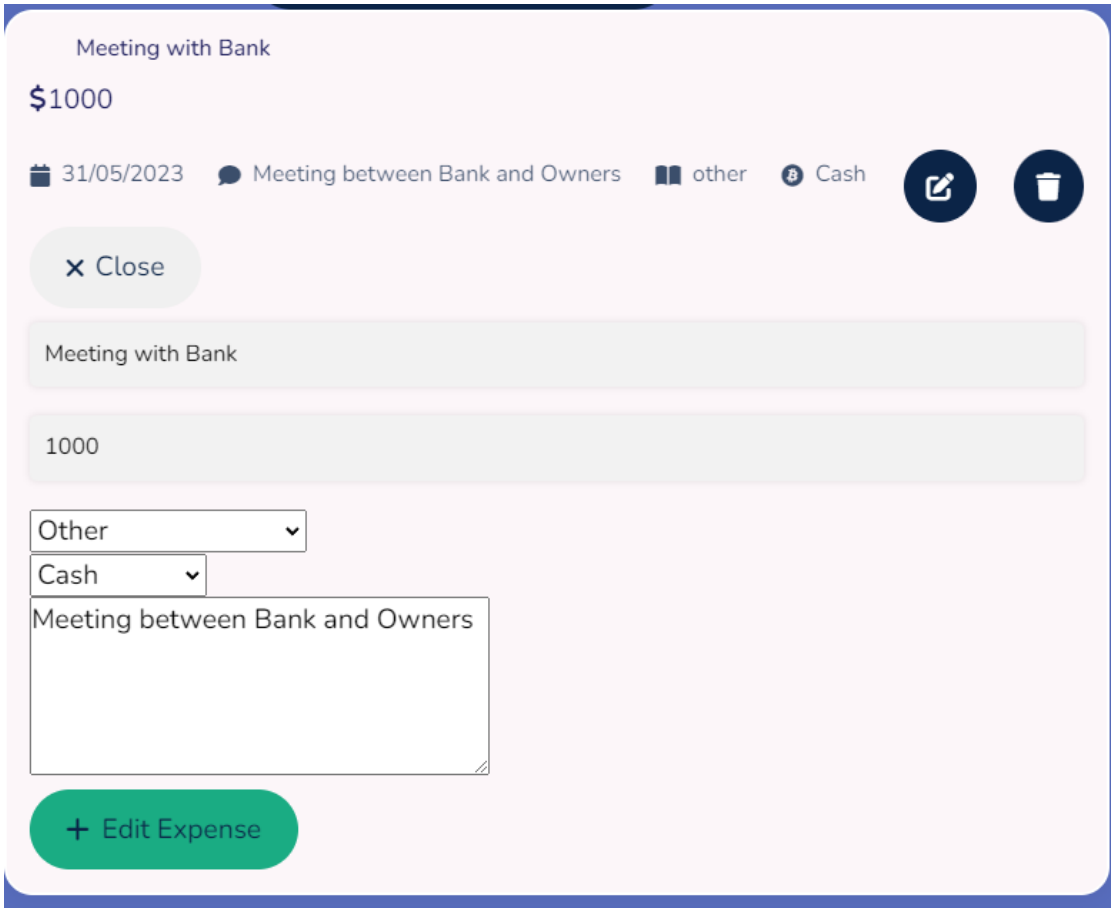
Expenses and Purchases UI



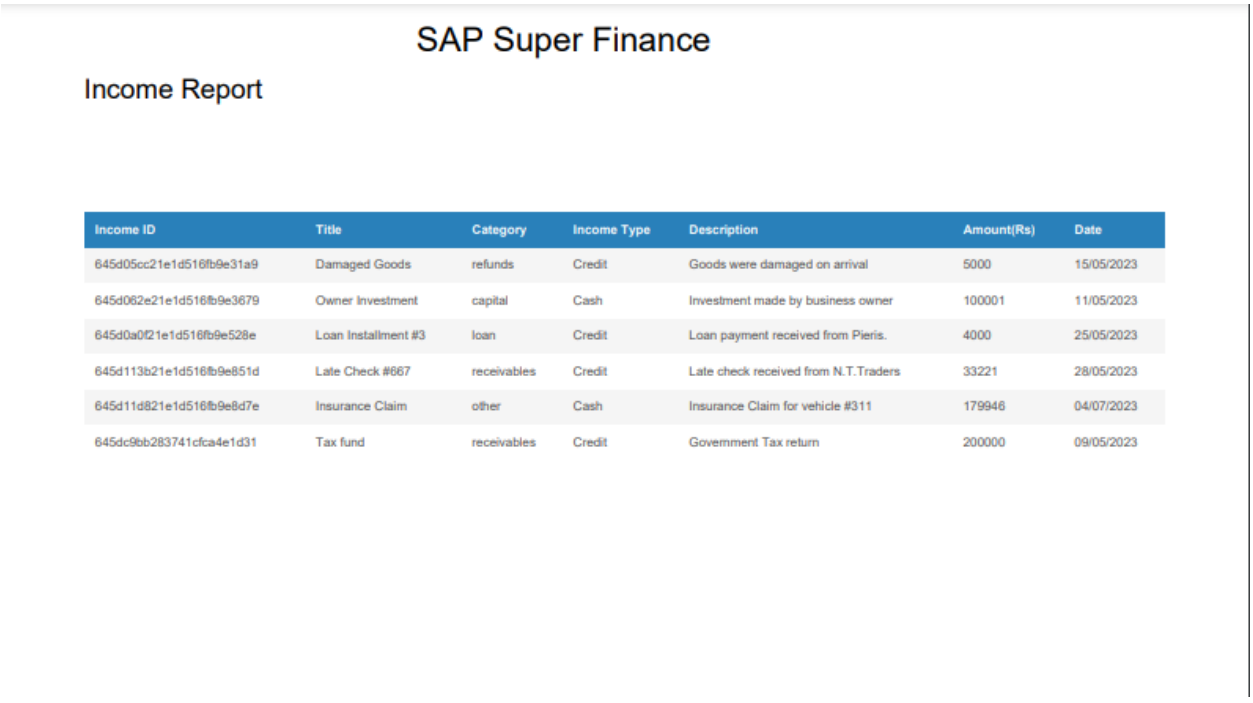
Add Expense Form



Edit Expense Form



Income Report



SAP Super Finance

Income Report

Income ID	Title	Category	Income Type	Description	Amount(Rs)	Date
645d05cc21e1d516fb9e31a9	Damaged Goods	refunds	Credit	Goods were damaged on arrival	5000	15/05/2023
645d062e21e1d516fb9e3679	Owner Investment	capital	Cash	Investment made by business owner	100001	11/05/2023
645d0a0f21e1d516fb9e528e	Loan Installment #3	loan	Credit	Loan payment received from Pieris.	4000	25/05/2023
645d113b21e1d516fb9e851d	Late Check #667	receivables	Credit	Late check received from N.T.Traders	33221	28/05/2023
645d11d821e1d516fb9e8d7e	Insurance Claim	other	Cash	Insurance Claim for vehicle #311	179946	04/07/2023
645dc9bb283741cfca4e1d31	Tax fund	receivables	Credit	Government Tax return	200000	09/05/2023

Expense Report

SAP Super Finance

Expense Report

Expense ID	Title	Category	Expense Type	Description	Amount(Rs)	Date
645cc23bc90b9f9554b4f309	Meeting with Bank	other	Cash	Meeting between Bank and Owners	1000	31/05/2023
645d167321e1d516fb9f0248	Loan Payment #44	payable	Credit	Payment made towards Sampath Bank Loan #442	40000	27/05/2023
645d170221e1d516fb9f1200	Fuel Charges	transport	Cash	Fuel charges for staff commute.	20000	16/05/2023
645d218421e1d516fba00507	Withdrawal	capital	Cash	The owner withdrew cash from the business.	200000	13/03/2023
645d21c121e1d516fba00b58	Sampath Bank Loan #13	interest	Credit	Interest Payed on Loan #13	20000	23/06/2023

Income Validations

Front End

```
const handleSubmit = e => {
    e.preventDefault()
    if(!title || !amount === 'number' || !type || !description || !category ||!date){
        alert("All fields required!!!")
    }
    else if( !amount || amount<=0 ){
        alert("Invalid Amount!!!")
    }
    else{
        addIncome(inputState)
    alert("Income Added")
    }
    setInputState({
        title: '',
        amount: '',
        date: '',
        category: '',
        description: '',
        type: ''
    })
}
```

Backend

```
//validations
try {

    if(!title || !category || !description || !type || !date){
        return res.status(400).json({message: 'All fields are required!'})
    }
    if(amount <= 0 || !amount === 'number'){
        return res.status(400).json({message: 'Valid amount requried'})
    }
    //saving data into the database
    await income.save()
    res.status(200).json({message: 'Income was added'})
} catch (error) {
    res.status(500).json({message: 'Server Error'})
}
```

Expense Validations

Front End

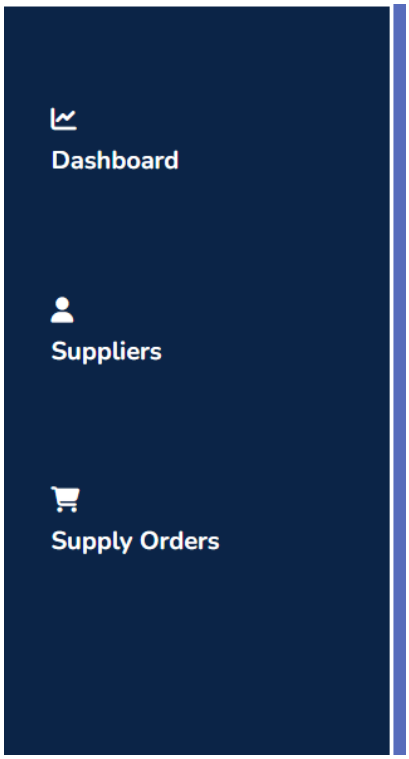
```
const handleSubmit = e => {
    e.preventDefault()
    if(!title || !amount === 'number' || !type || !description || !category ||!date){
        alert("All fields required!!!")
    }
    else if( !amount || amount<=0 ){
        alert("Invalid Amount!!!")
    }
    else{
        addExpense(inputState)
    alert("Expense Added")
    }
    setInputState({
        title: '',
        amount: '',
        date: '',
        category: '',
        description: '',
        type: ''
    })
}
```

Backend

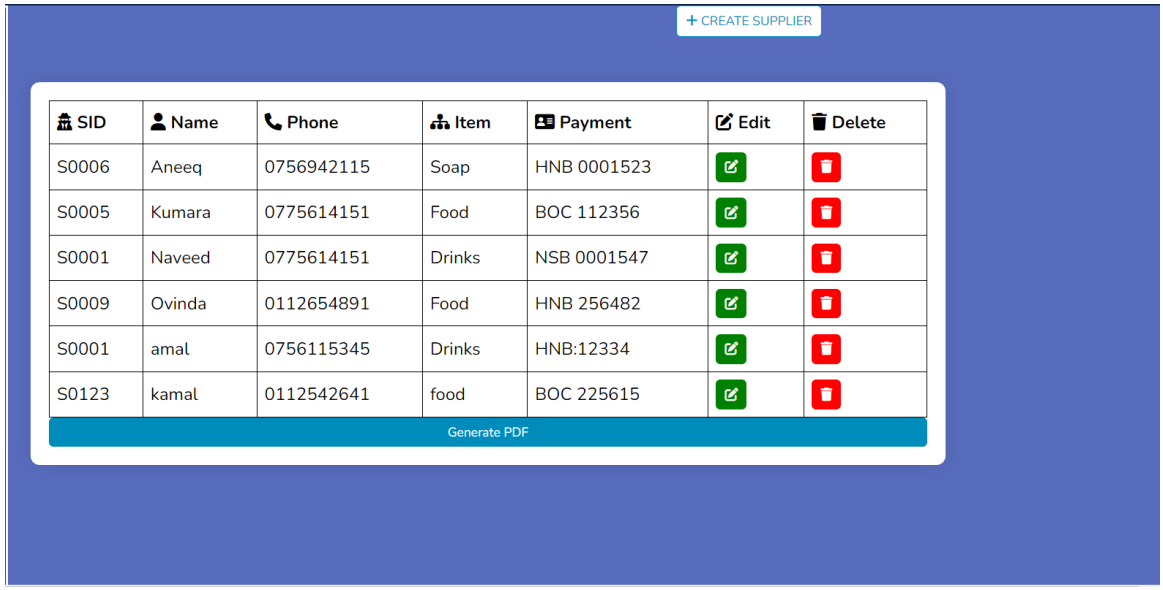
```
//validations
try {
    if(!title || !category || !description || !type || !date){
        return res.status(400).json({message: 'All fields are required!'})
    }
    if(amount <= 0 || !amount === 'number'){
        return res.status(400).json({message: 'Valid amount requried'})
    }
    //saving data into the database
    await expense.save()
    res.status(200).json({message: 'Expense was added'})
} catch (error) {
    res.status(500).json({message: 'Server Error'})
}
```

Supply Chain management System

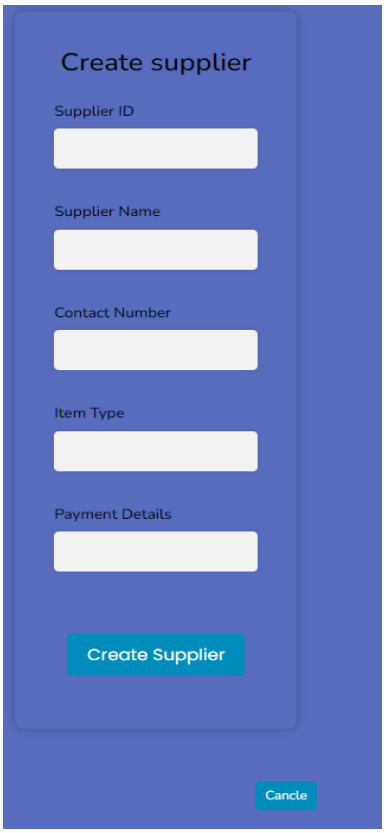
Supply chain management Side bar



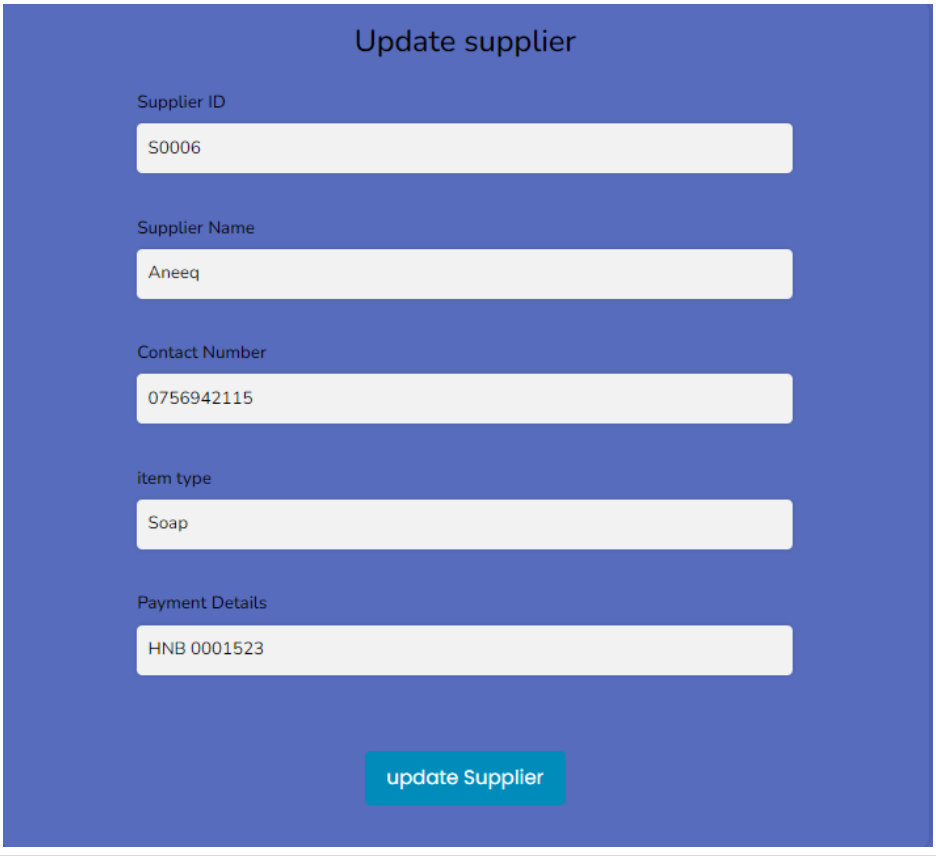
Supplier details



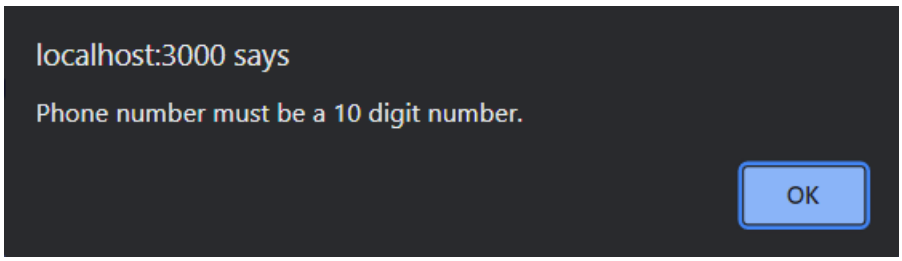
Create supplier Form



Update supplier



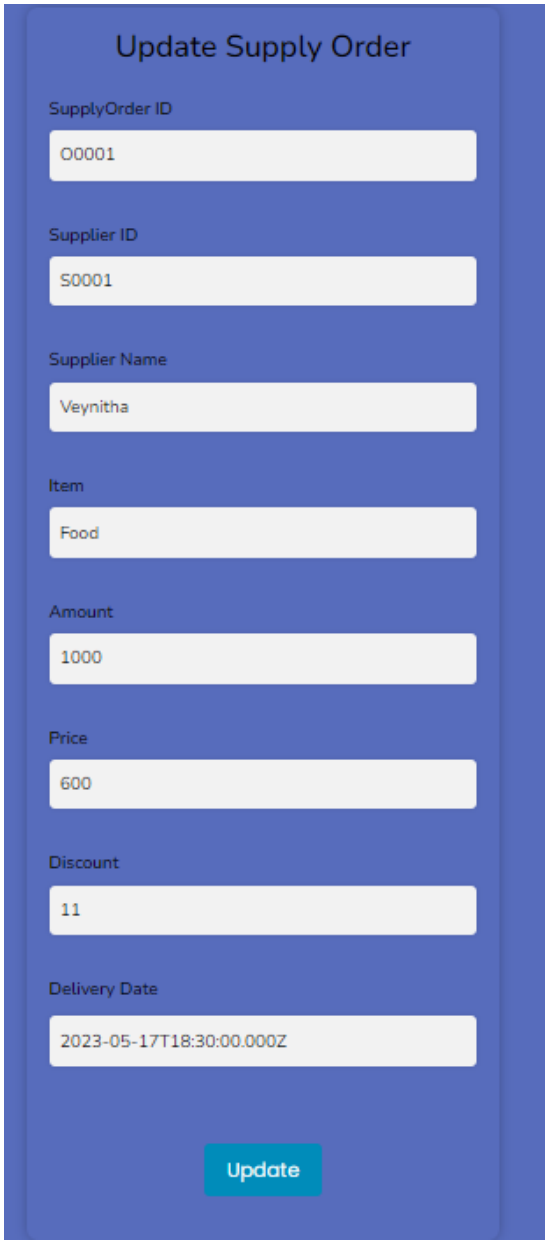
Validation messages



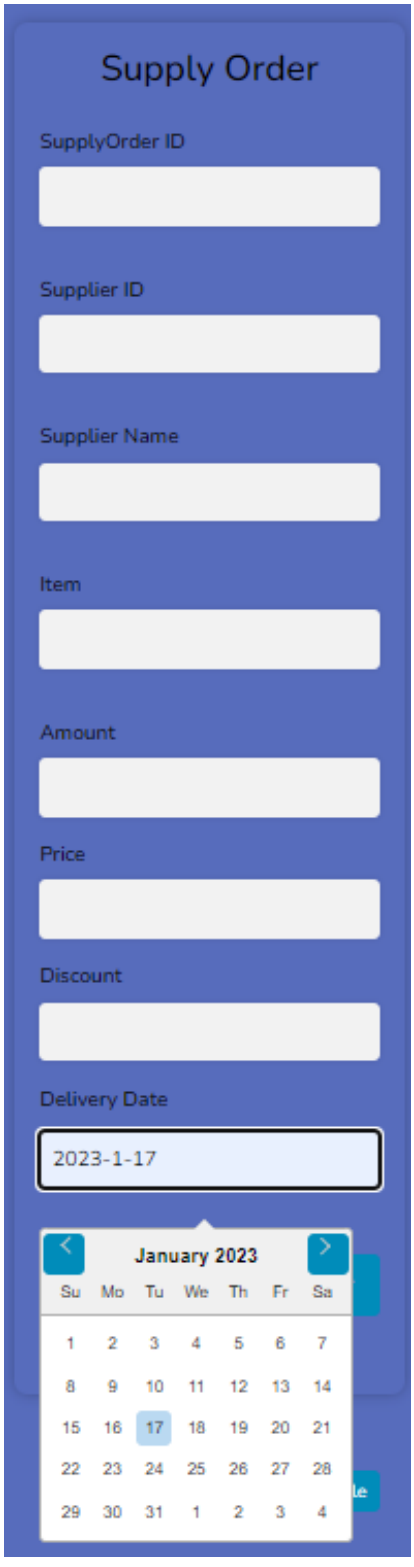
Genareted PDF

SID	Name	Phone	Item	Payment
S0006	Aneeq	0756942115	Soap	HNB 0001523
S0005	Kumara	0775614151	Food	BOC 112356
S0001	Naveed	0775614151	Drinks	NSB 0001547
S0009	Ovinda	0112654891	Food	HNB 256482
S0001	amal	0756115345	Drinks	HNB:12334
S0123	kamal	0112542641	food	BOC 225615

Update supply order



Date Selection



Val

```
},
createSupplier: async (e) => {
    e.preventDefault();
    const { createSupplierForm, supplierData } = SupplierStore.getState();

    //validations for inputs
    const { SID, supplierName, phone, itemType, paymentDetails } =
        createSupplierForm;
    if (!SID || !supplierName || !phone || !itemType || !paymentDetails) {
        alert("All fields are required.");
        return;
    } else if (isNaN(phone) || phone.length !== 10) {
        alert("Phone number must be a 10 digit number.");
        return;
    } else {
        const res = await axios.post(
            `${BASE_URL}addsupplier`,    "addsupplier": Unknown word.
            createSupplierForm
        );
        alert("Supplier Added");
        set({
            Supplier: [...supplierData, res.data],
            createSupplier: {
                SID: "",
                supplierName: "",
                phone: "",
                itemType: "",
                paymentDetails: "",
            },
        });
    }
    //create supplier
},
```

Pdf generation Code

```
const generatePDF = () => {
    const doc = new jsPDF();
    doc.setProperties({
        title: "Supplier List",
    });
    doc.autoTable({
        head: [["SID", "Name", "Phone", "Item", "Payment"]],
        body: store.supplierData.map((supplier) => [
            supplier.SID,
            supplier.supplierName,
            supplier.phone,
            supplier.itemType,
            supplier.paymentDetails,
        ]),
    });
    doc.save("supplier-list.pdf");
};
```

Sidebar

```jsx
function SideBar() {
    return (
        <div className="Sidebar" height="100%" width="250px" color="#FFFFFF">
            <ul className="SidebarList" height="100%" width="100%">
                {SideBarData.map((val, key) => {
                    return (
                        <li
                            key={key}
                            className="row"
                            onClick={() =>
                                (window.location.pathname = val.link)
                            }
                        >
                            <div className="icon">{val.icon}</div>
                            <div className="title">{val.title}</div>
                        </li>
                    );
                })}
            </ul>
        </div>
    );
}
export default SideBar;
```

```
nd > src > components > Sidebar > JS SideBarData.js > ...
import { dashboard, supplier, supplyorder } from "../../utils/Icons";    "supplyorder": Unknown word.

export const SideBarData = [
    {
        title: "Dashboard",
        icon: dashboard,
        link: "/",
    },
    {
        title: "Suppliers",
        icon: supplier,
        link: "/supplier",
    },
    {
        title: "Supply Orders",
        icon: supplyorder,    "supplyorder": Unknown word.
        link: "/supplyorder",    "supplyorder": Unknown word.
    },
];
```

Supply Order Validations

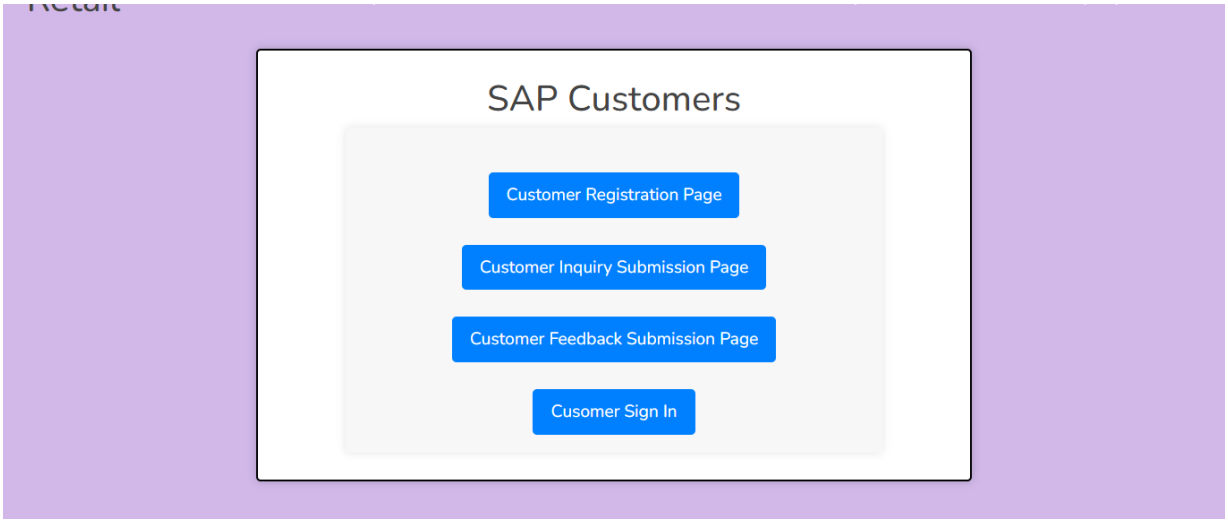
```
createSupplyOrder: async (e) => {
    e.preventDefault();
    const { createSupplyOrderForm, SupplyOrderData } =
        SupplyOrderStore.getState();
    const {
        orderID,
        SID,
        supplierName,
        item,
        amount,
        price,
        discount,
        deliverydate,    "deliverydate": Unknown word.
    } = createSupplyOrderForm;

    //validations for inputs
    if (
        !orderID ||
        !SID ||
        !supplierName ||
        !item ||
        !amount ||
        !price ||
        !discount ||
        !deliverydate    "deliverydate": Unknown word.
    ) {
        alert("All fields are required.");
        return;
    } else if (isNaN(price) || price <= 0) {
        alert("Invalid Price");
        return;
    } else if (isNaN(amount) || amount <= 0) {
        alert("Invalid Amount");
        return;
    } else {
```

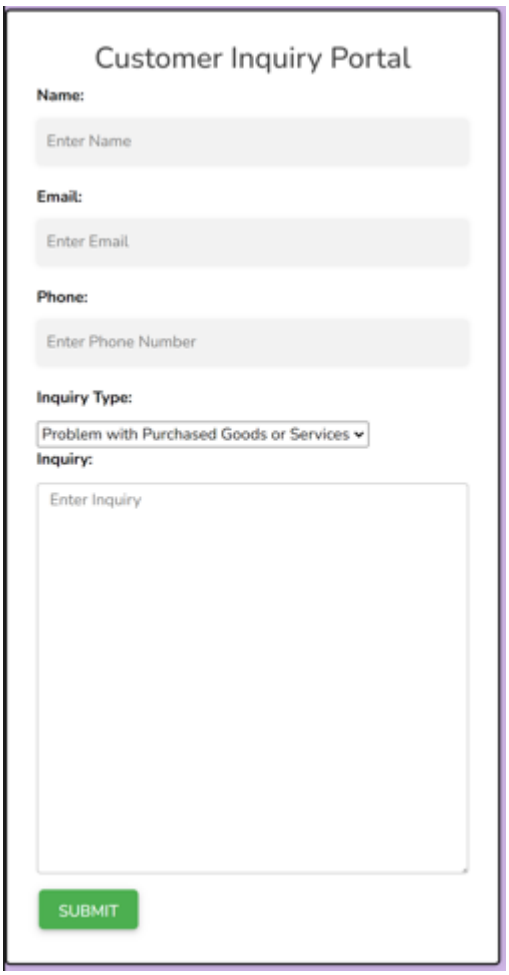
```
        return;
    } else {
        const res = await axios.post(
            `${BASE_URL}addsupplyorder`,    "addsupplyorder": Unknown word.
            createSupplyOrderForm
        );
        alert("Supply Order Created");
        set({
            SupplyOrder: [...SupplyOrderData, res.data],
            createSupplyOrder: {
                orderID: "",
                SID: "",
                supplierName: "",
                item: "",
                amount: "",
                price: "",
                discount: "",
                deliverydate: "",    "deliverydate": Unknown word.
            },
        });
    }
    //create SupplyOrder
},
```

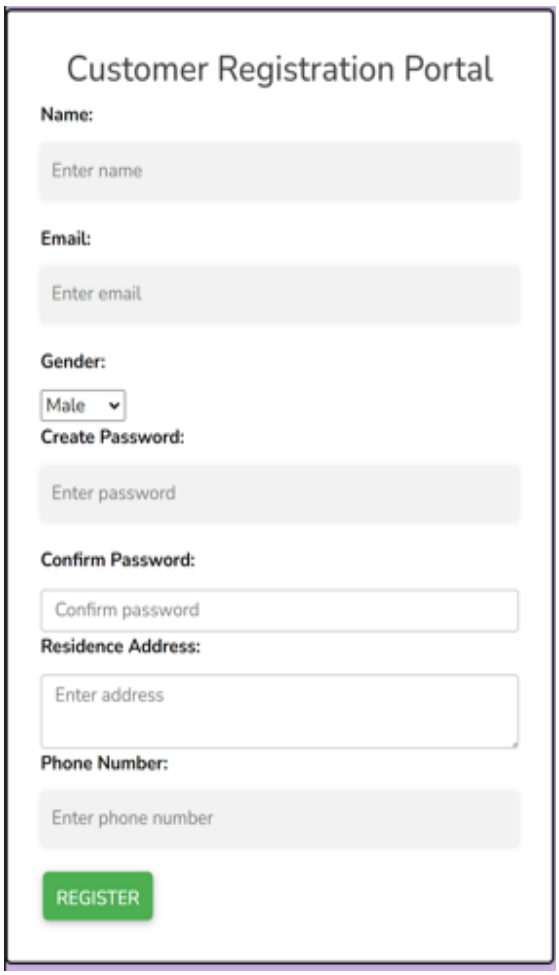
```
const generatePDF = () => {
    const doc = new jsPDF();
    doc.setProperties({
        title: "Supplier List",
    });
    doc.autoTable({
        head: [
            [
                "Order ID",
                "SID",
                "Supplier Name",
                "Item",
                "Amount",
                "Price",
                "Discount",
            ],
        ],
        body: store.SupplyOrderData.map((SupplyOrder) => [
            SupplyOrder.orderID,
            SupplyOrder.SID,
            SupplyOrder.supplierName,
            SupplyOrder.item,
            SupplyOrder.amount,
            SupplyOrder.price,
            SupplyOrder.discount,
        ]),
    });
    doc.save("supplier-order-list.pdf");
};
```

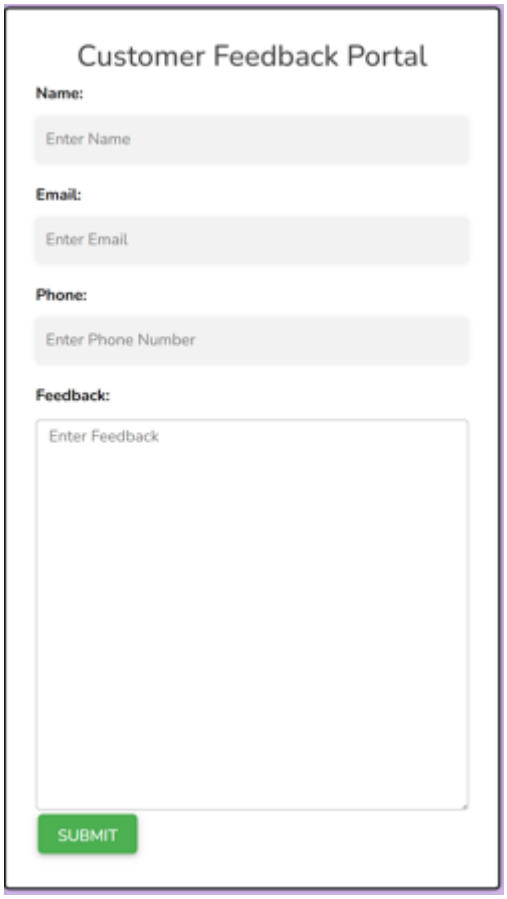
Customer Relationship Management System.

Interfaces

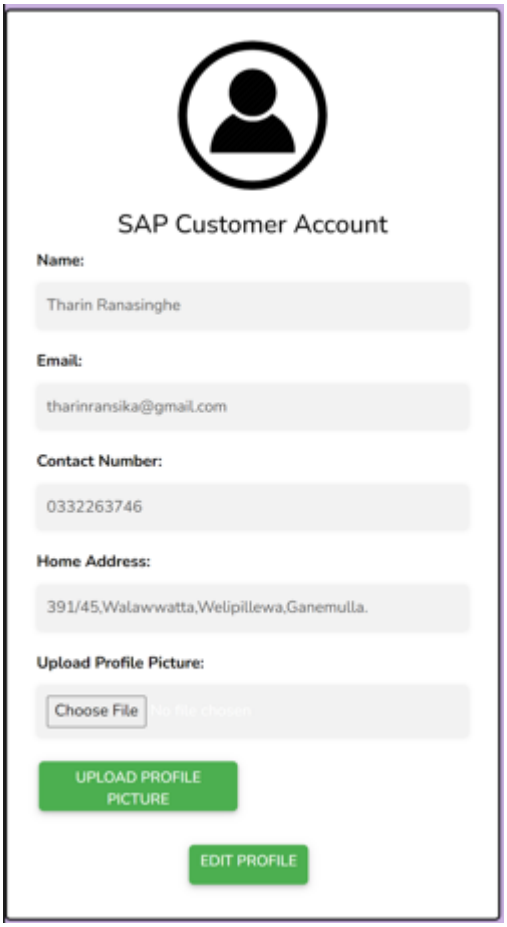


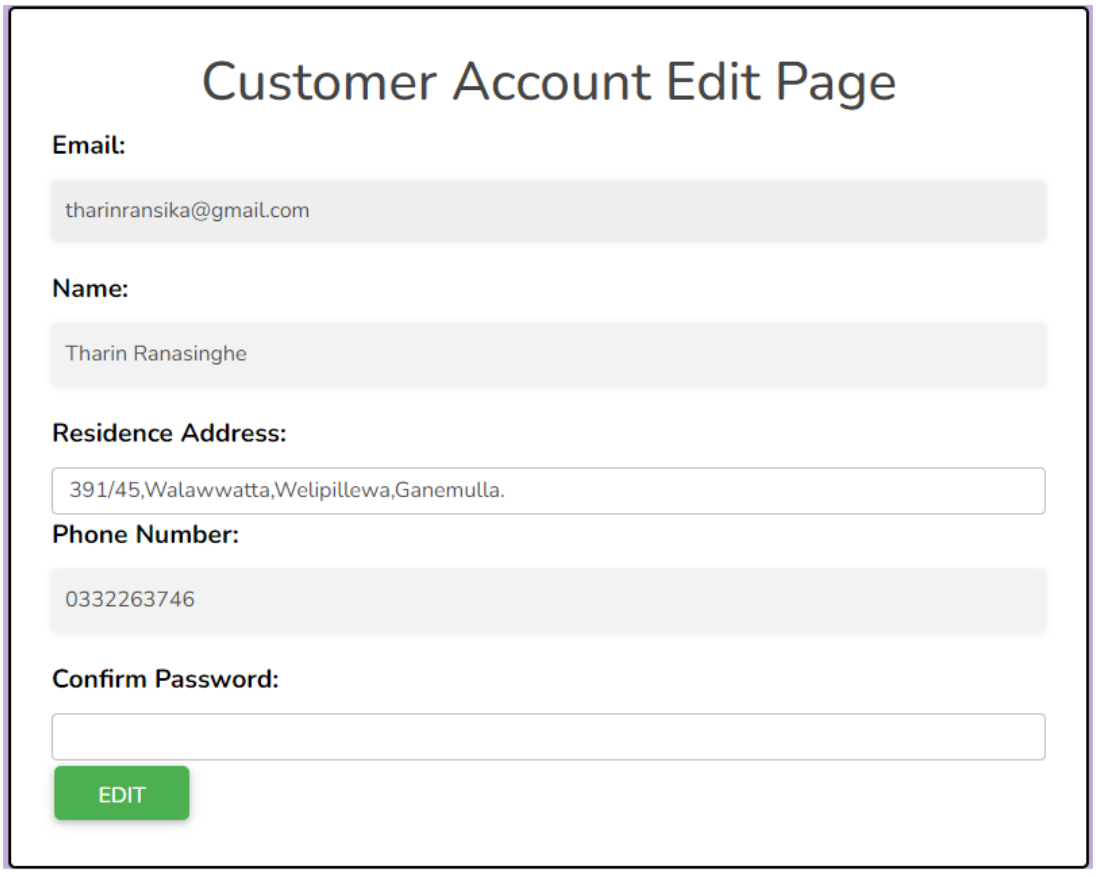


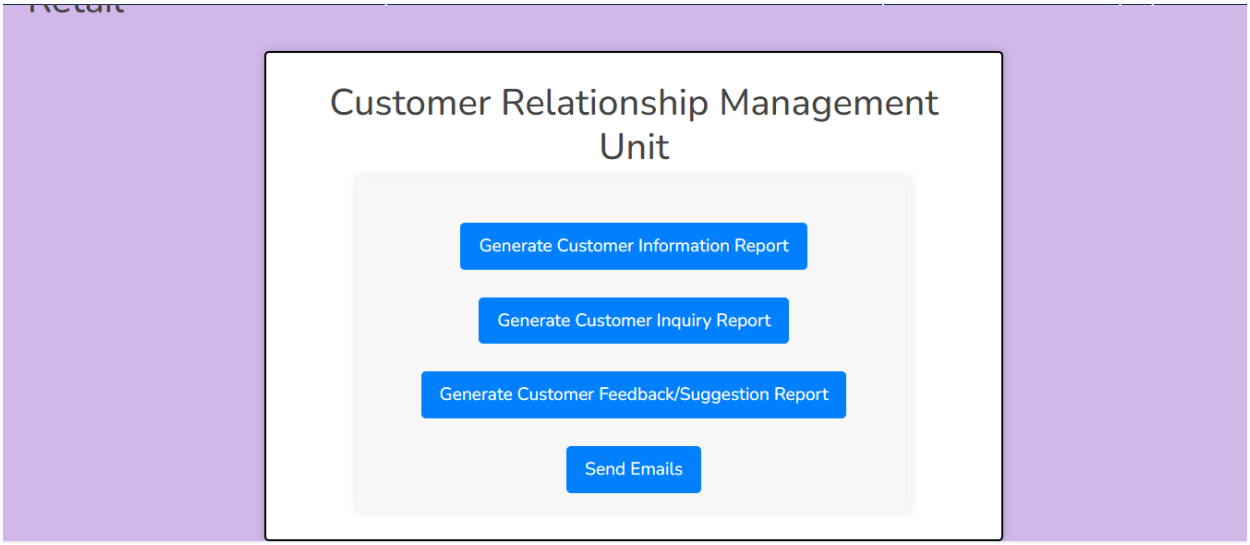



Customer Feedback Portal
Name:
Enter Name
Email:
Enter Email
Phone:
Enter Phone Number
Feedback:
Enter Feedback
SUBMIT


Sign In
Email address:
Enter email
Password:
Enter password
SUBMIT


SAP Customer Account
Name:
Tharin Ranasinghe
Email:
tharinransika@gmail.com
Contact Number:
0332263746
Home Address:
391/45,Walawwatta,Welipillewa,Ganemulla.
Upload Profile Picture:
Choose File
UPLOAD PROFILE PICTURE
EDIT PROFILE


Customer Account Edit Page
Email:
tharinransika@gmail.com
Name:
Tharin Ranasinghe
Residence Address:
391/45,Walawwatta,Welipillewa,Ganemulla.
Phone Number:
0332263746
Confirm Password:
EDIT

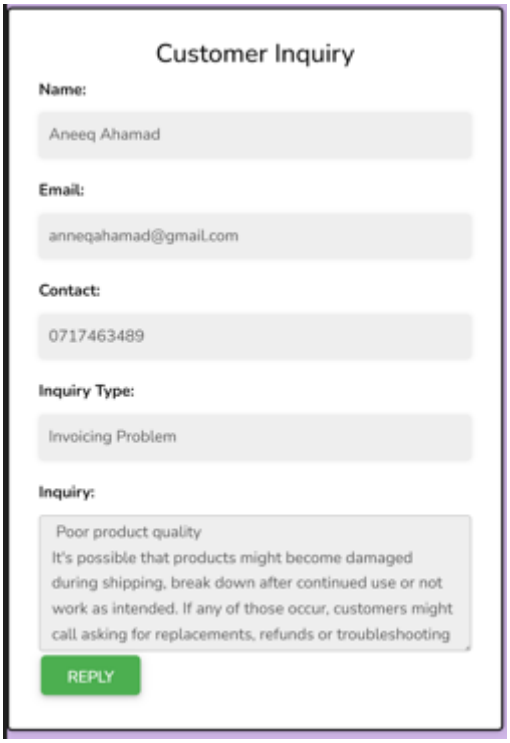


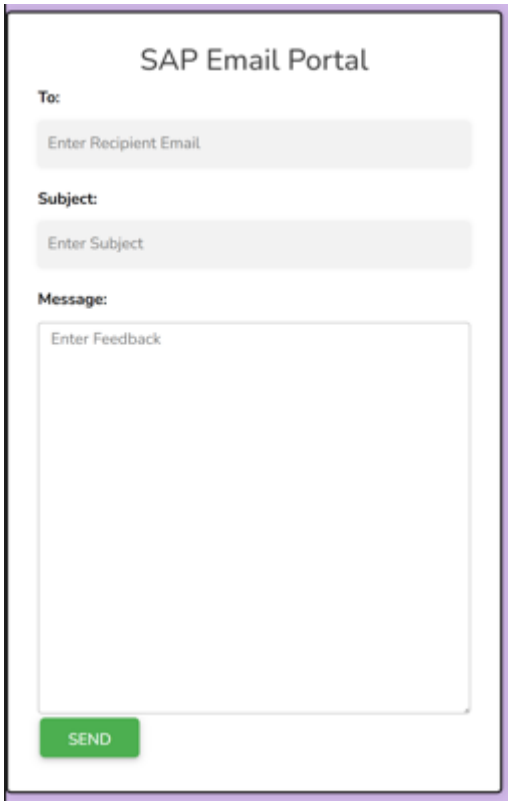
Customer Information Report

No	Name	Email	Home Address	Contact Number	Membership Started Date	Action
1	Tharin Ranasinghe	tharinransika@gmail.com	391/45,Walawwatta,Welipillewa,Ganemulla.	0332263746	2023-05-09T04:38:04.399Z	DELETE
2	Aneeq Ahamad	aneeqahamad@gmail.com	25/2,Kalubowila.	0704763282	2023-05-09T04:43:42.672Z	DELETE
3	Veynitha Bandara	veynithabandara@gmail.com	35/1,Malwatta Para, Ragama.	0718596453	2023-05-09T04:45:28.265Z	DELETE
4	Ovinda Namal	ovindanamal@gmail.com	72/A,Balangoda,Rathnapura	0723458241	2023-05-09T04:46:44.710Z	DELETE
5	Sindujan Paramnathan	sindujanparamanathan@gmail.com	31/A,Hatton,Nuwara-Eliya.	0574873524	2023-05-09T04:48:02.613Z	DELETE
6	Naveed Mohomad	naveedmohomad@gmail.com	32/F,Kingswood Road,Kandy	0574536752	2023-05-09T04:49:17.376Z	DELETE

Customer Inquiry Report

No	Name	Email	Contact Number	Inquiry Type	Inquiry Submitted Date	Inquiry Description	Action
1	Tharin Ransika	tharinransika@gmail.com	0711758782	Delivery Problem	2023-05-09T04:53:07.193Z	VIEW	DELETE
2	Aneeq Ahamad	anneqahamad@gmail.com	0717463489	Invoicing Problem	2023-05-09T04:54:57.800Z	VIEW	DELETE
3	Veynitha Bandara	veynithabandara@gmail.com	0714738972	Problem with Purchased Goods or Services	2023-05-09T04:56:44.386Z	VIEW	DELETE
4	Ovinda Namal	ovindanamal@gmail.com	0775738972	Invoicing Problem	2023-05-09T04:58:00.464Z	VIEW	DELETE
5	Chamika Karunaratne	chamika@gmail.com	0714563452	Invoicing Problem	2023-05-10T13:36:22.659Z	VIEW	DELETE





Customer Feedback Report

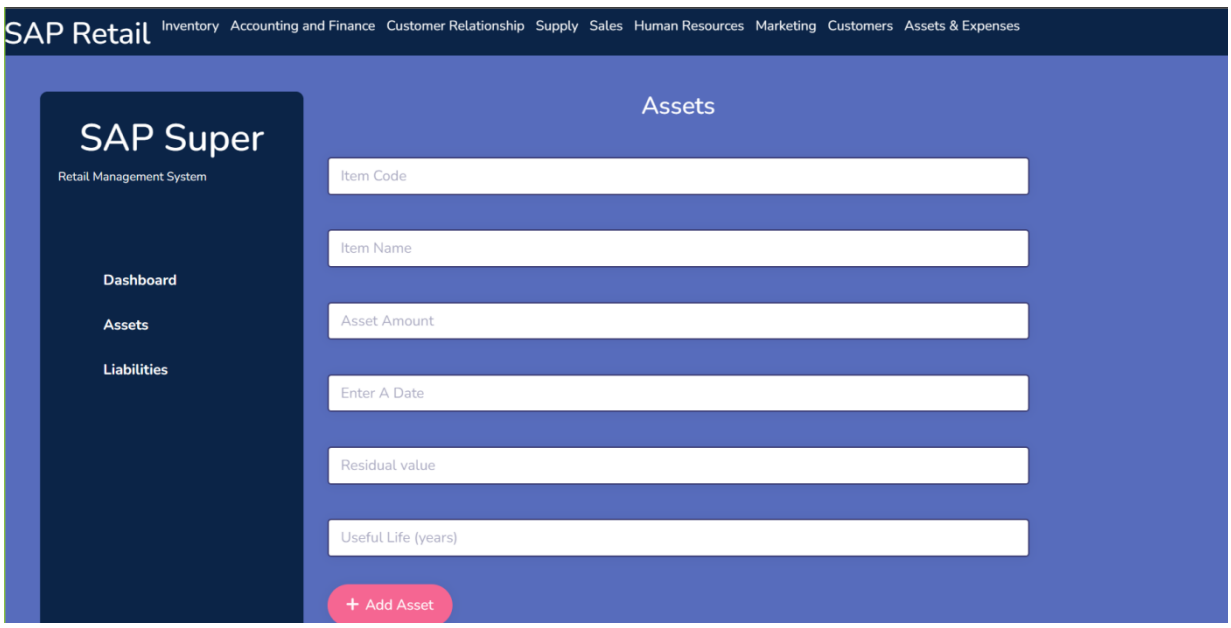
No	Name	Email	Contact Number	Feedaback Submitted Date	Feedaback Description	Action
1	Tharin Ransika	tharinransika@gmail.com	0711745637	2023-05-09T05:02:40.961Z	VIEW	DELETE
2	Veynitha Bandara	veynithabandara@gmail.com	0734645637	2023-05-09T05:04:16.134Z	VIEW	DELETE
3	Ovinda Namal	ovindanamal@gmail.com	0773456723	2023-05-09T05:05:50.450Z	VIEW	DELETE
4	Naveed Mohomad	naveedmohomad@gmail.com	0718345672	2023-05-09T05:06:44.989Z	VIEW	DELETE
5	Chamika Karunaratne	chamika@gmail.com	0712534673	2023-05-10T13:37:35.820Z	VIEW	DELETE
6	Sohan Perera	sohan@gmail.com	0711758782	2023-05-12T06:25:04.856Z	VIEW	DELETE

Assets and Expenses Tracking System

Use Interface

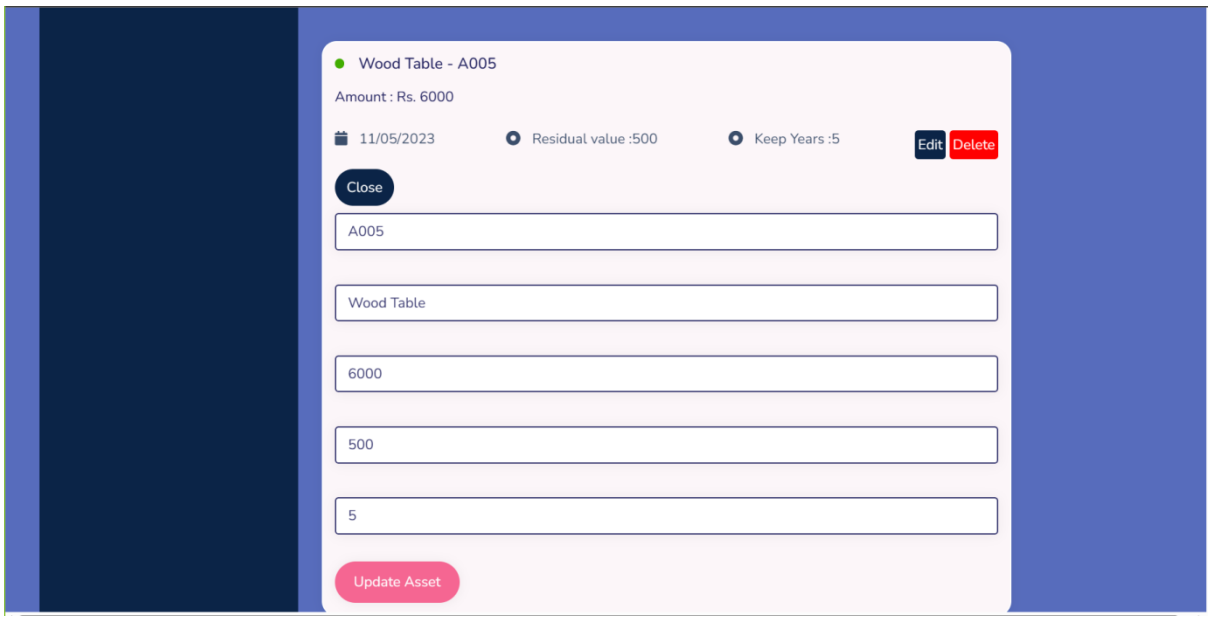
Dashboard

Asset form

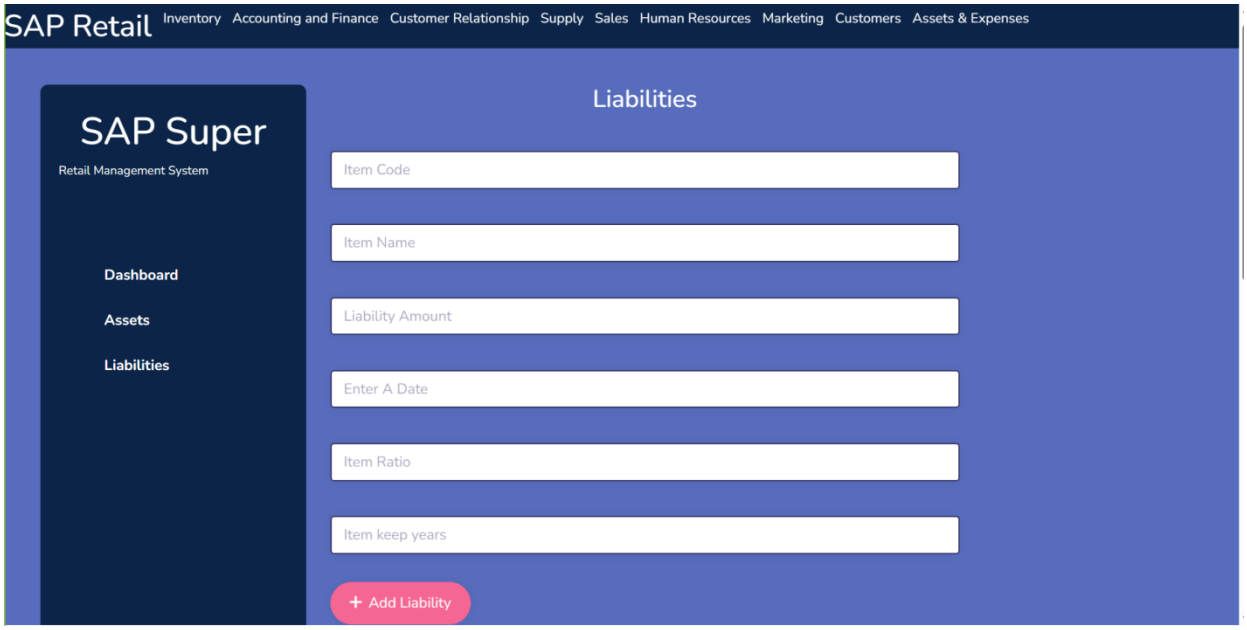


List of Assets

Update Assets

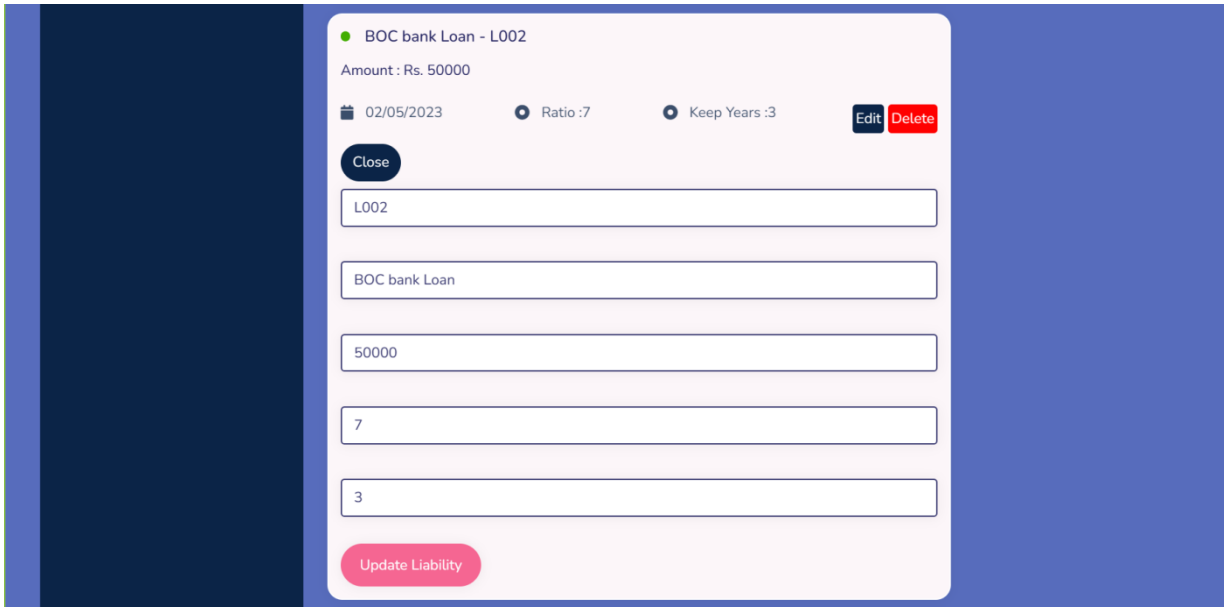


Expenses form (Liability)



List of Expenses (Liability)

Update Expenses (Liability)



Validation - Assets

Frontend

```
try {
    //validations
    if(!itemCode || !name || !date || !rValue || !years ){
        return res.status(400).json({message: 'All fields are required!'})
    }
    if(amount <= 0 || !amount === 'number'){
        return res.status(400).json({message: 'Amount must be a positive number!'})
    }
    await assets.save()
    res.status(200).json({message: 'Assets Added'})
} catch (error) {
    res.status(500).json({message: 'Server Error'})
}

console.log(assets)
```

Backend

```
const handleSubmit = e =>{
    e.preventDefault()
    if(!eItemCode || !eName || !eRValue || !eYears ){
        alert("All fields required!!!")
    }
    else if( !eAmount || eAmount<=0 ){
        alert("Invalid Amount!!!")
    }
    else{
        updateAsset(id, data)
        alert("Asset Updated!")
    }
}
```

Validation - Liability

Frontend

```
try {
    //validations
    if(!itemCode || !name || !date || !ratio || !years ){
        return res.status(400).json({message: 'All fields are required!'})
    }
    if(amount <= 0 || !amount === 'number'){
        return res.status(400).json({message: 'Amount must be a positive number!'})
    }
    await assets.save()
    res.status(200).json({message: 'Liability Added'})
} catch (error) {
    res.status(500).json({message: 'Server Error'})
}
```

Backend

```
const handleSubmit = e =>{
    e.preventDefault()
    if(!eItemCode || !eName || !eRatio || !eYears ){
        alert("All fields required!!!")
    }
    else if( !eAmount || eAmount<=0 ){
        alert("Invalid Amount!!!")
    }
    else{
        updateLiability(id, data)
        alert("Liability Updated!")
    }
}
```

Calculatins

Assets

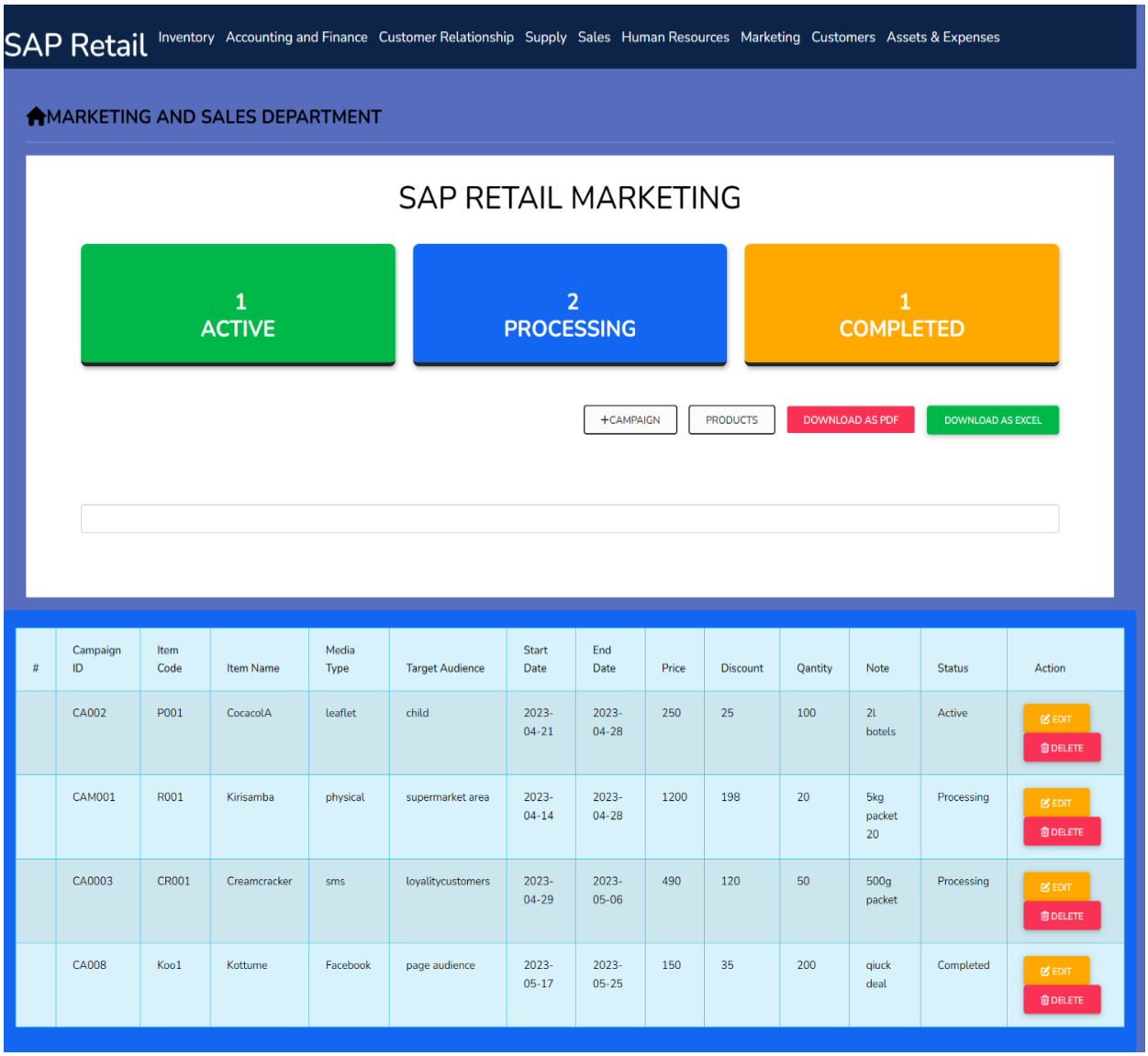
```
//Get total valu
    const totalAssets = () => {
        let totalAsset = 0;
        assets.forEach((asset) =>{
            totalAsset = totalAsset + asset.amount
        })

        return totalAsset;
    }

//Depreciation
    const totalDepreciation = () => {
        let totalDep= 0;
        assets.forEach((asset) =>{
            const dep = (asset.amount - asset.rValue) / asset.years;
            totalDep += parseFloat(dep.toFixed(2));
        })

        return totalDep;
    }
```

Expenses (Liability)

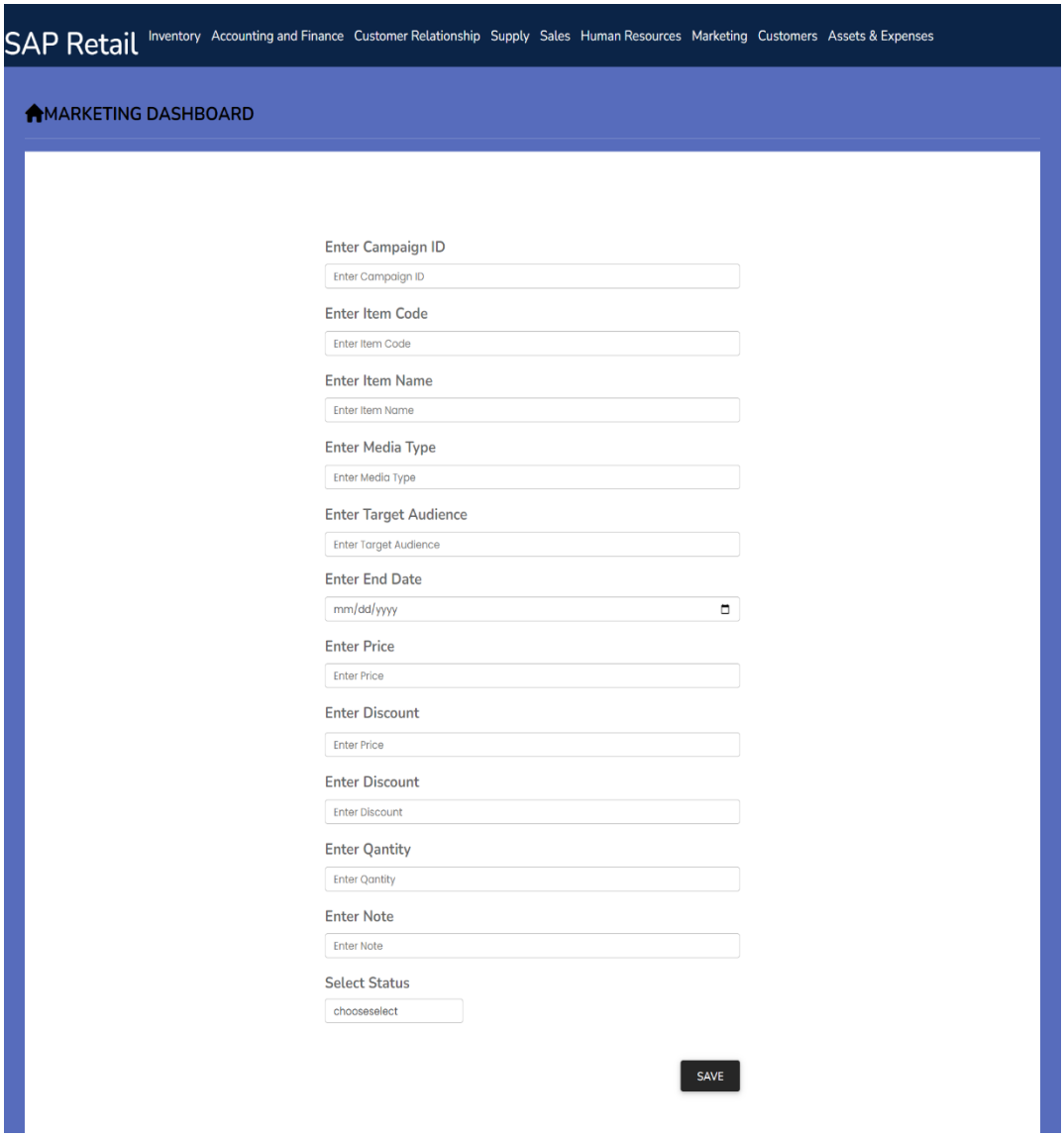
```
//Get total
    const totalLiabilities = () => {
        let totalLiability  = 0;
        liabilities.forEach((liability) =>{
            totalLiability  = totalLiability  + liability.amount
        })

        return totalLiability ;
    }
//calculate Interest
    const totalInterest =()=>{
        let interest = 0;
        liabilities.forEach((liability)=>{
            const rate = (liability.amount * liability.ratio * liability.years) / 100;
            interest += parseFloat(rate.toFixed(2));
        })
        return interest;
    }
```

Marketing and sales management system

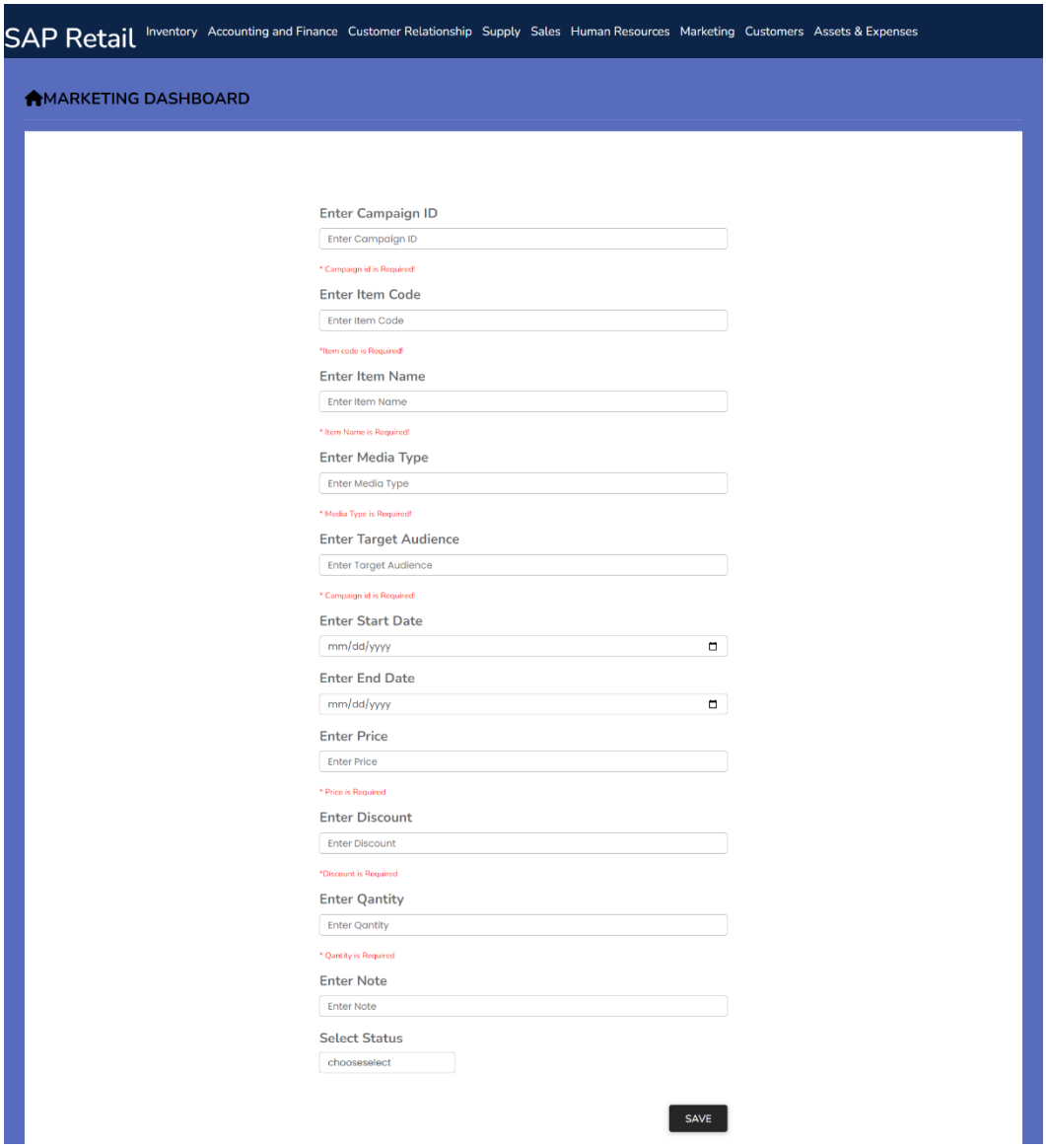
Dashboard



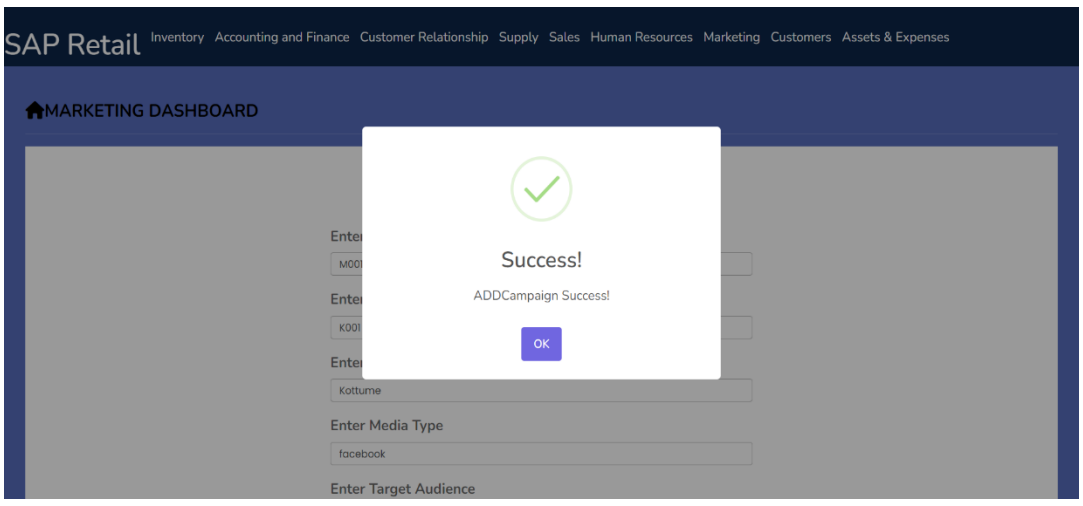
campaign form



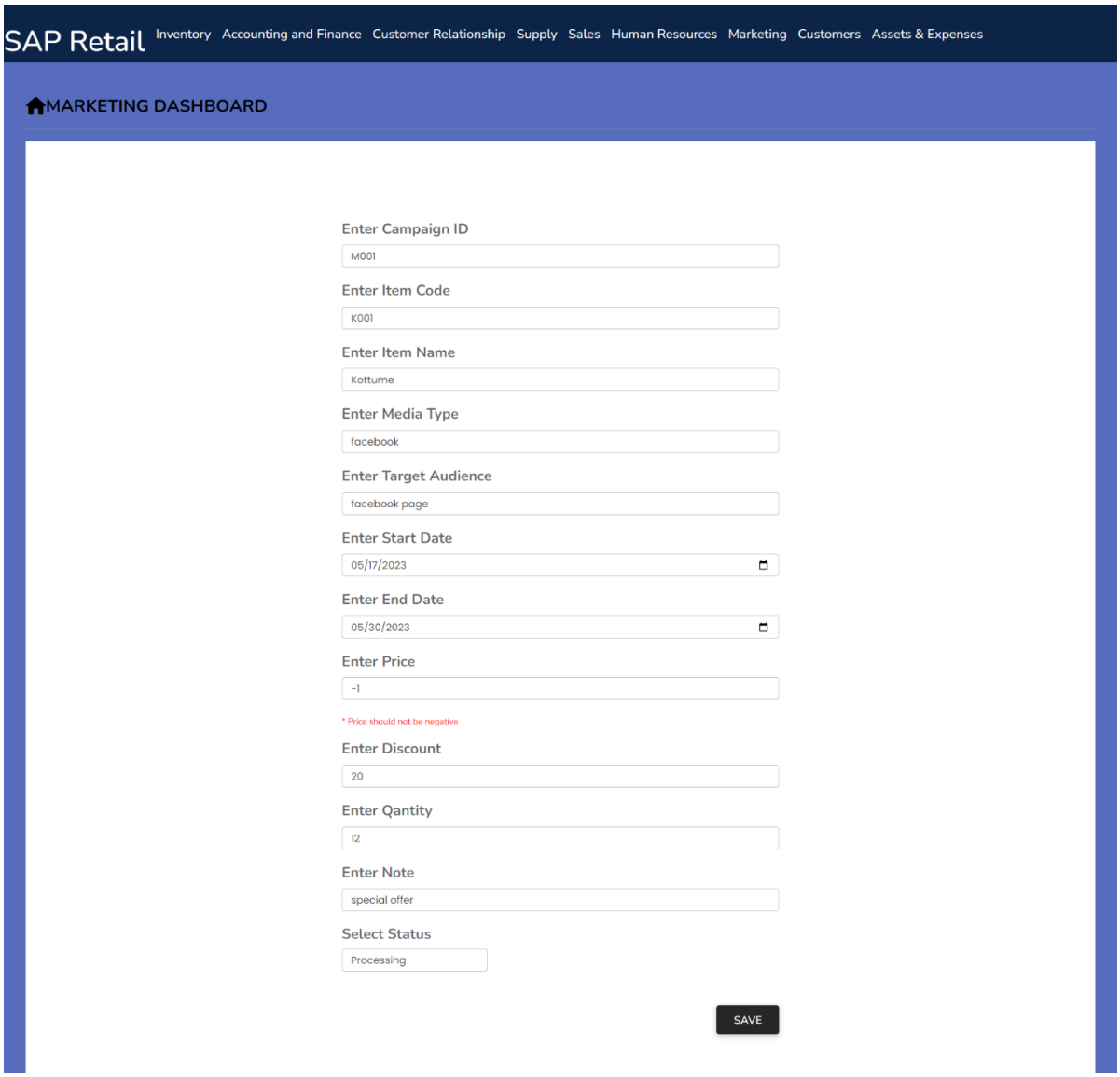
Validation display



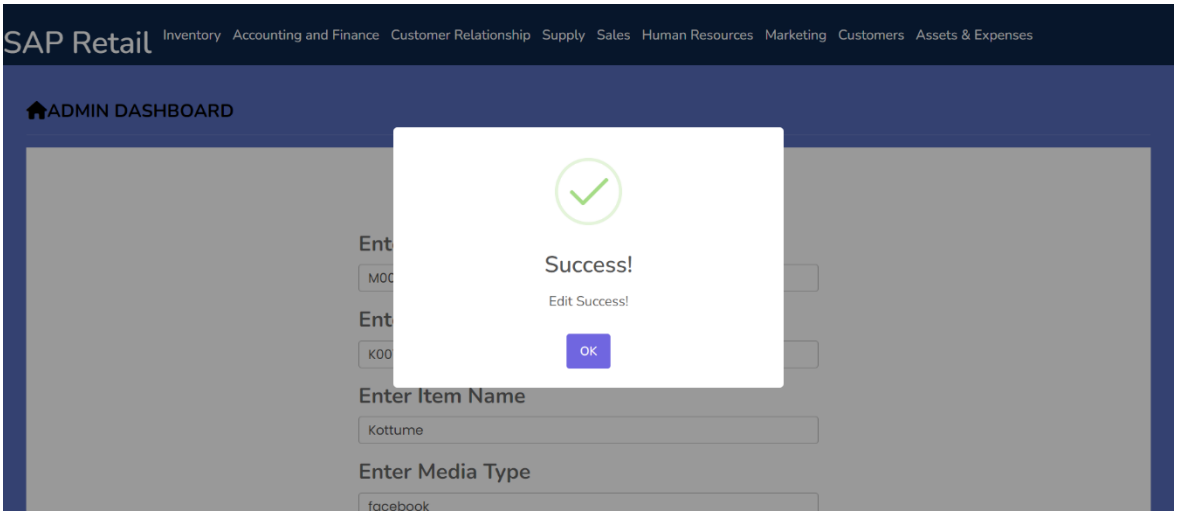
Successfully added pop up message box



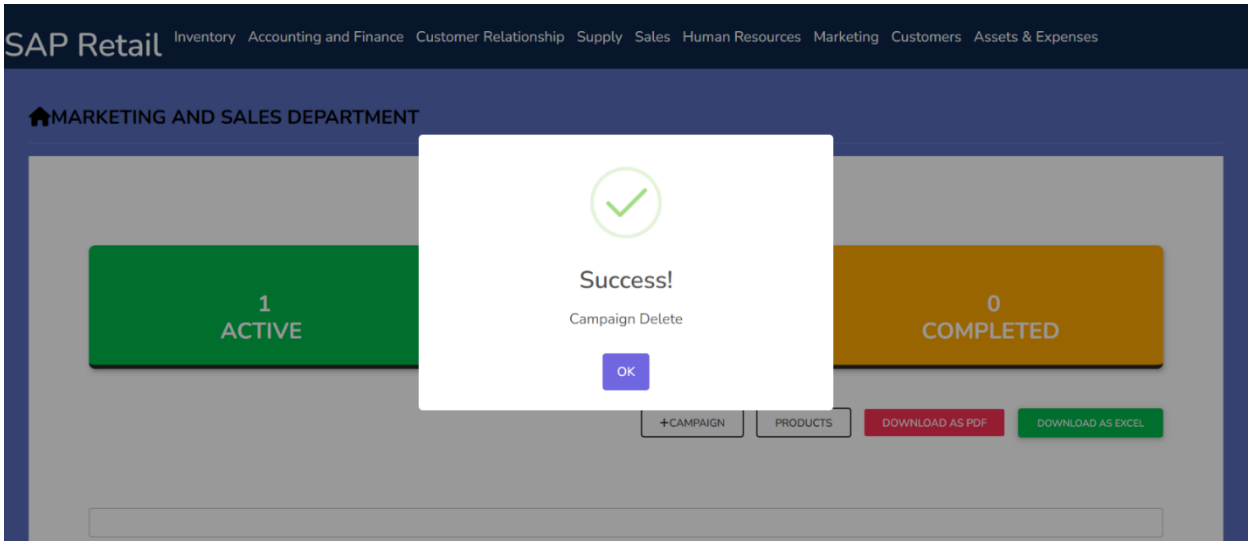
Campaign update form



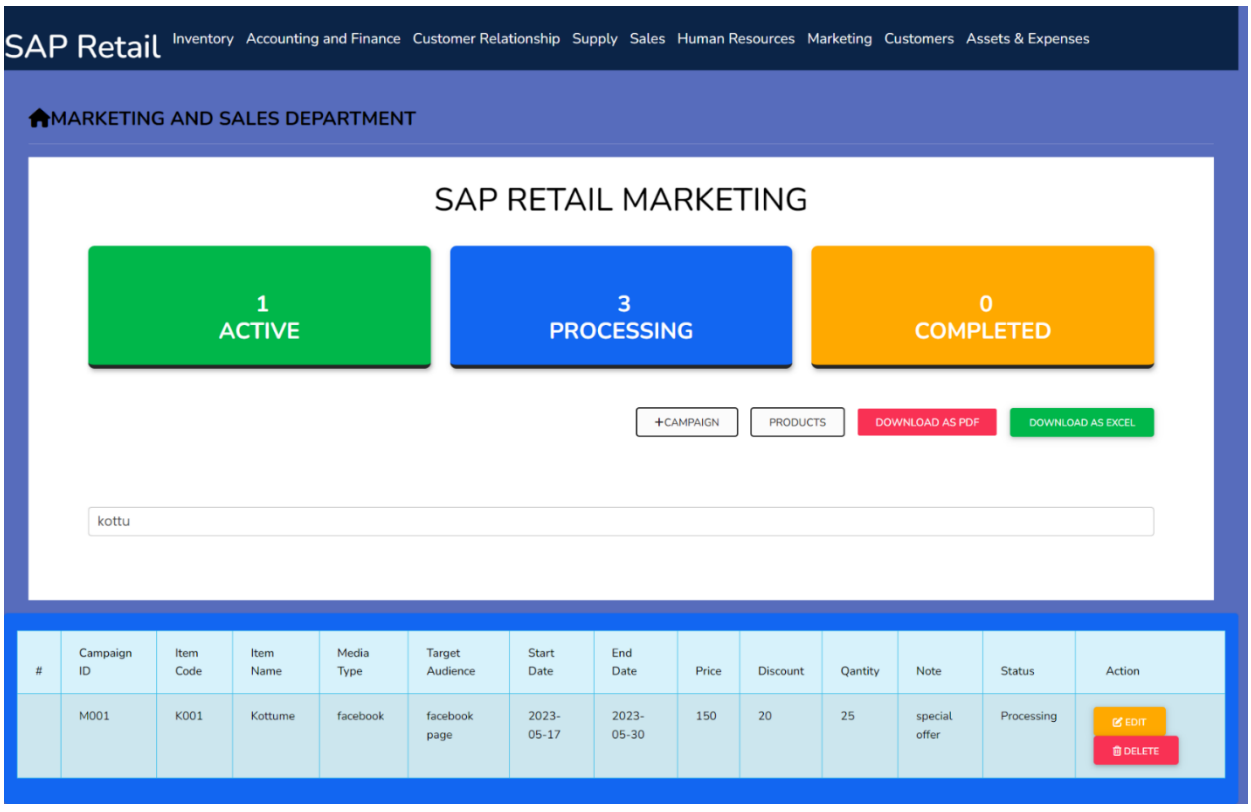
Update successful pop up message box



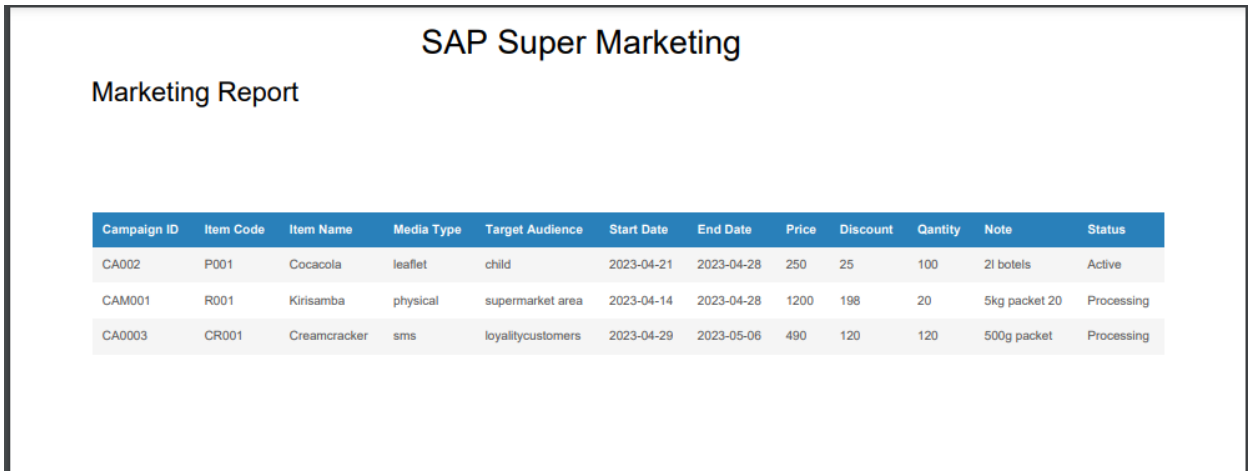
Delete success pop up message box



Search for item name



Generate report PDF

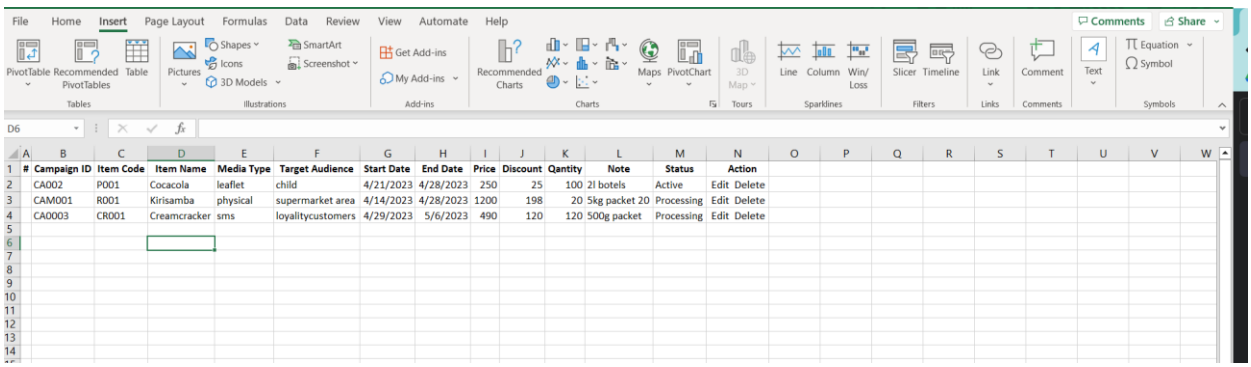


SAP Super Marketing

Marketing Report

Campaign ID	Item Code	Item Name	Media Type	Target Audience	Start Date	End Date	Price	Discount	Qantity	Note	Status
CA002	P001	Cocacola	leaflet	child	2023-04-21	2023-04-28	250	25	100	2l botels	Active
CAM001	R001	Kirisamba	physical	supermarket area	2023-04-14	2023-04-28	1200	198	20	5kg packet 20	Processing
CA0003	CR001	Creamcracker	sms	loyalitycustomers	2023-04-29	2023-05-06	490	120	120	500g packet	Processing

Generated report Excel



#	Campaign ID	Item Code	Item Name	Media Type	Target Audience	Start Date	End Date	Price	Discount	Qantity	Note	Status	Action
2	CA002	P001	Cocacola	leaflet	child	4/21/2023	4/28/2023	250	25	100	2l botels	Active	Edit Delete
3	CAM001	R001	Kirisamba	physical	supermarket area	4/14/2023	4/28/2023	1200	198	20	5kg packet 20	Processing	Edit Delete
4	CA0003	CR001	Creamcracker	sms	loyalitycustomers	4/29/2023	5/6/2023	490	120	120	500g packet	Processing	Edit Delete

Count of status card view

```
function countActiveElements(array) {

    const status = "Active"
    const count = array.reduce((acc, cur) => cur.Status === status ? ++acc : acc, 0);
    setActiveCount(count);
    console.log(count);
}

function countProcessingElements(array) {

    const status = "Processing"
    const count = array.reduce((acc, cur) => cur.Status === status ? ++acc : acc, 0);
    setProcesingCount(count);
    console.log(count);
}

function countCompeletedElements(array) {

    const status = "Completed"
    const count = array.reduce((acc, cur) => cur.Status === status ? ++acc : acc, 0);
    setCompltedCount(count);
    console.log(count);
}
```

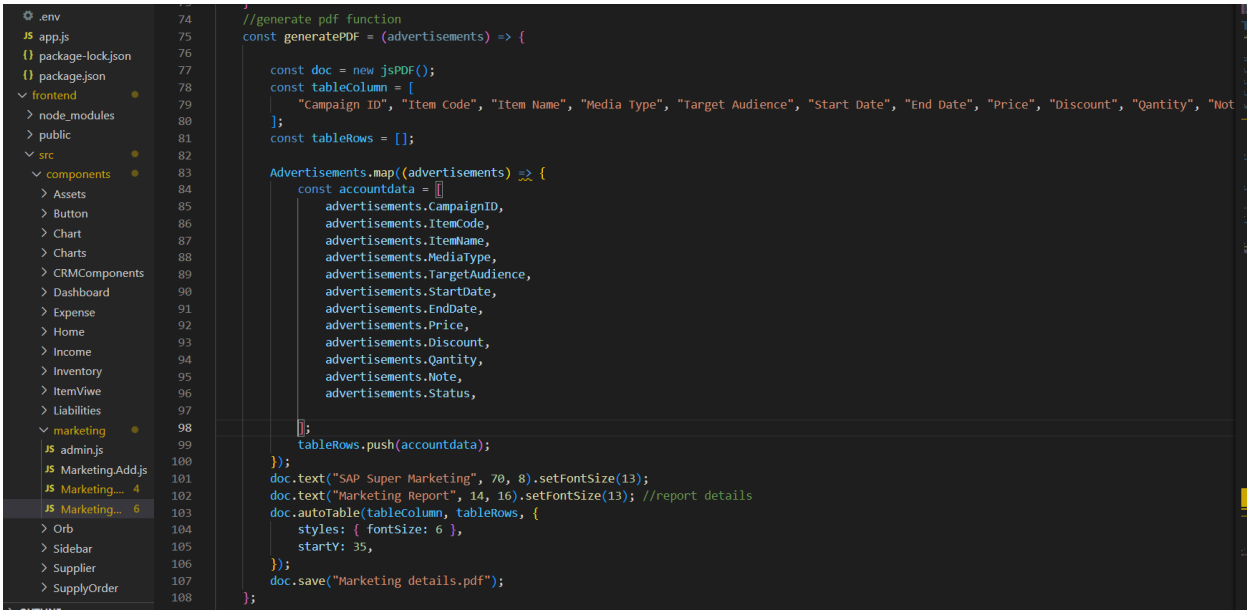
validations

```
//check statements and pass error message
if (!CampaignID) {
    CampaignIDError = "*  Campaign id is Required!"

}
if (!ItemCode) {
    ItemCodeError = "*Item code is Required!"

}
if (!ItemName) {
    ItemNameError = "* Item Name is Required!"

}
if (!MediaType) {
    MediaTypeError = "* Media Type is Required!"

}
if (!TargetAudience) {
    TargetAudienceError = "* Audience is Required!"

}

if (Price.toString().match("-")) {
    PriceError = "* Price should not be negative "
}
if (!Price) {
    PriceError = "* Price is Required"
}
if (Discount.toString().match("-")) {
    DiscountError = "* Discount amount should not be negative "
}
```

pop up message box

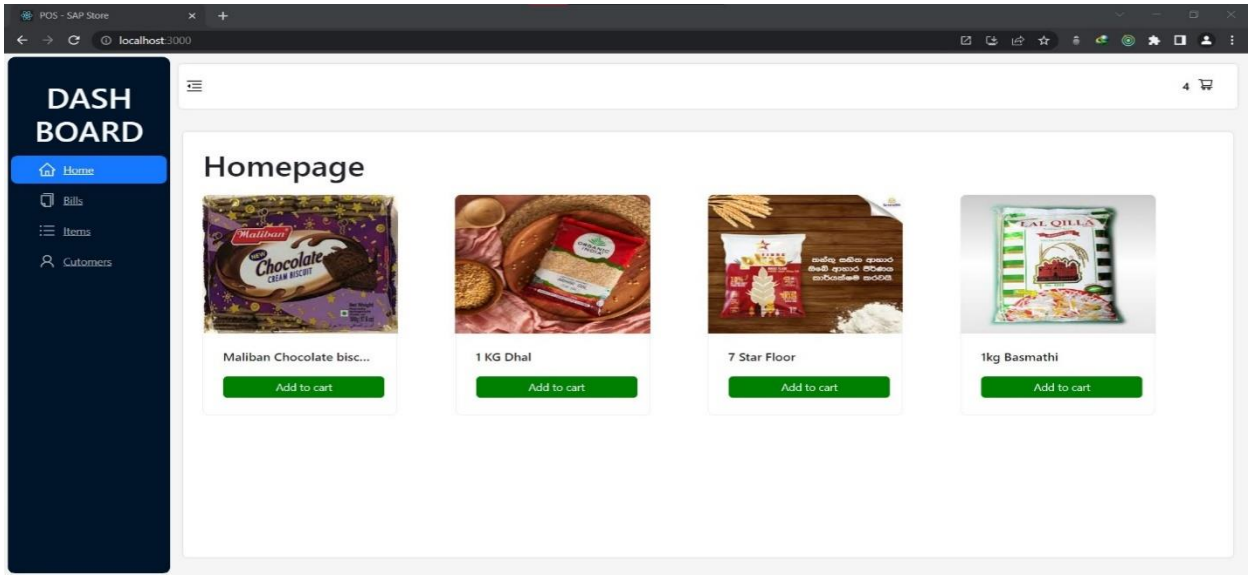
```
    status
}
debugger;
console.log(advertising);
if (isValid) {
    axios.post("http://localhost:5000/api/v1/addCampaign/", advertising).then(() => {

        Swal.fire({
            title: "Success!",
            text: "ADDCampaign Success!",
            icon: 'success',
            confirmButtonText: "OK",
            type: "success"
        })
    }).catch((err) => {

        Swal.fire({
            title: "Error!",
            text: "ADDCampaign Not Success",
            icon: 'error',
            confirmButtonText: "OK",
            type: "success"
        })
    })
}
```

Generate pdf

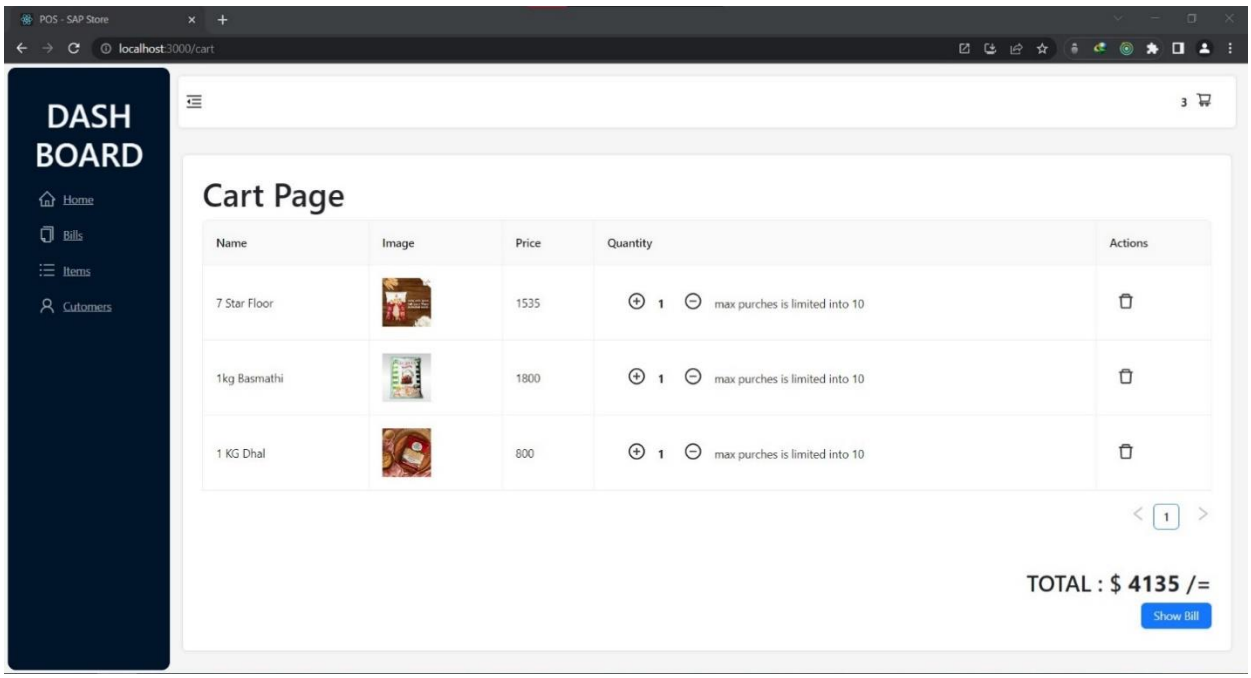


Point of Sales system

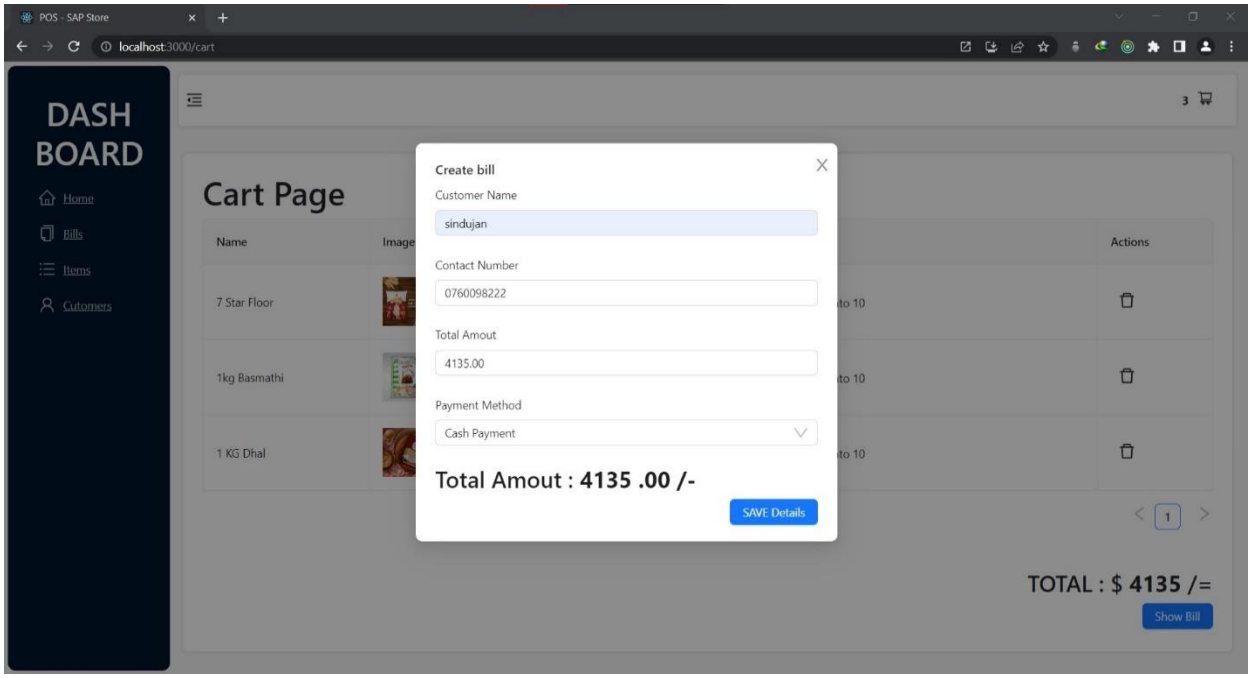
Point of sales dashboard



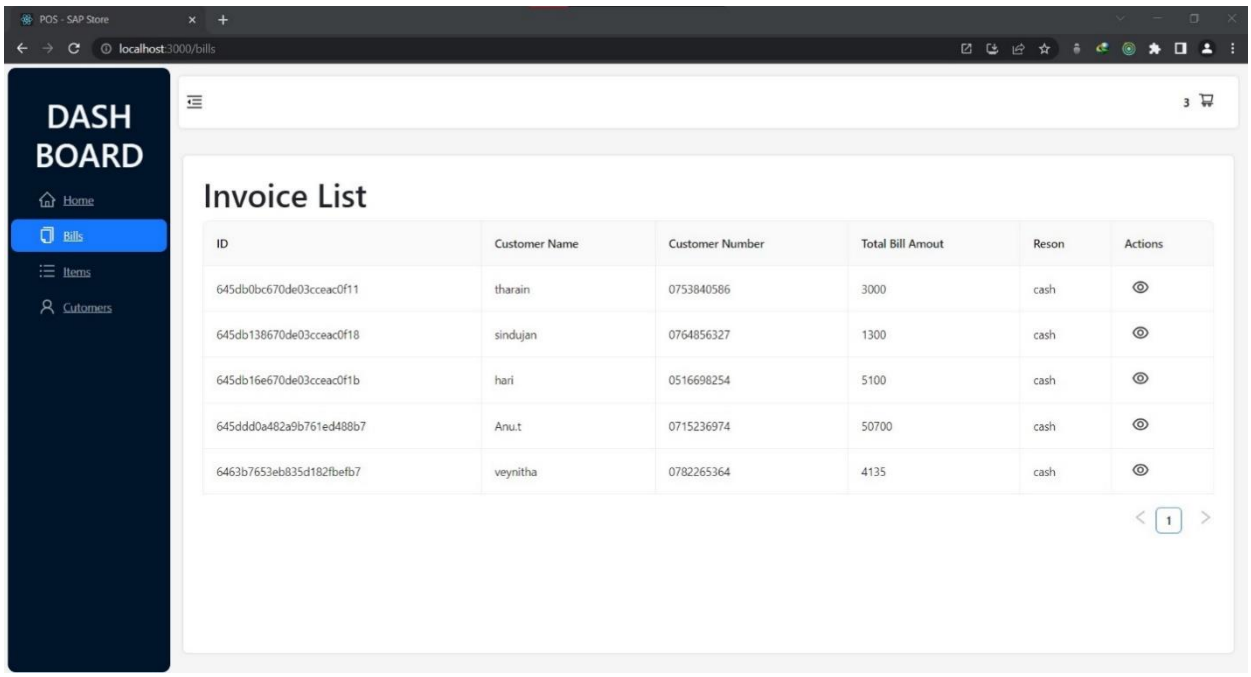
Point of sales Cart page



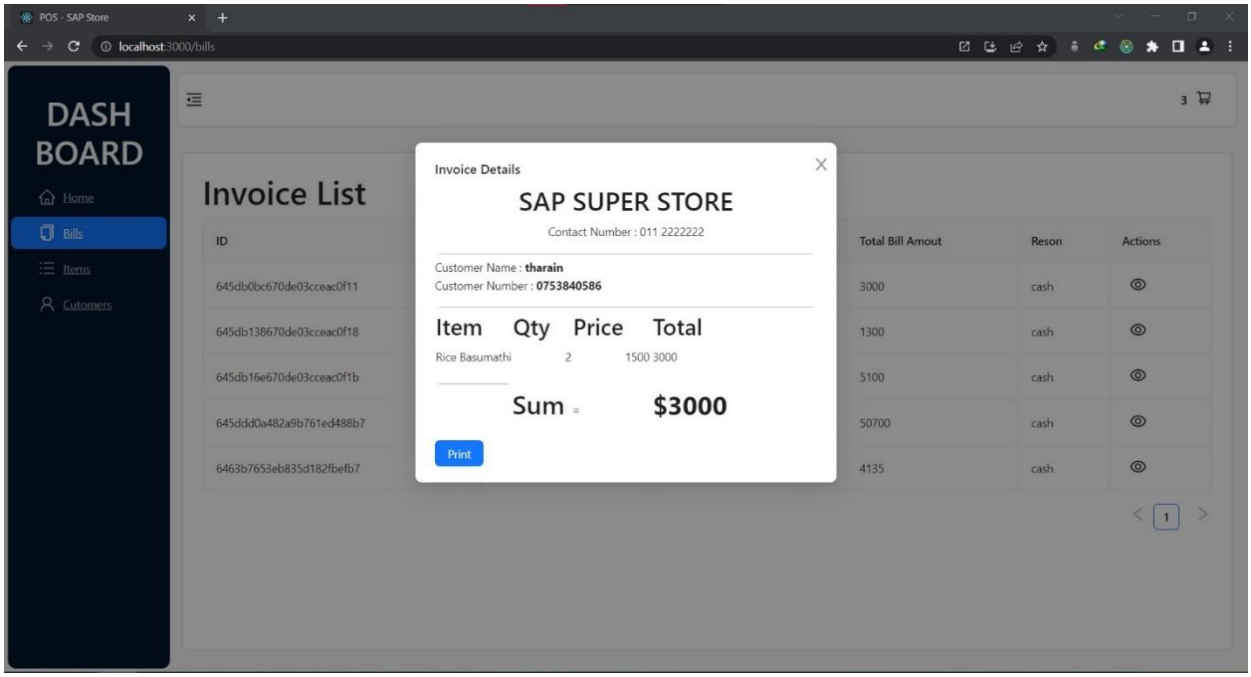
Point of sales Get customer name form



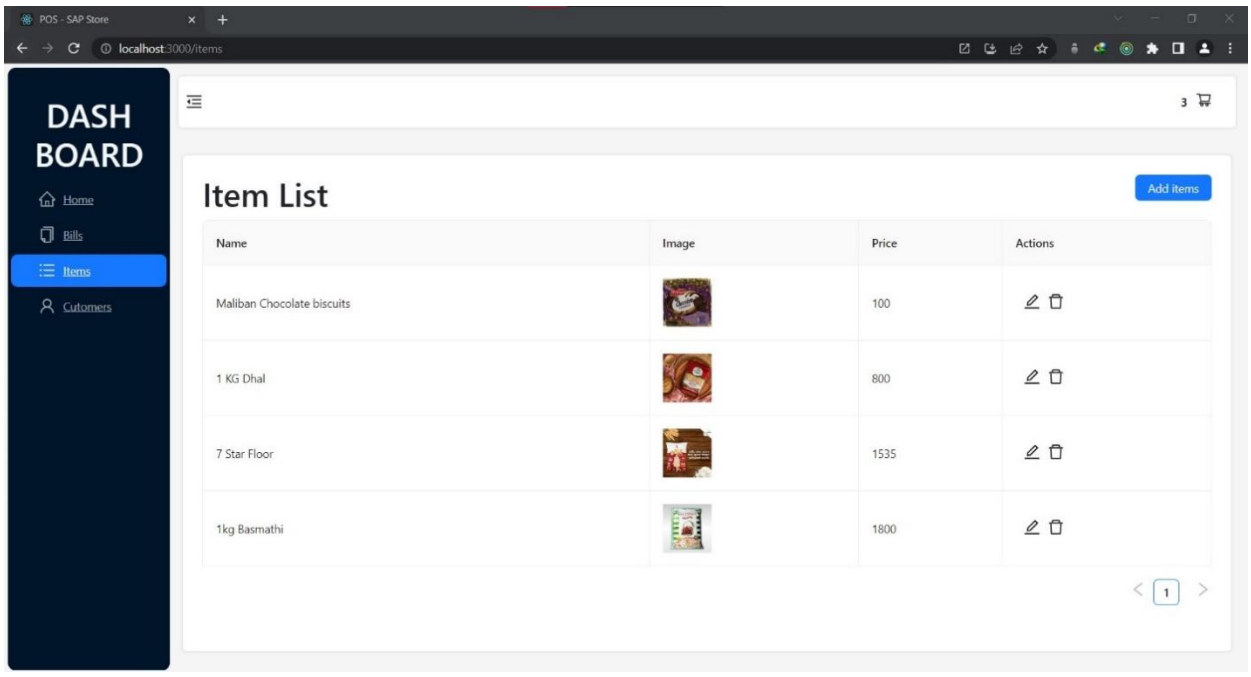
Point of sales bill page



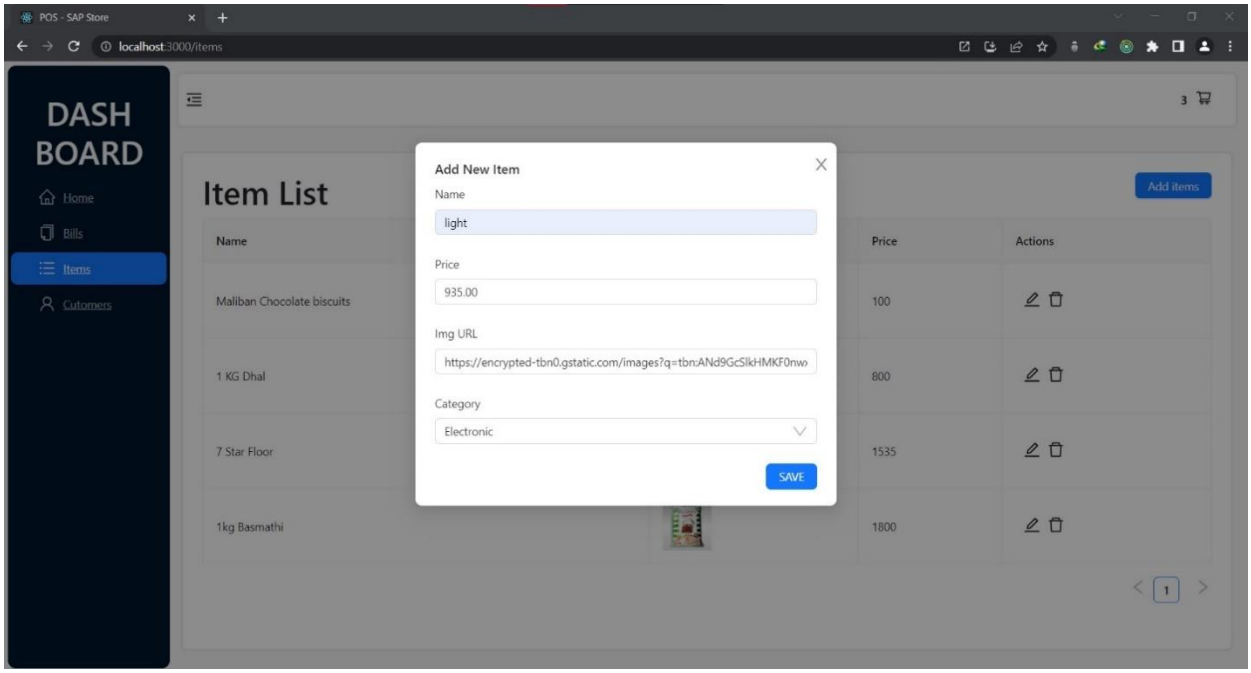
Point of sales invoice generation page



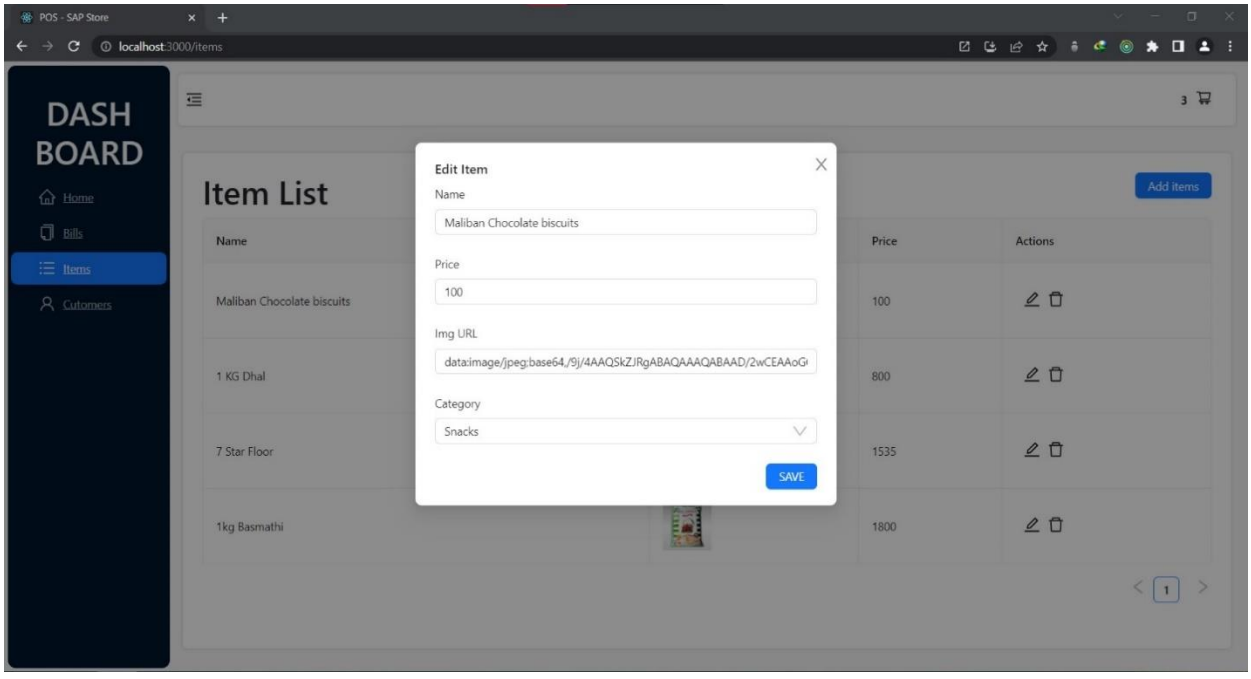
Point of sales Show Item page



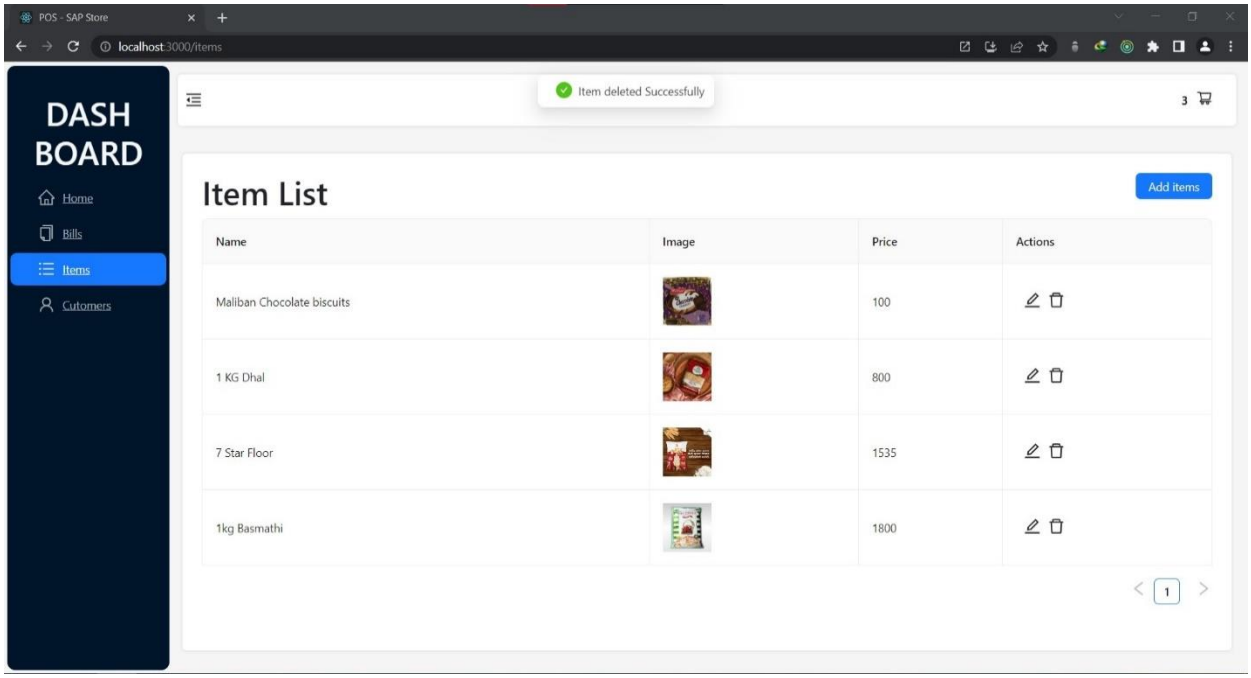
Point of sales Add Items page



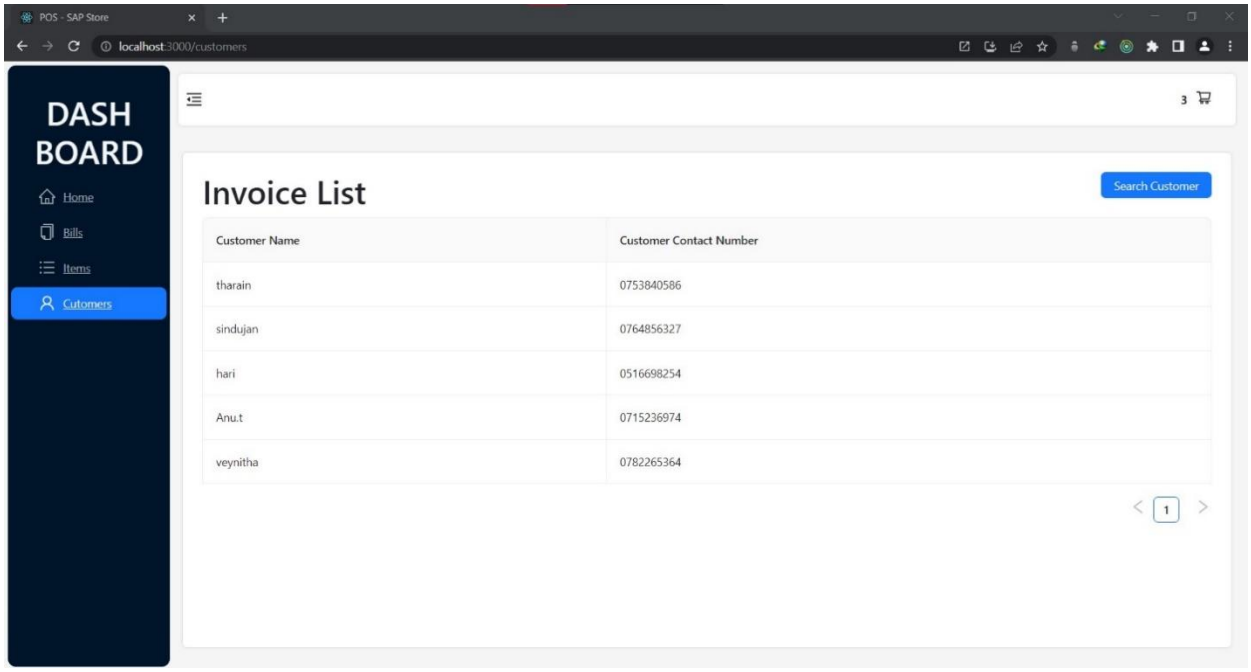
Point of sales Edit items page



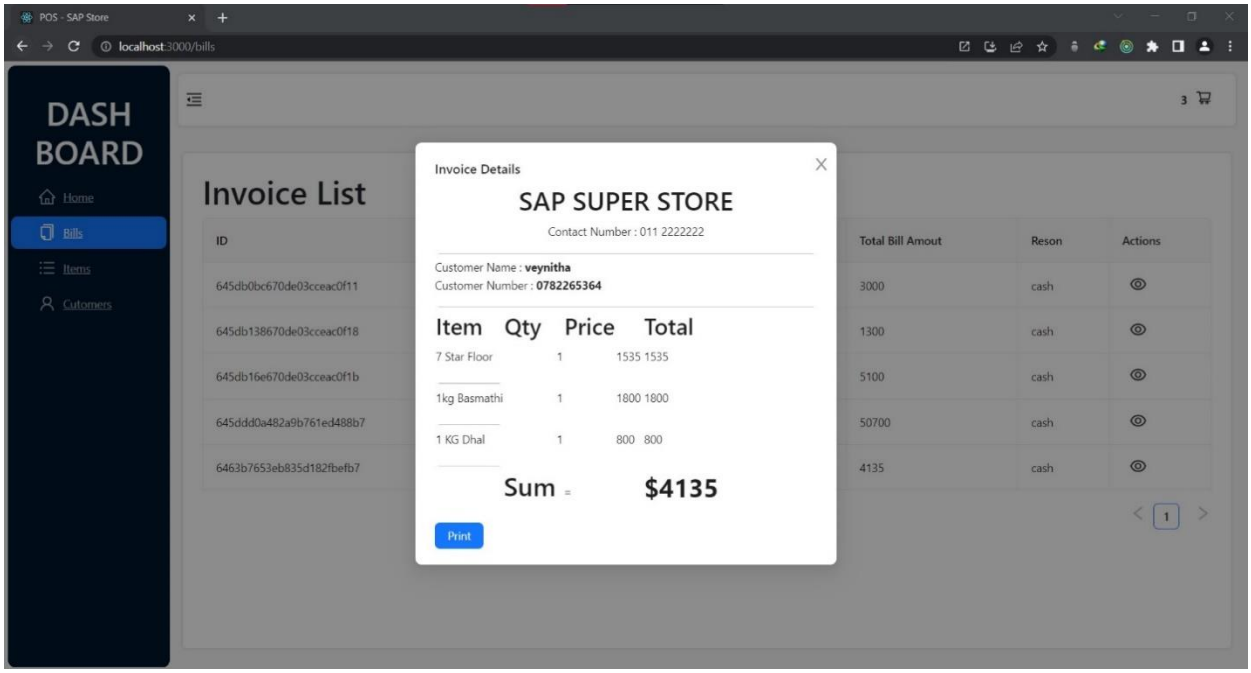
Point of sales delete items page



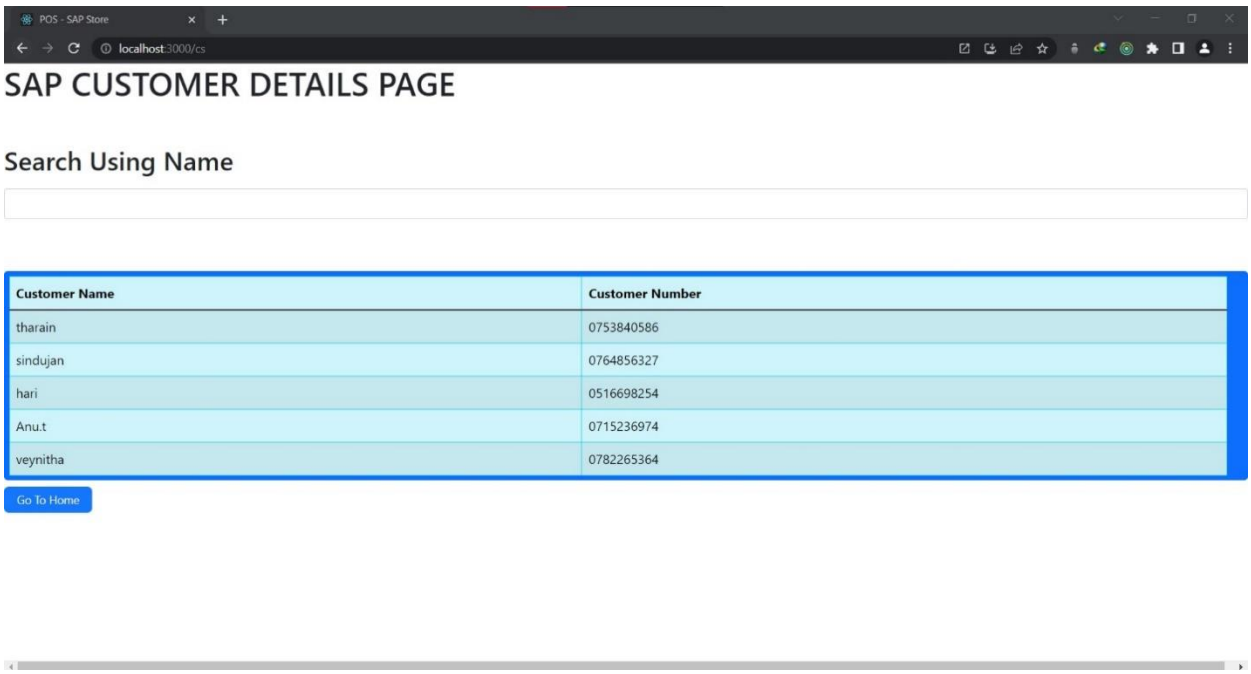
Point of sales customer list page



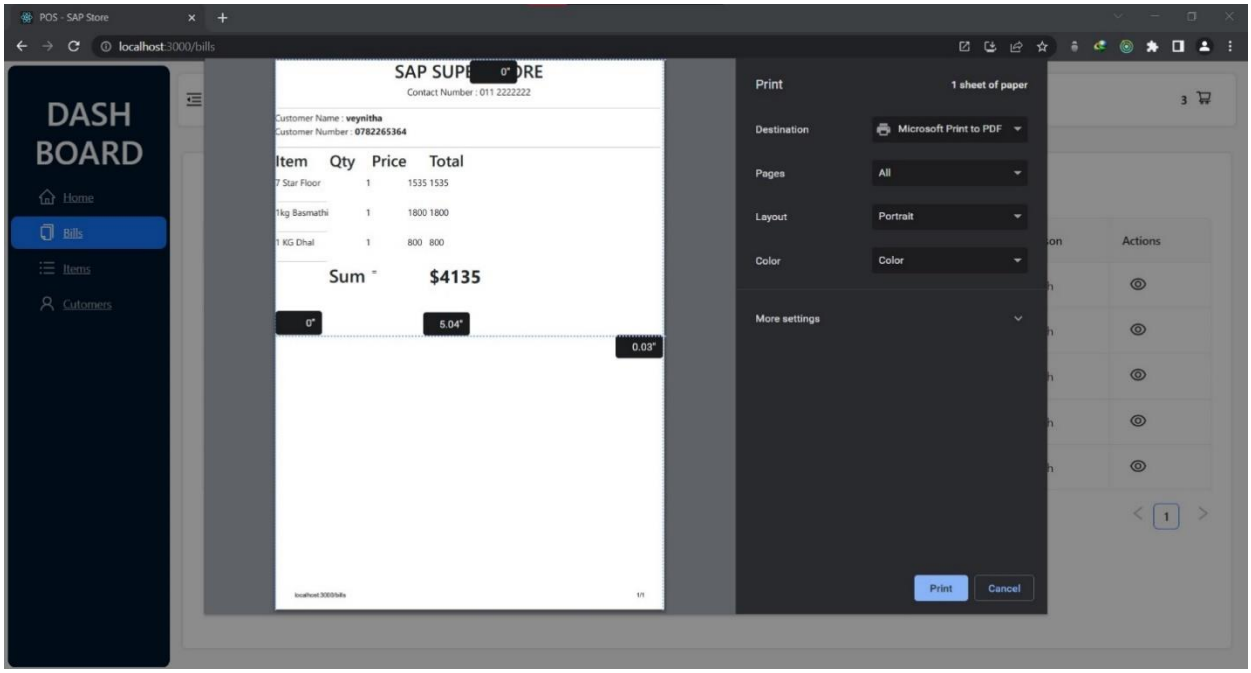
Point of sales invoice generation



Point of sales Search customer page



Point of sales Bill print page



Route

```
backend > routes > JS itemRoutes.js > ...
    editItemController,
    deleteItemController
} = require("../controllers/itemControllers");

const router = express.Router();

//routes
//Method - get
router.get("/get-item", getItemController);

//MEthod - POST
router.post("/add-item", addItemController);

//method - PUT
router.put("/edit-item", editItemController);

//delete-method
router.post('/delete-item',deleteItemController)

router.post("/add-bill", addBillController);

router.get("/get-bill",getbillController);

module.exports = router;
```

Controllers

```
backend > controllers > itemControllers.js > getItemController
const itemModel = require("../models/itemModel");

// get items
const getItemController = async (req, res) => {
  try {
    const items = await itemModel.find();
    res.status(200).send(items);
  } catch (error) {
    console.log(error);
  }
};

//add items
const addItemController = async (req, res) => {
  try {
    const newItem = new itemModel(req.body);
    await newItem.save();
    res.status(201).send("Item Created Successfully!");
  } catch (error) {
    res.status(400).send("error", error);
    console.log(error);
  }
};
```

```
//update item
const editItemController = async (req, res) => {
  try {
    await itemModel.findOneAndUpdate({_id:req.body.itemId},req.body);

    res.status(201).json("item Updated");
  } catch (error) {
    res.status(400).send(error);
    console.log(error);
  }
};
//delete item
const deleteItemController = async (req, res) => {
  try {
    const {itemId} = req.body;
    await itemModel.findOneAndDelete({_id:itemId});
    res.status(201).json("item Deleted");
  } catch (error) {
    res.status(400).send(error);
    console.log(error);
  }
};
module.exports = { getItemController, addItemController, editItemController,deleteItemController };
```

backend > controllers > JS billControllers.js > ...

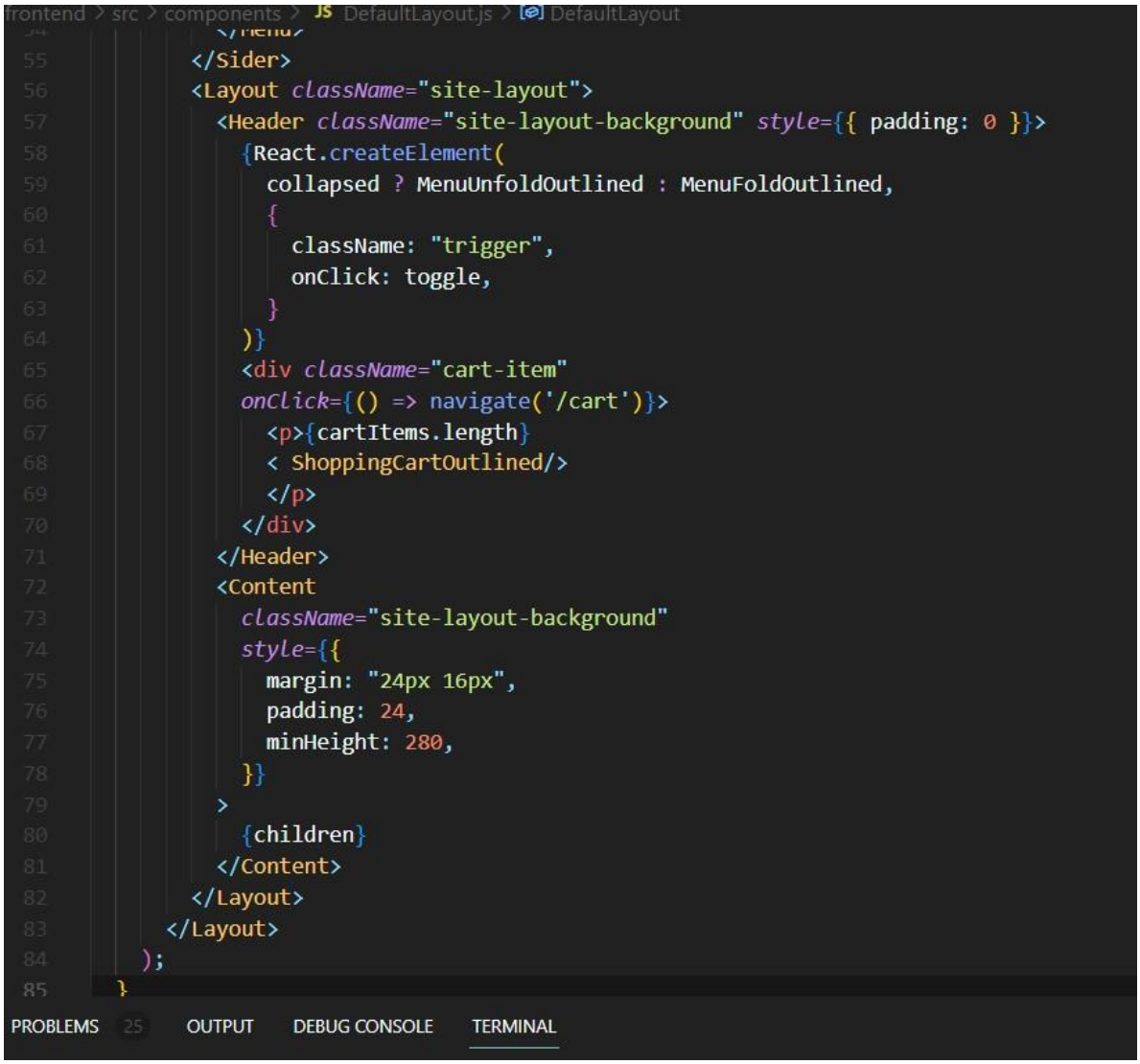
```
const billsModel = require("../models/billsModel");


//add bills
const addBillController = async (req, res) => {
  try {
    const newBill = new billsModel(req.body);
    await newBill.save();
    res.send("bill Created Successfully!");
  } catch (error) {
    res.send("Something went wrong");
    console.log(error);
  }
};

const getbillController = async (req, res) => {
  try {
    const bills = await billsModel.find();
    res.send(bills);
  } catch (error) {
    console.log(error);
  }
};
```

Layouts

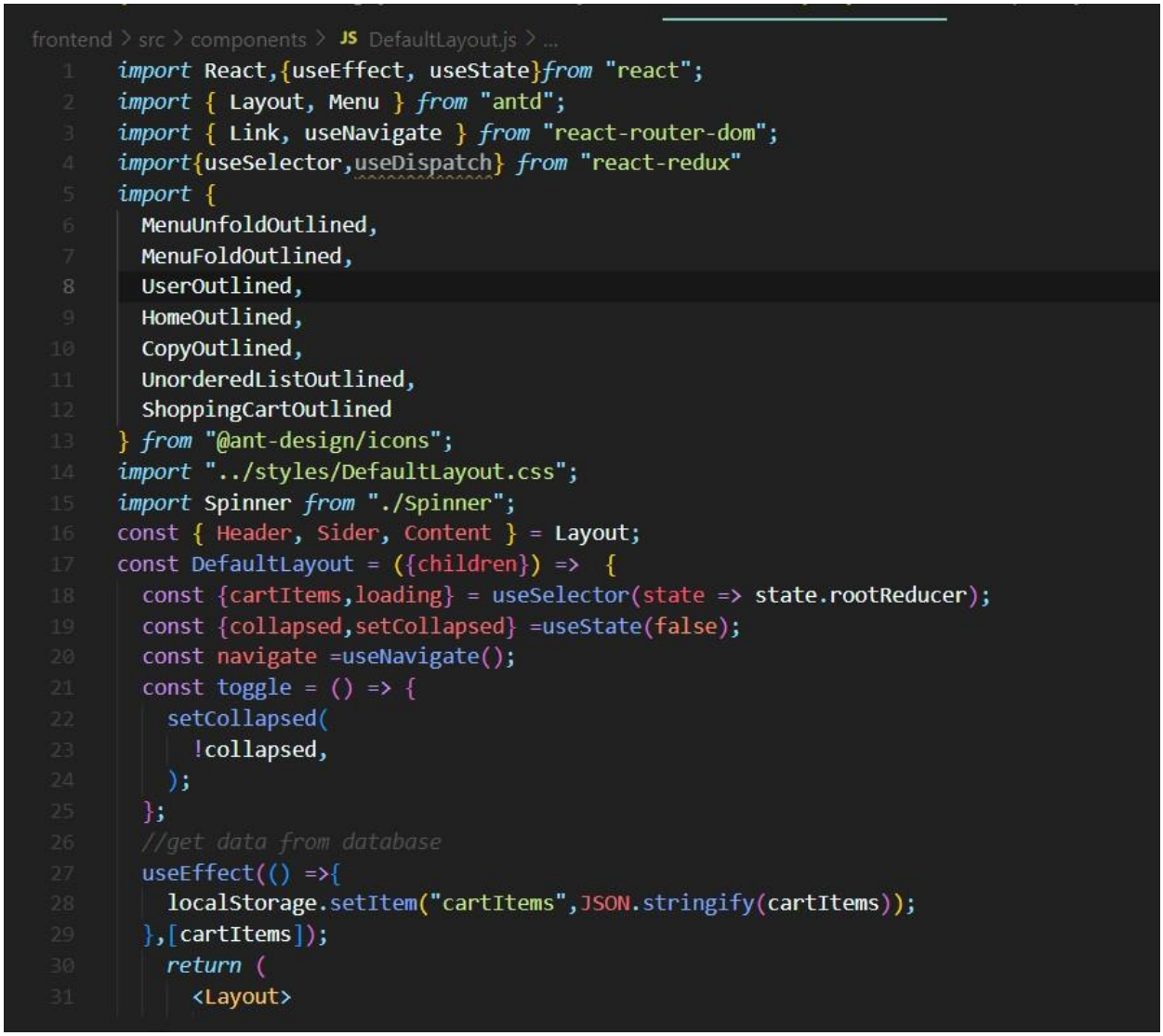
frontend > src > components > JS DefaultLayout.js > ...

```
        <Layout>
          {loading && <Spinner />}
          <Sider trigger={null} collapsible collapsed={collapsed}>
            <div className="logo">
              <h1 className="text-center text-light font-wight-bold mt-4">DASH BOARD</h1>
            </div>
            <Menu
              theme="dark"
              mode="inline"
              defaultSelectedKeys={window.location.pathname}
            >
              <Menu.Item key="/" icon={<HomeOutlined />}>
                <Link to="/">Home</Link>
              </Menu.Item>
              <Menu.Item key="/bills" icon={<CopyOutlined />}>
                <Link to="/bills">Bills</Link>
              </Menu.Item>
              <Menu.Item key="/items" icon={<UnorderedListOutlined />}>
                <Link to="/items">Items</Link>
              </Menu.Item>
              <Menu.Item key="/customers" icon={<UserOutlined />}>
                <Link to="/customers">Cutomers</Link>
              </Menu.Item>
            </Menu>
          </Sider>
          <Layout className="site-layout">
            <Header className="site-layout-background" style={{ padding: 0 }}>
              {React.createElement(
                collapsed ? MenuUnfoldOutlined : MenuFoldOutlined,
                {
                  className: "trigger",
```

PROBLEMS 25 OUTPUT DEBUG CONSOLE TERMINAL


frontend > src > components > JS DefaultLayout.js > DefaultLayout
</Sider>
<Layout className="site-layout">
<Header className="site-layout-background" style={{ padding: 0 }}>
{React.createElement(
collapsed ? MenuUnfoldOutlined : MenuFoldOutlined,
{
className: "trigger",
onClick: toggle,
}
)}
<div className="cart-item"
onClick={() => navigate('/cart')}>
{cartItems.length}
< ShoppingCartOutlined/>

</Header>
<Content
className="site-layout-background"
style={{
margin: "24px 16px",
padding: 24,
minHeight: 280,
}}
>
{children}
</Content>
</Layout>
</Layout>
);
}
PROBLEMS 25
OUTPUT
DEBUG CONSOLE
TERMINAL


frontend > src > components > JS DefaultLayout.js > ...
import React,{useEffect, useState}from "react";
import { Layout, Menu } from "antd";
import { Link, useNavigate } from "react-router-dom";
import{useSelector,useDispatch} from "react-redux"
import {
MenuUnfoldOutlined,
MenuFoldOutlined,
UserOutlined,
HomeOutlined,
CopyOutlined,
UnorderedListOutlined,
ShoppingCartOutlined
} from "@ant-design/icons";
import "../styles/DefaultLayout.css";
import Spinner from "./Spinner";
const { Header, Sider, Content } = Layout;
const DefaultLayout = ({children}) => {
const {cartItems,loading} = useSelector(state => state.rootReducer);
const {collapsed,setCollapsed} =useState(false);
const navigate =useNavigate();
const toggle = () => {
setCollapsed(
!collapsed,
);
};
//get data from database
useEffect(() =>{
localStorage.setItem("cartItems",JSON.stringify(cartItems));
},[cartItems]);
return (
<Layout>

Validations

backend > controllers > JS userController.js > submitForm > submitForm

```
const User = require('../models/User');
// Handle form submission
exports.submitForm = async (req, res) => {
  const { name, email, password } = req.body;
  // Validate form data
  const errors = {};
  if (!name) {
    errors.name = 'bill name is requires';
  }
  if (!quantity) {
    errors.quantity = 'quantity is required';
  } else if (!/\S+@\S+\.\S+/.test(quantity)) {
    errors.quantity = 'quantity is invalid';
  }
  if (!price) {
    errors.price = 'price is required';
  } else if (!price) {
    errors.price = 'price required';
  }
  try {
    // Create a new user
    const user = new User({ name, quantity, price });
    // Save the user to the database
    await user.save();
    // Return success response
    res.json({ message: 'Form submitted successfully' });
  } catch (error) {
    // Handle server-side errors
    console.error(error);
    res.status(500).json({ message: 'Internal server error' });
  }
```

PROBLEMS 30 OUTPUT DEBUG CONSOLE TERMINAL

```
GET /api/bills/get-bill 304 677.887 ms - -
GET /api/items/get-item 304 330.989 ms - -
GET /api/items/get-item 304 362.315 ms - -
```

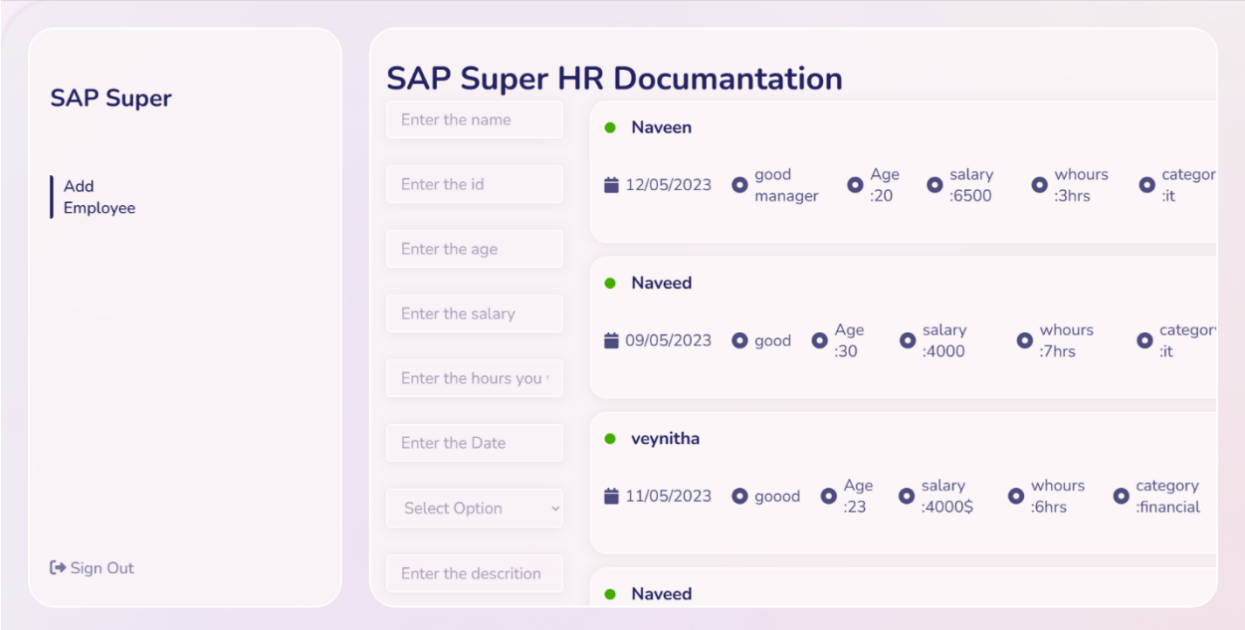
Model

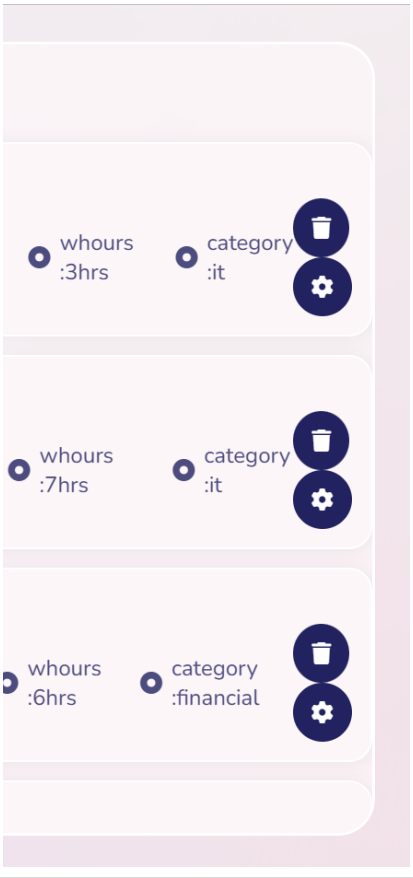
backend > models > JS billsModel.js > ...

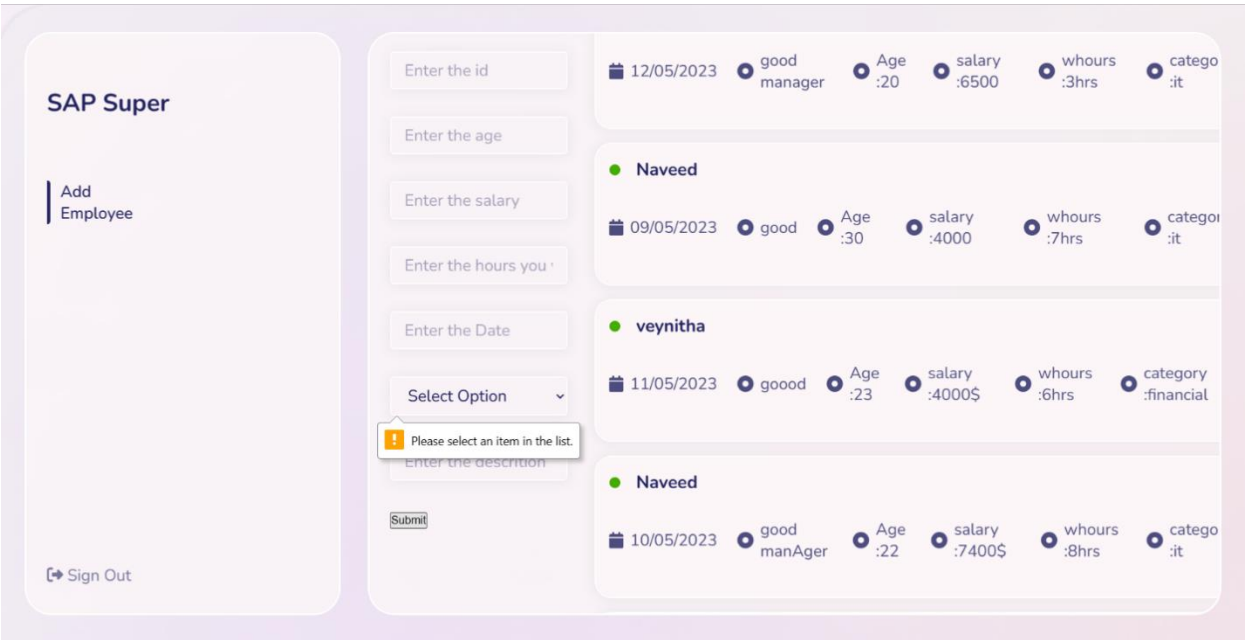
```
const mongoose = require("mongoose");
const billSchema = mongoose.Schema(
  {
    customername: {
      type: String,
      required: true,
    },
    customernumber: {
      type:String,
      required: true,
    },
    totalAmout: {
        type: Number,
        required: true,
    },
    reson: {
      type: String,
      required: true,
   },
    cartItems: {
        type:Array,
        required:true,
    },
  },
  { timestamp: true }
);
const Bills = mongoose.model("Bills",billSchema);
module.exports = Bills;

```

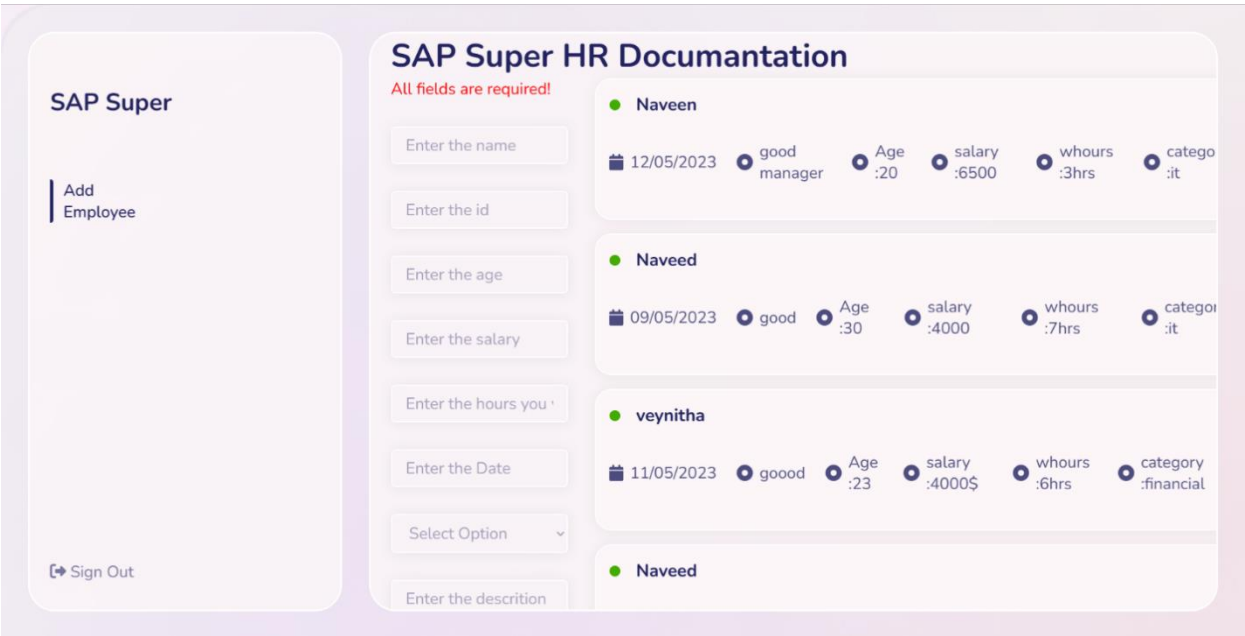
Human Resource Management System

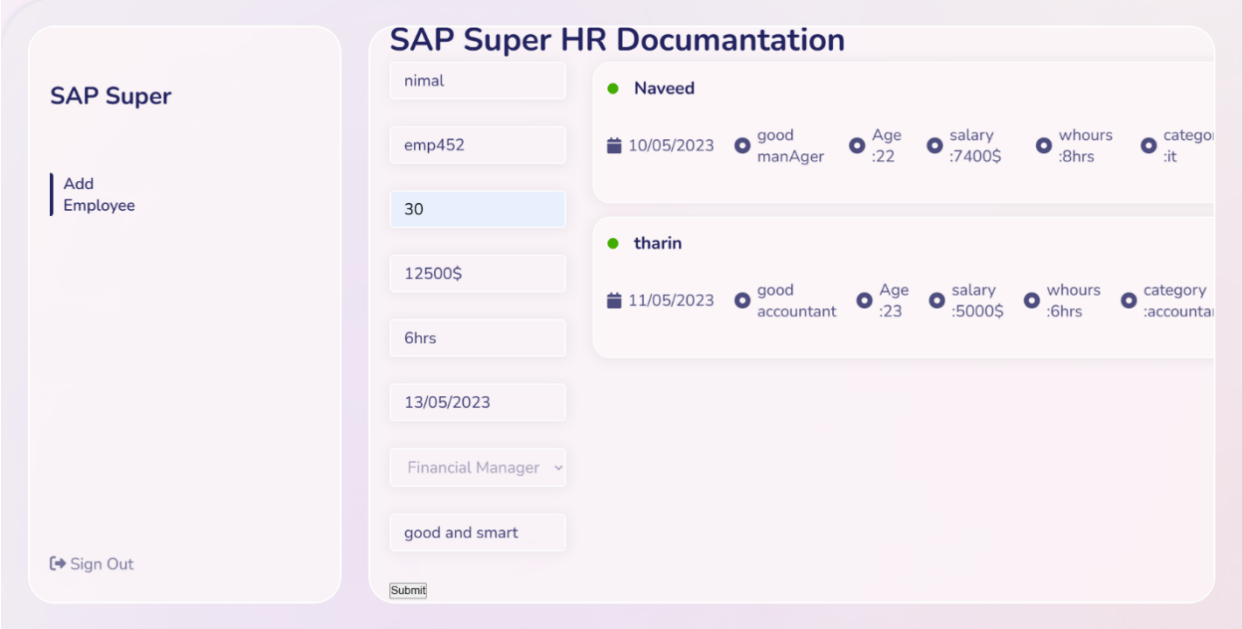
UI Interfaces

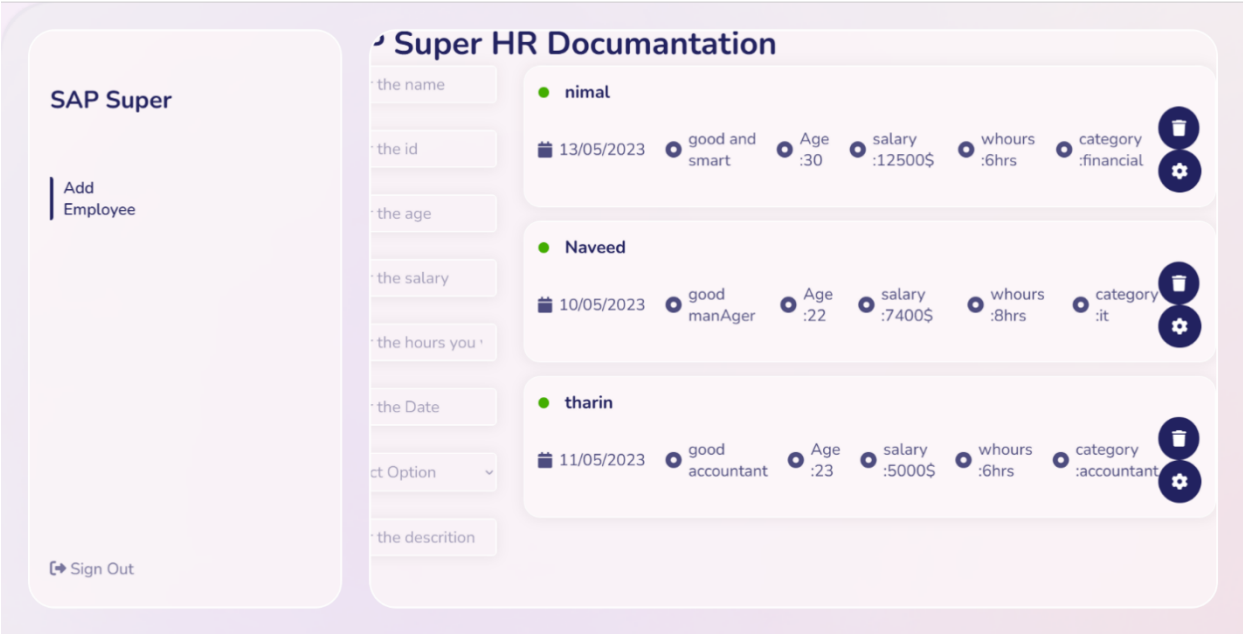


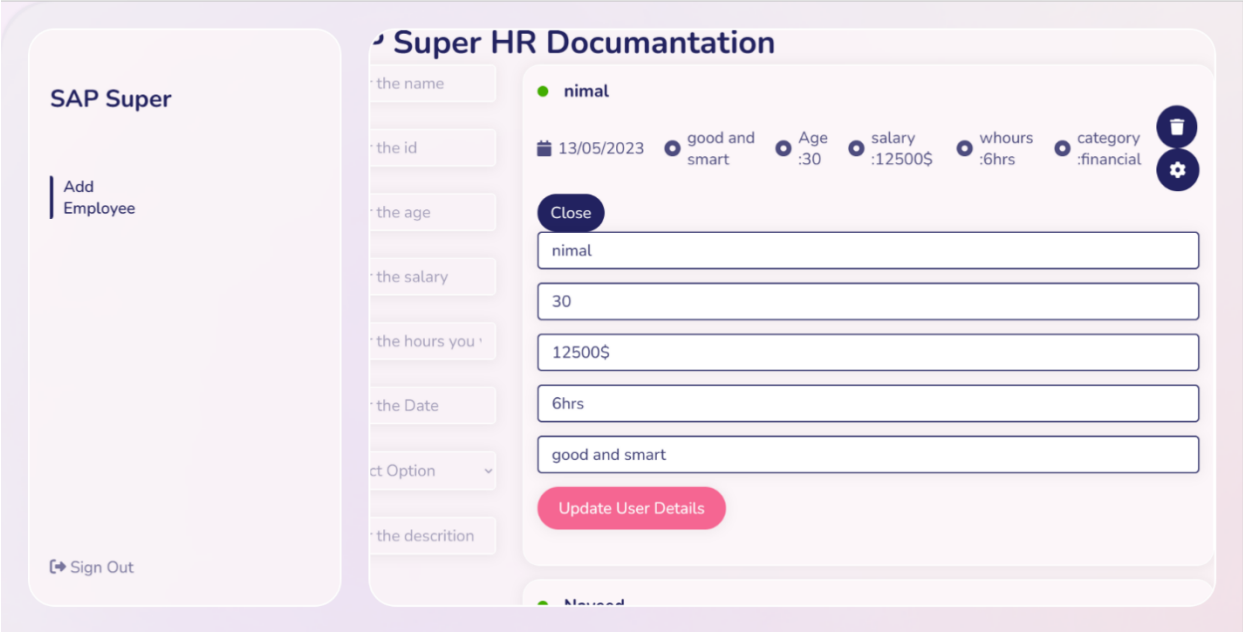


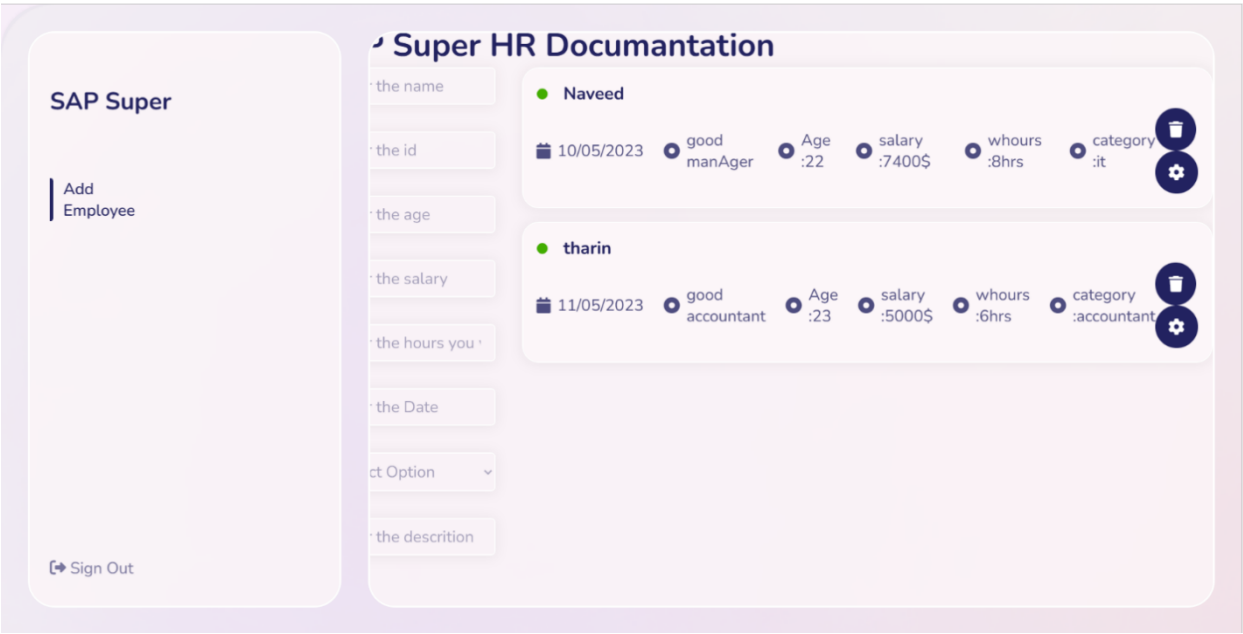

SAP Super
Add Employee
Sign Out
Enter the id
Enter the age
Enter the salary
Enter the hours you
Enter the Date
Select Option
Please select an item in the list.
Enter the description
Submit
12/05/2023 good manager Age :20 salary :6500 whours :3hrs catego :it
Naveed
09/05/2023 good Age :30 salary :4000 whours :7hrs categor :it
veynitha
11/05/2023 goood Age :23 salary :4000$ whours :6hrs category :financial
Naveed
10/05/2023 good manAger Age :22 salary :7400$ whours :8hrs catego :it


SAP Super
Add Employee
Sign Out
SAP Super HR Documantation
All fields are required!
Enter the name
Enter the id
Enter the age
Enter the salary
Enter the hours you
Enter the Date
Select Option
Enter the description
Naveen
12/05/2023 good manager Age :20 salary :6500 whours :3hrs catego :it
Naveed
09/05/2023 good Age :30 salary :4000 whours :7hrs categor :it
veynitha
11/05/2023 goood Age :23 salary :4000$ whours :6hrs category :financial
Naveed

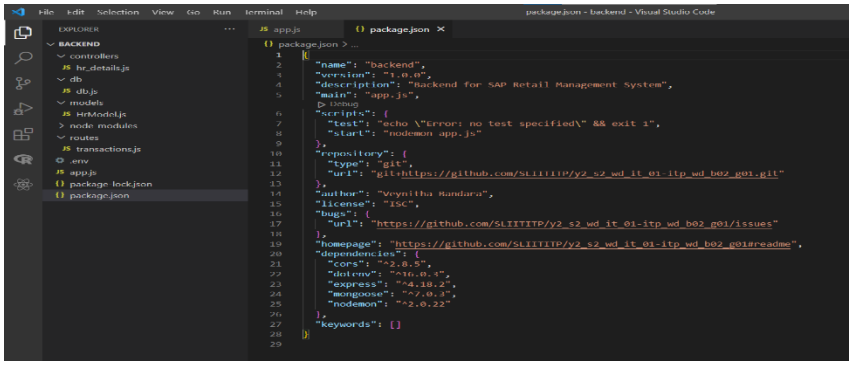

SAP Super
Add Employee
Sign Out
SAP Super HR Documantation
nimal
emp452
30
12500$
6hrs
13/05/2023
Financial Manager
good and smart
Submit
Naveed
10/05/2023
good manAger
Age :22
salary :7400$
whours :8hrs
tharin
11/05/2023
good accountant
Age :23
salary :5000$
whours :6hrs
category :accounta


SAP Super
Add Employee
Sign Out
Super HR Documantation
the name
the id
the age
the salary
the hours you
the Date
ct Option
the descrition
nimal
13/05/2023
good and smart
Age :30
salary :12500$
whours :6hrs
category :financial
Naveed
10/05/2023
good manAger
Age :22
salary :7400$
whours :8hrs
category :it
tharin
11/05/2023
good accountant
Age :23
salary :5000$
whours :6hrs
category :accountant

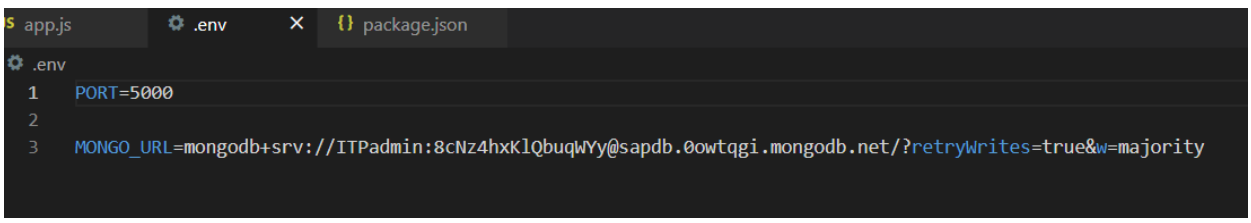

SAP Super
Add Employee
Sign Out
Super HR Documantation
the name
the id
the age
the salary
the hours you
the Date
ct Option
the descrition
nimal
13/05/2023
good and smart
Age :30
salary :12500$
whours :6hrs
category :financial
Close
nimal
30
12500$
6hrs
good and smart
Update User Details


SAP Super
Add Employee
Sign Out
Super HR Documantation
the name
the id
the age
the salary
the hours you
the Date
ct Option
the descrition
Naveed
10/05/2023
good manAger
Age :22
salary :7400$
whours :8hrs
category :it
tharin
11/05/2023
good accountant
Age :23
salary :5000$
whours :6hrs
category :accountant

Ui Codes


File Edit Selection View Go Run Terminal Help
package.json - backend - Visual Studio Code
EXPLORER
BACKEND
controllers
hr_details.js
db
db.js
models
HrModel.js
node_modules
routes
transactions.js
.env
app.js
package-lock.json
package.json
app.js
package.json
{
"name": "backend",
"version": "1.0.0",
"description": "Backend for SAP Retail Management System",
"main": "app.js",
"scripts": {
"test": "echo \"Error: no test specified\" && exit 1",
"start": "nodemon app.js"
},
"repository": {
"type": "git",
"url": "git+https://github.com/SLIITITP/y2_s2_wd_it_01-itp_wd_b02_g01.git"
},
"author": "Veynitha Randara",
"license": "ISC",
"bugs": {
"url": "https://github.com/SLIITITP/y2_s2_wd_it_01-itp_wd_b02_g01/issues"
},
"homepage": "https://github.com/SLIITITP/y2_s2_wd_it_01-itp_wd_b02_g01#readme",
"dependencies": {
"cors": "^2.8.5",
"dotenv": "^16.0.1",
"express": "^4.18.2",
"mongoose": "^7.0.3",
"nodemon": "^2.0.22"
},
"keywords": []
}


app.js
package.json
app.js > ...
const express = require("express");
const cors = require("cors");
const { db } = require("./db/db");
const { readdirSync } = require("fs");
const app = express();
require("dotenv").config();
const PORT = process.env.PORT;
//middlewares
app.use(express.json());
app.use(cors());
//routes
readdirSync("./routes").map((route) =>
app.use("/api/v1", require("./routes/" + route))
);
const server = () => {
db();
app.listen(PORT, () => {
console.log("listening to port:", PORT);
});
};
server();


app.js
.env
package.json
.env
PORT=5000
MONGO_URL=mongodb+srv://ITPadmin:8cNz4hxKlQbuqWYy@sapdb.0owtqgi.mongodb.net/?retryWrites=true&w=majority

app.js .env transactions.js package.json

routes > transactions.js > ...

```
const {
  addHr,
  getHrm,
  deleteHr,
  UpdateHrm,
} = require("../controllers/hr_details");

const router = require("express").Router();

router
  .post("/add-hr", addHr)
  .put("/update-hrm/:id", UpdateHrm)
  .get("/get-hrm", getHrm)
  .delete("/delete-hr/:id", deleteHr);

module.exports = router;

```

app.js .env transactions.js HrModel.js package.json

models > HrModel.js > ...

```
const mongoose = require('mongoose');


const HrSchema = new mongoose.Schema({
    id: {
        type: String,
        required: true,
        trim: true,
        maxLength: 50
    },
    name: {
        type: String,
        required: true,
        trim: true,
        maxLength: 50
    },
    age: {
        type: String,
        required: true,
        trim: true,
        maxLength: 50
    },

    salary: {
        type: String,
        required: true,
        trim: true,
        maxLength: 50
    },

    whours: {
        type: String,
        required: true,
        trim: true,
        maxLength: 50
    },

    date: {
```

app.js | .env | transactions.js | HrModel.js | package.json

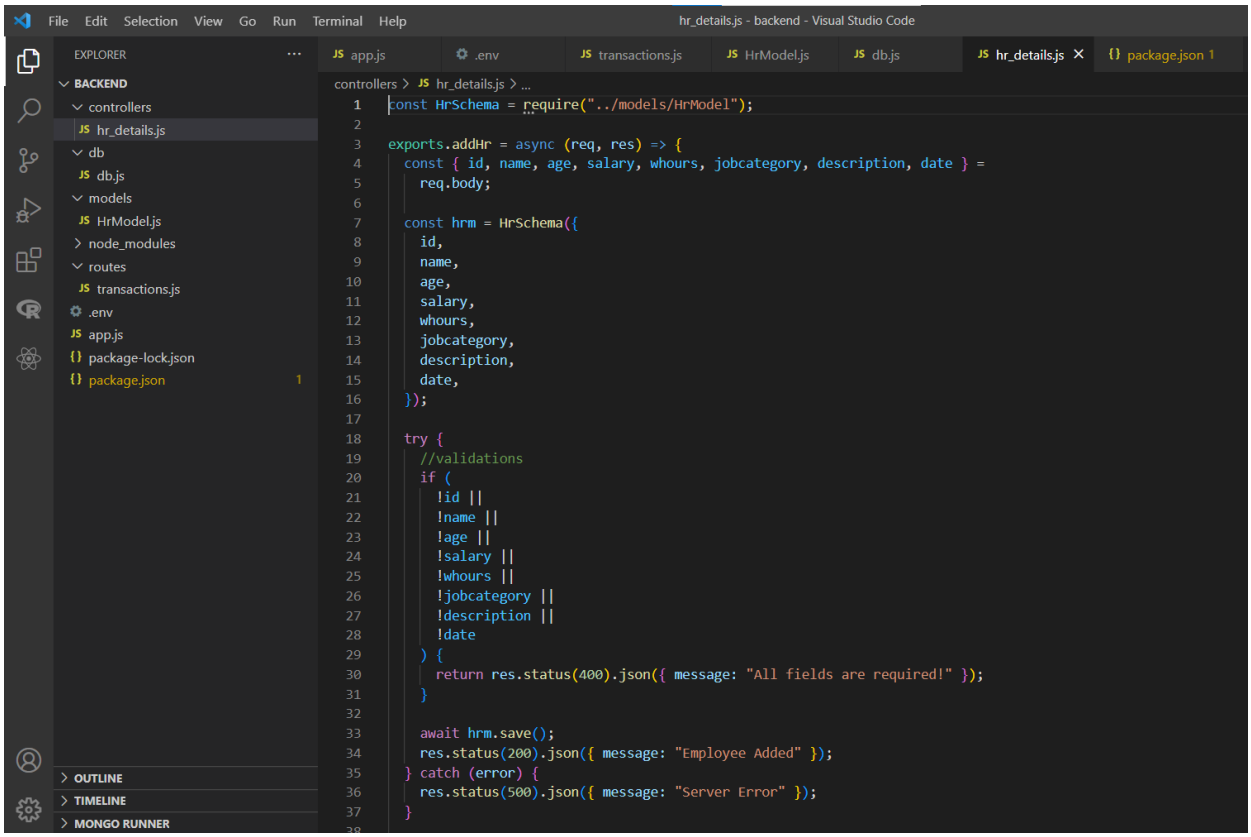
models > HrModel.js > ...

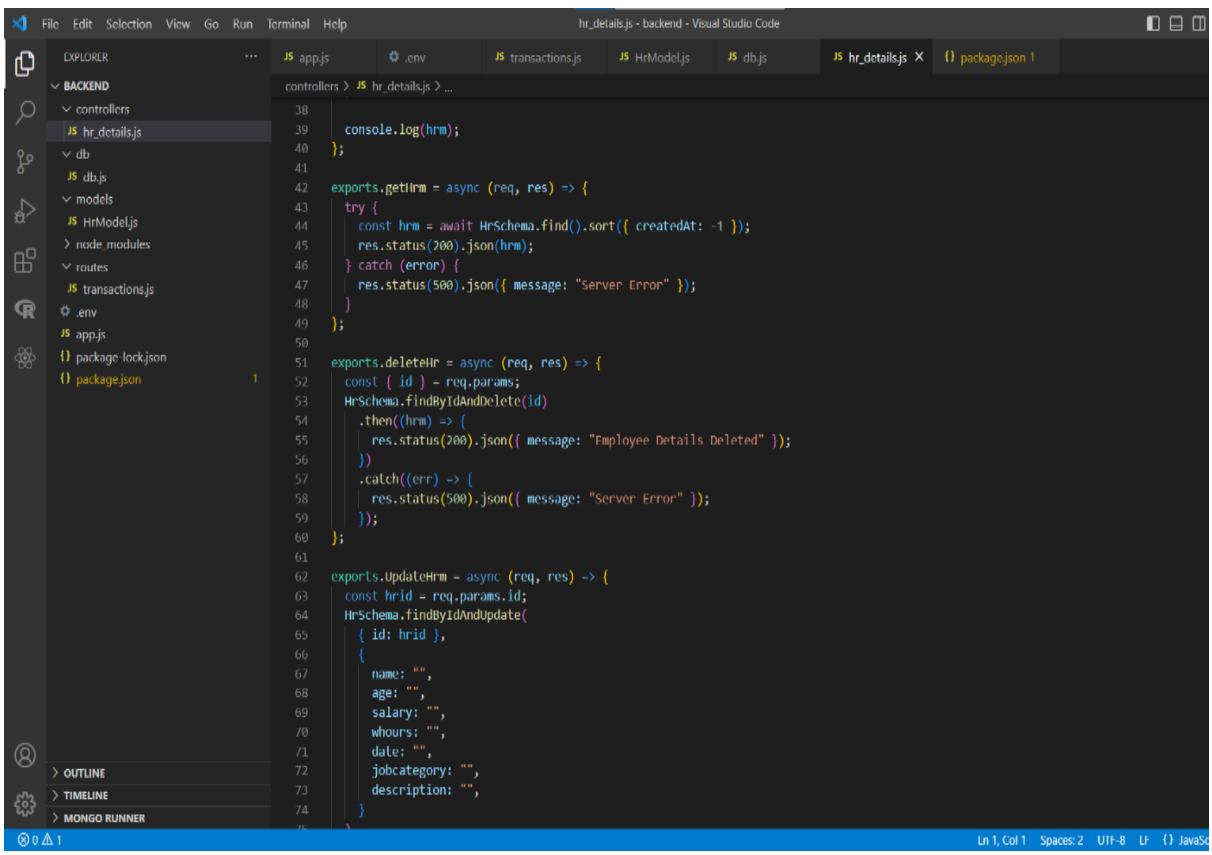
```
    whours: {
        type: String,
        required: true,
        trim: true,
        maxLength: 50
    },

    date: {
        type: Date,
        required: true,
        trim: true
    },
    jobcategory: {
        type: String,
        required: true,
        trim: true
    },
    description: {
        type: String,
        required: true,
        maxLength: 20,
        trim: true
    },
}, {timestamps: true})

module.exports = mongoose.model('Hr', HrSchema)
```

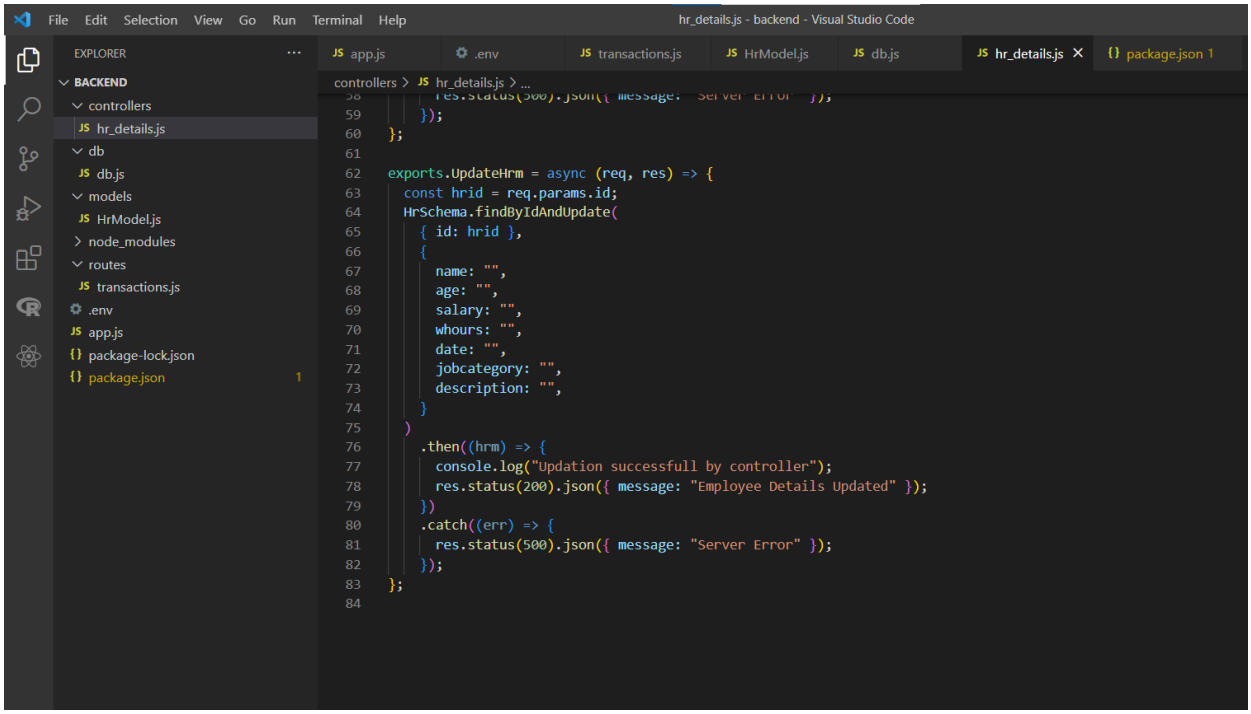
app.js | .env | transactions.js | HrModel.js | db.js | package.json 1

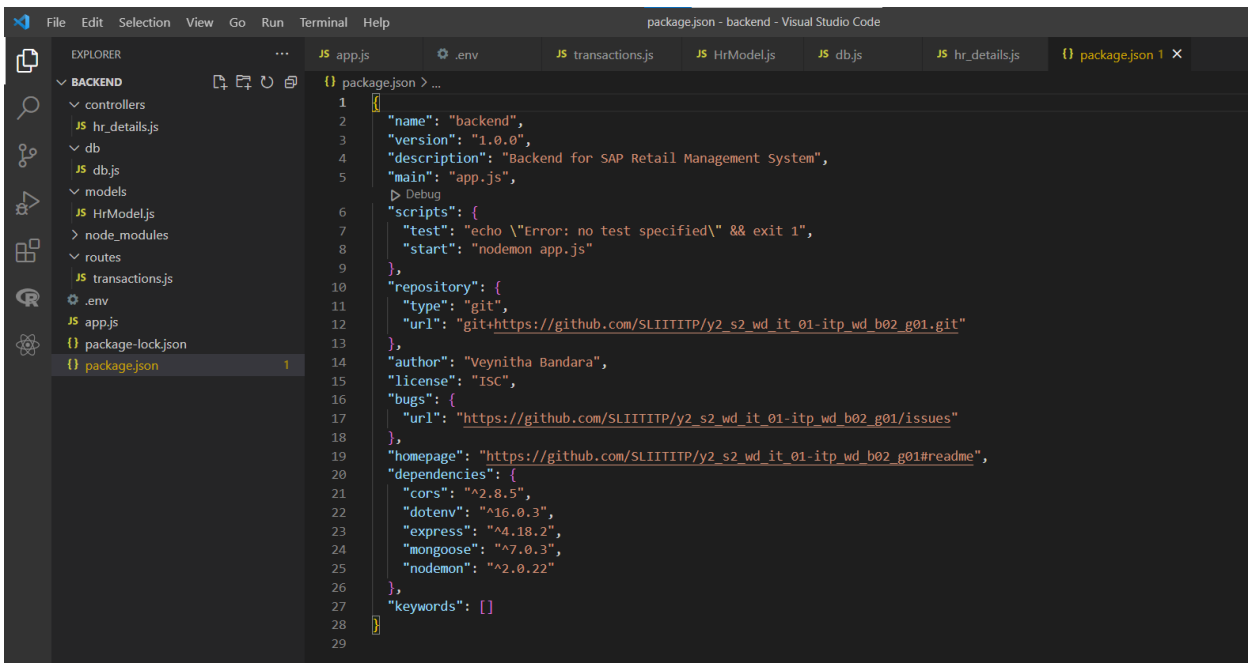
db > db.js > ...

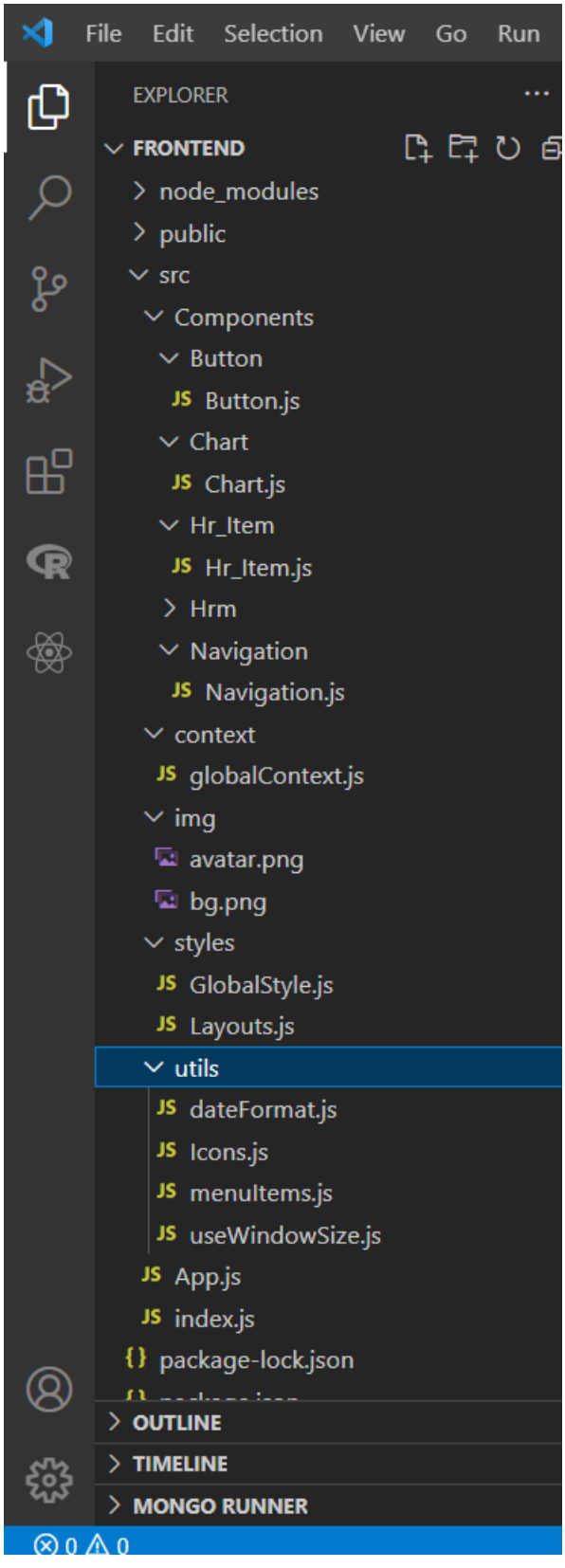
```
const mongoose = require('mongoose');

const db = async () => {
    try {
        mongoose.set('strictQuery', false)
        await mongoose.connect(process.env.MONGO_URL)
        console.log('Db Connected')
    } catch (error) {
        console.log('DB Connection Error');
    }
}

module.exports = {db}
```


File Edit Selection View Go Run Terminal Help
hr_details.js - backend - Visual Studio Code
EXPLORER
BACKEND
controllers
hr_details.js
db
db.js
models
HrModel.js
node_modules
routes
transactions.js
.env
app.js
package-lock.json
package.json
app.js
.env
transactions.js
HrModel.js
db.js
hr_details.js
package.json 1
controllers > hr_details.js > ...
const HrSchema = require("../models/HrModel");
exports.addHr = async (req, res) => {
const { id, name, age, salary, whours, jobcategory, description, date } =
req.body;
const hrm = HrSchema({
id,
name,
age,
salary,
whours,
jobcategory,
description,
date,
});
try {
//validations
if (
!id ||
!name ||
!age ||
!salary ||
!whours ||
!jobcategory ||
!description ||
!date
) {
return res.status(400).json({ message: "All fields are required!" });
}
await hrm.save();
res.status(200).json({ message: "Employee Added" });
} catch (error) {
res.status(500).json({ message: "Server Error" });
}
OUTLINE
TIMELINE
MONGO RUNNER


File Edit Selection View Go Run Terminal Help
hr_details.js - backend - Visual Studio Code
EXPLORER
BACKEND
controllers
hr_details.js
db
db.js
models
HrModel.js
node_modules
routes
transactions.js
.env
app.js
package-lock.json
package.json
app.js
.env
transactions.js
HrModel.js
db.js
hr_details.js
package.json 1
controllers > hr_details.js > ...
console.log(hrm);
};
exports.getHrm = async (req, res) => {
try {
const hrm = await HrSchema.find().sort({ createdAt: -1 });
res.status(200).json(hrm);
} catch (error) {
res.status(500).json({ message: "Server Error" });
}
};
exports.deleteHr = async (req, res) => {
const { id } = req.params;
HrSchema.findByIdAndDelete(id)
.then((hrm) => {
res.status(200).json({ message: "Employee Details Deleted" });
})
.catch((err) => {
res.status(500).json({ message: "Server Error" });
});
};
exports.UpdateHrm = async (req, res) => {
const hrid = req.params.id;
HrSchema.findByIdAndUpdate(
{ id: hrid },
{
name: "",
age: "",
salary: "",
whours: "",
date: "",
jobcategory: "",
description: "",
}
OUTLINE
TIMELINE
MONGO RUNNER
Ln 1, Col 1 Spaces: 2 UTF-8 LF


File Edit Selection View Go Run Terminal Help
hr_details.js - backend - Visual Studio Code
EXPLORER
BACKEND
controllers
hr_details.js
db
db.js
models
HrModel.js
node_modules
routes
transactions.js
.env
app.js
package-lock.json
package.json 1
app.js
.env
transactions.js
HrModel.js
db.js
hr_details.js
package.json 1
controllers > hr_details.js > ...
});
};
exports.UpdateHrm = async (req, res) => {
const hrid = req.params.id;
HrSchema.findByIdAndUpdate(
{ id: hrid },
{
name: "",
age: "",
salary: "",
whours: "",
date: "",
jobcategory: "",
description: "",
}
)
.then((hrm) => {
console.log("Updation successfull by controller");
res.status(200).json({ message: "Employee Details Updated" });
})
.catch((err) => {
res.status(500).json({ message: "Server Error" });
});
};


File Edit Selection View Go Run Terminal Help
package.json - backend - Visual Studio Code
EXPLORER
BACKEND
controllers
hr_details.js
db
db.js
models
HrModel.js
node_modules
routes
transactions.js
.env
app.js
package-lock.json
package.json 1
app.js
.env
transactions.js
HrModel.js
db.js
hr_details.js
package.json 1
package.json > ...
{
"name": "backend",
"version": "1.0.0",
"description": "Backend for SAP Retail Management System",
"main": "app.js",
Debug
"scripts": {
"test": "echo \"Error: no test specified\" && exit 1",
"start": "nodemon app.js"
},
"repository": {
"type": "git",
"url": "git+https://github.com/SLIITITP/y2_s2_wd_it_01-itp_wd_b02_g01.git"
},
"author": "Veynitha Bandara",
"license": "ISC",
"bugs": {
"url": "https://github.com/SLIITITP/y2_s2_wd_it_01-itp_wd_b02_g01/issues"
},
"homepage": "https://github.com/SLIITITP/y2_s2_wd_it_01-itp_wd_b02_g01#readme",
"dependencies": {
"cors": "^2.8.5",
"dotenv": "^16.0.3",
"express": "^4.18.2",
"mongoose": "^7.0.3",
"nodemon": "^2.0.22"
},
"keywords": []
}


File
Edit
Selection
View
Go
Run
EXPLORER
FRONTEND
node_modules
public
src
Components
Button
Button.js
Chart
Chart.js
Hr_Item
Hr_Item.js
Hrm
Navigation
Navigation.js
context
globalContext.js
img
avatar.png
bg.png
styles
GlobalStyle.js
Layouts.js
utils
dateFormat.js
Icons.js
menuItems.js
useWindowSize.js
App.js
index.js
package-lock.json
OUTLINE
TIMELINE
MONGO RUNNER
0 0

Button.js

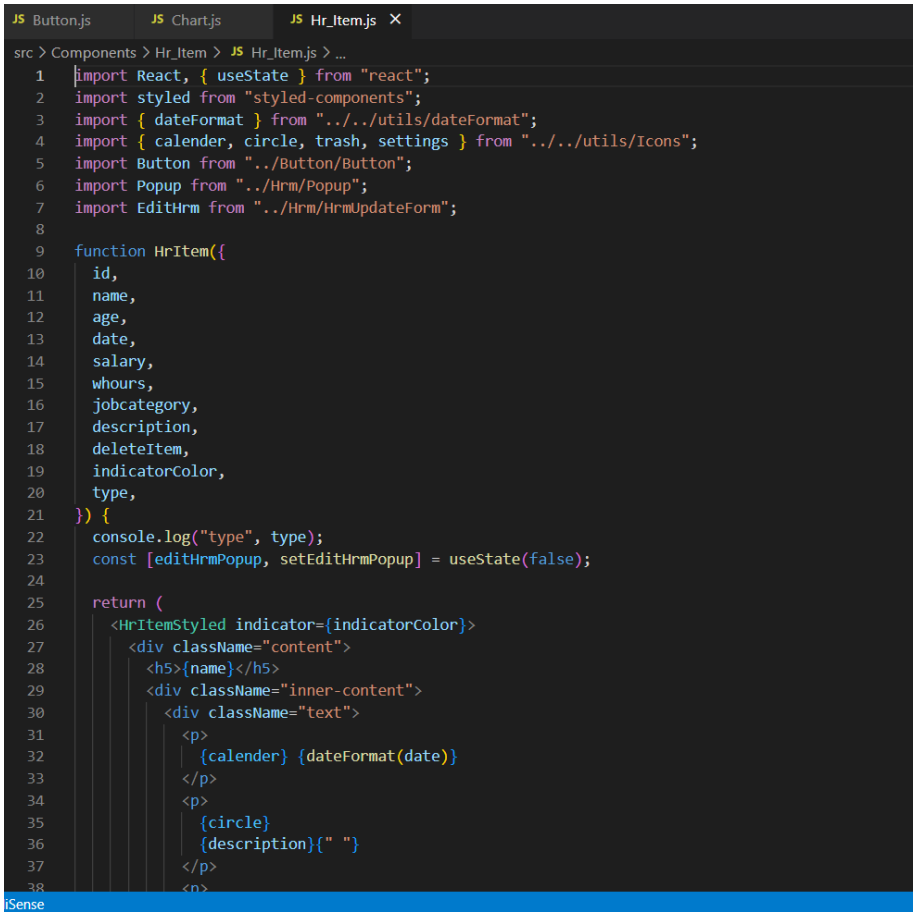
```
src > Components > Button > Button.js > ...
import React from "react";
import styled from "styled-components";

function Button({ name, icon, onClick, bg, bPad, color, bRad }) {
  return (
    <ButtonStyled
      style={{
        background: bg,
        padding: bPad,
        borderRadius: bRad,
        color: color,
      }}
      onClick={onClick}
    >
      {icon}
      {name}
    </ButtonStyled>
  );
}

const ButtonStyled = styled.button`
  outline: none;
  border: none;
  font-family: inherit;
  font-size: inherit;
  display: flex;
  align-items: center;
  gap: 0.5rem;
  cursor: pointer;
  transition: all 0.4s ease-in-out;
`;

export default Button;

```

Button.js | Chart.js

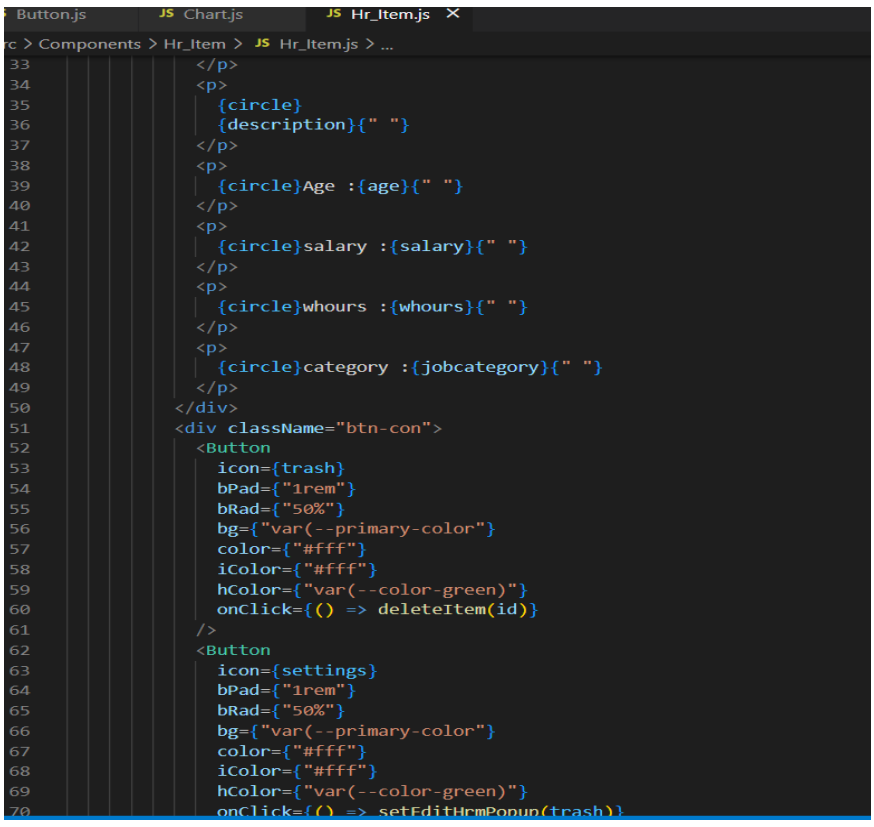
```
src > Components > Chart > Chart.js > ...
import React from 'react'
import {Chart as ChartJs,
    CategoryScale,
    LinearScale,
    PointElement,
    LineElement,
    Title,
    Tooltip,
    Legend,
    ArcElement,
} from 'chart.js'

import {Line} from 'react-chartjs-2'
import styled from 'styled-components'
import { useGlobalContext } from '../../context/globalContext'
import { dateFormat } from '../../utils/dateFormat'

ChartJs.register(
    CategoryScale,
    LinearScale,
    PointElement,
    LineElement,
    Title,
    Tooltip,
    Legend,
    ArcElement,
)

function Chart() {
    const {incomes, expenses} = useGlobalContext()

    const data = {
        labels: incomes.map((inc) =>{
            const {date} = inc
            return dateFormat(date)
        }),
        datasets: [
            {
```


Button.js
Chart.js
src > Components > Chart > Chart.js > ...
const data = {
labels: incomes.map((inc) =>{
const {date} = inc
return dateFormat(date)
}),
datasets: [
{
label: 'Income',
data: [
...incomes.map((income) => {
const {amount} = income
return amount
})
],
backgroundColor: 'green',
tension: .2
},
{
label: 'Expenses',
data: [
...expenses.map((expense) => {
const {amount} = expense
return amount
})
],
backgroundColor: 'red',
tension: .2
}
]
}
return (
<ChartStyled >
<Line data={data} />
</ChartStyled>
)
}


return (
<ChartStyled >
<Line data={data} />
</ChartStyled>
)
}
const ChartStyled = styled.div`
background: #FCF6F9;
border: 2px solid #FFFFFF;
box-shadow: 0px 1px 15px rgba(0, 0, 0, 0.06);
padding: 1rem;
border-radius: 20px;
height: 100%;
`;
export default Chart



```
Button.js  Chart.js  Hr_Item.js
src > Components > Hr_Item > Hr_Item.js > ...
import React, { useState } from "react";
import styled from "styled-components";
import { dateFormat } from "../../utils/dateFormat";
import { calender, circle, trash, settings } from "../../utils/Icons";
import Button from "../Button/Button";
import Popup from "../Hrm/Popup";
import EditHrm from "../Hrm/HrmUpdateForm";

function HrItem({
  id,
  name,
  age,
  date,
  salary,
  whours,
  jobcategory,
  description,
  deleteItem,
  indicatorColor,
  type,
}) {
  console.log("type", type);
  const [editHrmPopup, setEditHrmPopup] = useState(false);

  return (
    <HrItemStyled indicator={indicatorColor}>
      <div className="content">
        <h5>{name}</h5>
        <div className="inner-content">
          <div className="text">
            <p>
              {calender} {dateFormat(date)}
            </p>
            <p>
              {circle}
              {description}{" "}
            </p>
            <p>
iSense
```



```
Button.js  Chart.js  Hr_Item.js
rc > Components > Hr_Item > Hr_Item.js > ...
            </p>
            <p>
              {circle}
              {description}{" "}
            </p>
            <p>
              {circle}Age :{age}{" "}
            </p>
            <p>
              {circle}salary :{salary}{" "}
            </p>
            <p>
              {circle}whours :{whours}{" "}
            </p>
            <p>
              {circle}category :{jobcategory}{" "}
            </p>
          </div>
          <div className="btn-con">
            <Button
              icon={trash}
              bPad={"1rem"}
              bRad={"50%"}
              bg={"var(--primary-color"}
              color={"#fff"}
              iColor={"#fff"}
              hColor={"var(--color-green)"}
              onClick={() => deleteItem(id)}
            />
            <Button
              icon={settings}
              bPad={"1rem"}
              bRad={"50%"}
              bg={"var(--primary-color"}
              color={"#fff"}
              iColor={"#fff"}
              hColor={"var(--color-green)"}
              onClick={() => setEditHrmPopup(trash)}
```

useWindowSize.js | App.js | index.js

{} package.json > ...

```
23      "scripts": {
24        "start": "react-scripts start",
25        "build": "react-scripts build",
26        "test": "react-scripts test",
27        "eject": "react-scripts eject"
28      },
29      "eslintConfig": {
30        "extends": [
31          "react-app",
32          "react-app/jest"
33        ]
34      },
35      "browserslist": {
36        "production": [
37          ">0.2%",
38          "not dead",
39          "not op_mini all"
40        ],
41        "development": [
42          "last 1 chrome version",
43          "last 1 firefox version",
44          "last 1 safari version"
45        ]
46      }
47    }
48
```

useWindowSize.js | App.js | index.js | {} package.json 1

{} package.json > ...

```
1   {
2     "name": "frontend",
3     "version": "0.1.0",
4     "private": true,
5     "dependencies": {
6       "@testing-library/jest-dom": "^5.16.5",
7       "@testing-library/react": "^13.4.0",
8       "@testing-library/user-event": "^13.5.0",
9       "axios": "^1.4.0",
10      "chart.js": "^4.2.0",
11      "jspdf": "^2.5.1",
12      "mdb-react-ui-kit": "^6.0.0",
13      "moment": "^2.29.4",
14      "react": "^18.2.0",
15      "react-chartjs-2": "^5.2.0",
16      "react-datepicker": "^4.10.0",
17      "react-dom": "^18.2.0",
18      "react-router-dom": "^6.11.1",
19      "react-scripts": "5.0.1",
20      "styled-components": "^5.3.6",
21      "web-vitals": "^2.1.4"
22    },
      ▷ Debug
23    "scripts": {
24      "start": "react-scripts start",
25      "build": "react-scripts build",
26      "test": "react-scripts test",
27      "eject": "react-scripts eject"
28    },
29    "eslintConfig": {
30      "extends": [
31        "react-app",
32        "react-app/jest"
33      ]
34    },
35    "browserslist": {
36      "production": [
37        ">0.2%",
```

useWindowSize.js | App.js | index.js

src > index.js > ...

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import App from './App';
import { GlobalProvider } from './context/globalContext';
import { GlobalStyle } from './styles/GlobalStyle';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <GlobalStyle />
    <GlobalProvider>
      <App />
    </GlobalProvider>
  </React.StrictMode>
);


```

useWindowSize.js | App.js

src > App.js > ...

```
      <AppStyled bg={bg} className="App">
        <MainLayout>
          <Navigation active={active} setActive={setActive} />
          <main>{displayData()}</main>
        </MainLayout>
      </AppStyled>
    );
  }

  const AppStyled = styled.div`
    height: 100vh;
    background-image: url(${(props) => props.bg});
    position: relative;
    main {
      flex: 1;
      background: rgba(252, 246, 249, 0.78);
      border: 3px solid #ffffff;
      backdrop-filter: blur(4.5px);
      border-radius: 32px;
      overflow-x: hidden;
      &::-webkit-scrollbar {
        width: 0;
      }
    }
  `;

  export default App;

```

useWindowSize.js App.js

src > App.js > ...

```
import React, { useState, useMemo } from "react";
import styled from "styled-components";
import bg from "./img/bg.png";
import { MainLayout } from "./styles/Layouts";
import Navigation from "./Components/Navigation/Navigation";
import Hrm from "./Components/Hrm/Hrm";
import { useGlobalContext } from "./context/globalContext";
import EmployeeList from "./Components/Hrm/EmployeeList";

function App() {
  const [active, setActive] = useState(1);

  const global = useGlobalContext();
  console.log(global);

  const displayData = () => {
    switch (active) {
      case 1:
        return <Hrm />;
      case 2:
        return <EmployeeList />;
      // case 3:
      //   return <Hrm />;
      // case 4:
      //   return <EmployeeList />;
      default:
        return <Hrm />;
    }
  };

  return (
    <AppStyled bg={bg} className="App">
      <MainLayout>
        <Navigation active={active} setActive={setActive} />
        <main>{displayData()}</main>
      </MainLayout>
    </AppStyled>
```

useWindowSize.js App.js

src > App.js > ...

```
import React, { useState, useMemo } from "react";
import styled from "styled-components";
import bg from "./img/bg.png";
import { MainLayout } from "./styles/Layouts";
import Navigation from "./Components/Navigation/Navigation";
import Hrm from "./Components/Hrm/Hrm";
import { useGlobalContext } from "./context/globalContext";
import EmployeeList from "./Components/Hrm/EmployeeList";

function App() {
  const [active, setActive] = useState(1);

  const global = useGlobalContext();
  console.log(global);

  const displayData = () => {
    switch (active) {
      case 1:
        return <Hrm />;
      case 2:
        return <EmployeeList />;
      // case 3:
      //   return <Hrm />;
      // case 4:
      //   return <EmployeeList />;
      default:
        return <Hrm />;
    }
  };

  return (
    <AppStyled bg={bg} className="App">
      <MainLayout>
        <Navigation active={active} setActive={setActive} />
        <main>{displayData()}</main>
      </MainLayout>
    </AppStyled>
  );
```

useWindowSize.js

```
src > utils > useWindowSize.js > useWindowSize > useEffect() callback
import { useEffect, useState } from "react"


export const useWindowSize = () =>{
    const [size, setSize] = useState([window.innerWidth, window.innerHeight])

    useEffect(() => {
        const updateSize = () => {
            setSize([window.innerWidth, window.innerHeight])
        }
        window.addEventListener('resize', updateSize)

        return () => window.removeEventListener('resize', updateSize)
    }, [])

    return {
        width: size[0],
        height: size[1]
    }
}
```

Navigation.js | menuItems.js

```
src > utils > menuItems.js > ...
export const menuItems = [
  {
    id: 1,
    title: "Add Employee",
    link: "/addEmployee",
  },
];

```

Navigation.js | globalContext.js | GlobalStyle.js | Layouts.js | dateFormat.js | Icons.js

```
src > utils > Icons.js > ...
export const dashboard = <i className="fa-solid fa-chart-line"></i>
export const transactions = <i className="fa-solid fa-credit-card"></i>
export const categories = <i className="fa-solid fa-tags"></i>
export const accounts = <i className="fa-solid fa-wallet"></i>
export const settings = <i className="fa-solid fa-cog"></i>
export const logout = <i className="fa-solid fa-sign-out"></i>
export const trend = <i className="fa-solid fa-money-bill-trend-up"></i>
export const expenses = <i className="fa-solid fa-money-bill-transfer"></i>
export const money = <i className="fa-solid fa-money-bill"></i>
export const freelance = <i className ="fa-solid fa-earth-americas"></i>
export const stocks = <i className="fa-solid fa-arrow-trend-up"></i>
export const bitcoin = <i className="fa-brands fa-bitcoin"></i>
export const piggy = <i className="fa-solid fa-piggy-bank"></i>
export const yt = <i className="fa-brands fa-youtube"></i>
export const card = <i className="fa-brands fa-cc-visa"></i>
export const users = <i className="fa-solid fa-users-between-lines"></i>
export const dollar = <i className="fa-solid fa-dollar-sign"></i>
export const calender = <i className="fa-solid fa-calendar"></i>
export const comment = <i className="fa-solid fa-comment"></i>
export const plus = <i className="fa-solid fa-plus"></i>
export const trash = <i className="fa-solid fa-trash"></i>
export const signout = <i className="fa-solid fa-right-from-bracket"></i>
export const takeaway = <i className="fa-solid fa-utensils"></i>
export const clothing = <i className="fa-solid fa-shirt"></i>
export const book = <i className="fa-solid fa-book-open"></i>
export const food = <i className="fa-solid fa-bowl-food"></i>
export const medical = <i className="fa-solid fa-briefcase-medical"></i>
export const tv = <i className="fa-solid fa-tv"></i>
export const circle = <i className="fa-solid fa-circle-dot"></i>
```

Navigation.js | globalContext.js | GlobalStyle.js | Layouts.js | dateFormat.js

src > utils > dateFormat.js > ...

```
import moment from 'moment'


export const dateFormat = (date) =>{
    return moment(date).format('DD/MM/YYYY')
}
```

Navigation.js | globalContext.js | GlobalStyle.js | Layouts.js

src > styles > Layouts.js > ...

```
import styled from "styled-components";

export const MainLayout = styled.div`
    padding: 2rem;
    height: 100%;
    display: flex;
    gap: 2rem;
`;

export const InnerLayout = styled.div`
    padding: 2rem 1.5rem;
    width: 100%;
`;
```

Navigation.js | globalContext.js | GlobalStyle.js

src > styles > GlobalStyle.js > ...

```
        font-size: clamp(1rem, 1.5vw, 1.2rem);
        overflow: hidden;
        color: rgba(34, 34, 96, .6);
    }

    h1, h2, h3, h4, h5, h6{
        color: var(--primary-color);
    }

    .error{
        color: red;
        animation: shake 0.5s ease-in-out;
        @keyframes shake {
            0%{
                transform: translateX(0);
            }
            25%{
                transform: translateX(10px);
            }
            50%{
                transform: translateX(-10px);
            }
            75%{
                transform: translateX(10px);
            }
            100%{
                transform: translateX(0);
            }
        }
    }
`;
```

vigation.js | globalContext.js | GlobalStyle.js

styles > GlobalStyle.js > ...

```
import {createGlobalStyle} from 'styled-components'

export const GlobalStyle = createGlobalStyle`
    *{
        margin: 0;
        padding: 0;
        box-sizing: border-box;
        list-style: none;
    }

    :root{
        --primary-color: #222260;
        --primary-color2: 'color: rgba(34, 34, 96, .6)';
        --primary-color3: 'color: rgba(34, 34, 96, .4)';
        --color-green: #42AD00;
        --color-grey: #aaa;
        --color-accent: #F56692;
        --color-delete: #FF0000;
    }

    body{
        font-family: 'Nunito', sans-serif;
        font-size: clamp(1rem, 1.5vw, 1.2rem);
        overflow: hidden;
        color: rgba(34, 34, 96, .6);
    }

    h1, h2, h3, h4, h5, h6{
        color: var(--primary-color);
    }

    .error{
        color: red;
        animation: shake 0.5s ease-in-out;
        @keyframes shake {
            0%{
                transform: translateX(0);
            }
```

Navigation.js globalContext.js

src > context > globalContext.js > ...

```
    //updating function
    const UpdateHrm = async (id) => {
      const res = await axios.get(`${BASE_URL}update-hrm/${id}`);
    };

    return (
      <GlobalContext.Provider
        value={{
          hrm,
          addHr,
          getHrm,
          deleteHr,
          UpdateHrm,
          error,
          setError,
        }}
      >
        {children}
      </GlobalContext.Provider>
    );
  };

  export const useGlobalContext = () => {
    return useContext(GlobalContext);
  };
```

Navigation.js globalContext.js

src > context > globalContext.js > ...

```
import React, { useContext, useState } from "react";
import axios from "axios";

const BASE_URL = "http://localhost:5000/api/v1/";

const GlobalContext = React.createContext();

export const GlobalProvider = ({ children }) => {
  const [hrm, setHrm] = useState([]);
  const [error, setError] = useState(null);

  const addHr = async (hrm) => {
    const response = await axios.post(`${BASE_URL}add-hr`, hrm).catch((err) => {
      setError(err.response.data.message);
    });
    getHrm();
  };

  const getHrm = async () => {
    const response = await axios.get(`${BASE_URL}get-hrm`);
    setHrm(response.data);
    console.log(response.data);
  };

  const deleteHr = async (id) => {
    const res = await axios.delete(`${BASE_URL}delete-hr/${id}`);
    getHrm();
  };

  //updating function
  const UpdateHrm = async (id) => {
    const res = await axios.get(`${BASE_URL}update-hrm/${id}`);
  };
```

```
        .active{
            color: rgba(34, 34, 96, 1) !important;
            i{
                color: rgba(34, 34, 96, 1) !important;
            }
            &::before{
                content: "";
                position: absolute;
                left: 0;
                top: 0;
                width: 4px;
                height: 100%;
                background: #222260;
                border-radius: 0 10px 10px 0;
            }
        }
    `;

    export default Navigation
```

Navigation.js

src > Components > Navigation > Navigation.js > ...

```
    .menu-items{
        flex: 1;
        display: flex;
        flex-direction: column;
        li{
            display: grid;
            grid-template-columns: 40px auto;
            align-items: center;
            margin: .6rem 0;
            font-weight: 500;
            cursor: pointer;
            transition: all .4s ease-in-out;
            color: rgba(34, 34, 96, .6);
            padding-left: 1rem;
            position: relative;
            i{
                color: rgba(34, 34, 96, 0.6);
                font-size: 1.4rem;
                transition: all .4s ease-in-out;
            }
        }
    }

    .active{
        color: rgba(34, 34, 96, 1) !important;
        i{
            color: rgba(34, 34, 96, 1) !important;
        }
        &::before{
            content: "";
            position: absolute;
            left: 0;
            top: 0;
            width: 4px;
            height: 100%;
            background: #222260;
            border-radius: 0 10px 10px 0;
        }
```

Navigation.js

src > Components > Navigation > Navigation.js > ...

```
const NavStyled = styled.nav`
    padding: 2rem 1.5rem;
    width: 374px;
    height: 100%;
    background: rgba(252, 246, 249, 0.78);
    border: 3px solid #FFFFFF;
    backdrop-filter: blur(4.5px);
    border-radius: 32px;
    display: flex;
    flex-direction: column;
    justify-content: space-between;
    gap: 2rem;
    .user-con{
        height: 100px;
        display: flex;
        align-items: center;
        gap: 1rem;
        img{
            width: 80px;
            height: 80px;
            border-radius: 50%;
            object-fit: cover;
            background: #fcf6f9;
            border: 2px solid #FFFFFF;
            padding: .2rem;
            box-shadow: 0px 1px 17px rgba(0, 0, 0, 0.06);
        }
        h1{
            color: rgba(34, 34, 96, 1);
        }
        p{
            color: rgba(34, 34, 96, .6);
        }
    }
```

Navigation.js

src > Components > Navigation > Navigation.js > ...

```
import React, { useState } from 'react'
import styled from 'styled-components'
import { signout } from '../../utils/Icons'
import { menuItems } from '../../utils/menuItems'

function Navigation({active, setActive}) {

    return (
        <NavStyled>
            <div className="user-con">

                <div className="text">
                    <h1>SAP Super</h1>

                </div>
            </div>
            <ul className="menu-items">
                {menuItems.map((item) => {
                    return <li
                        key={item.id}
                        onClick={() => setActive(item.id)}
                        className={active === item.id ? 'active': ''}
                    >
                        {item.icon}
                        <span>{item.title}</span>
                    </li>
                })}
            </ul>
            <div className="bottom-nav">
                <li>
                    {signout} Sign Out
                </li>
            </div>
        </NavStyled>
    )
}
```

HrmForm.js | HrmList.js | HrmUpdateForm.js | Popup.js

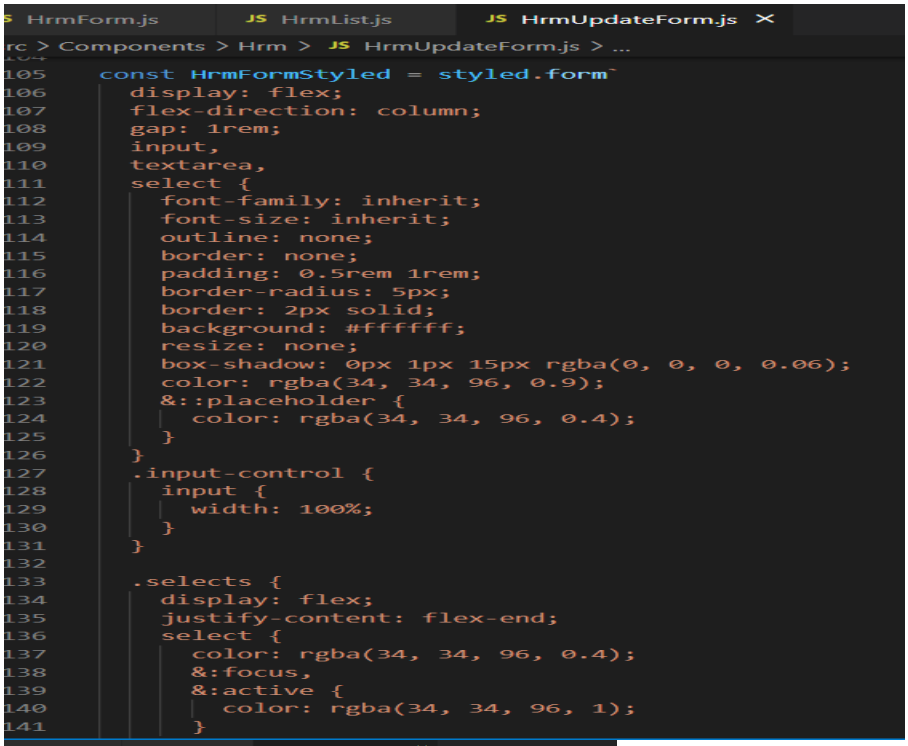
src > Components > Hrm > Popup.js > ...

```jsx
import React from "react";
import Button from "../Button/Button";

function Popup(props) {
  return props.trigger ? (
    <div className="popup">
      <div className="popup-inner">
        <Button
          name={"Close"}
          bPad={".6rem 1rem"}
          bRad={"25px"}
          bg={"var(--primary-color"}
          color={"#fff"}
          iColor={"#fff"}
          onClick={() => props.setTrigger(false)}
        ></Button>
        {props.children}
      </div>
    </div>
  ) : (
    ""
  );
}

export default Popup;
```

HrmForm.js | HrmList.js | HrmUpdateForm.js

src > Components > Hrm > HrmUpdateForm.js > ...

```
    }

    .selects {
      display: flex;
      justify-content: flex-end;
      select {
        color: rgba(34, 34, 96, 0.4);
        &:focus,
        &:active {
          color: rgba(34, 34, 96, 1);
        }
      }
    }

    .submit-btn {
      button {
        box-shadow: 0px 1px 15px rgba(0, 0, 0, 0.06);
        &:hover {
          background: var(--color-green) !important;
        }
      }
    }
  `;

export default EditHrm;
```



```
HrmForm.js  HrmList.js  HrmUpdateForm.js
src > Components > Hrm > HrmUpdateForm.js > ...
105     const HrmFormStyled = styled.form`
106       display: flex;
107       flex-direction: column;
108       gap: 1rem;
109       input,
110       textarea,
111       select {
112         font-family: inherit;
113         font-size: inherit;
114         outline: none;
115         border: none;
116         padding: 0.5rem 1rem;
117         border-radius: 5px;
118         border: 2px solid;
119         background: #ffffff;
120         resize: none;
121         box-shadow: 0px 1px 15px rgba(0, 0, 0, 0.06);
122         color: rgba(34, 34, 96, 0.9);
123         &::placeholder {
124           color: rgba(34, 34, 96, 0.4);
125         }
126       }
127       .input-control {
128         input {
129           width: 100%;
130         }
131       }
132
133       .selects {
134         display: flex;
135         justify-content: flex-end;
136         select {
137           color: rgba(34, 34, 96, 0.4);
138           &:focus,
139           &:active {
140             color: rgba(34, 34, 96, 1);
141           }
```

```
HrmForm.js  HrmList.js  HrmUpdateForm.js
src > Components > Hrm > HrmUpdateForm.js > ...
71
72        <div className="input-control">
73          <input
74            type="text"
75            value={eWhours}
76            name={"hour"}
77            placeholder="working hours"
78            onChange={(e) => setEWhours(e.target.value)}
79          />
80        </div>
81
82        <div className="input-control">
83          <input
84            type="text"
85            value={eDescription}
86            name={"description"}
87            placeholder="Enter Description"
88            onChange={(e) => setEDescription(e.target.value)}
89          />
90        </div>
91
92        <div className="submit-btn">
93          <Button
94            name={"Update User Details"}
95            bPad={".8rem 1.6rem"}
96            bRad={"30px"}
97            bg={"var(--color-accent"}
98            color={"#fff"}
99          />
100       </div>
101     </HrmFormStyled>
102   );
103 }
```


HrmForm.js
HrmList.js
HrmUpdateForm.js
src > Components > Hrm > HrmUpdateForm.js > ...
description: eDescription,
};
return (
<HrmFormStyled onSubmit={handleSubmit}>

<input
type="text"
value={eName}
name={"name"}
placeholder="Name"
onChange={(e) => setEName(e.target.value)}
/>

<input
type="text"
value={eAge}
name={"age"}
placeholder="Age"
onChange={(e) => setEAge(e.target.value)}
/>

<input
value={eSalary}
type="text"
name={"salary"}
placeholder={"Salary"}
onChange={(e) => setESalary(e.target.value)}
/>

<input
type="text"


HrmForm.js
HrmList.js
HrmUpdateForm.js
src > Components > Hrm > HrmUpdateForm.js > ...
import React, { useState } from "react";
import Button from "../Button/Button";
import { useGlobalContext } from "../../context/globalContext";
import styled from "styled-components";
function EditHrm({ id, name, age, salary, whours, description }) {
const { updateHrm } = useGlobalContext();
const [eId, setEid] = useState(id);
const [eName, setEName] = useState(name);
const [eAge, setEAge] = useState(age);
const [eSalary, setESalary] = useState(salary);
const [eWhours, setEWhours] = useState(whours);
const [eDescription, setEDescription] = useState(description);
const handleInputChange = (e) => {
console.log(e.target.value);
setEAge(e.target.value);
};
const handleSubmit = (e) => {
e.preventDefault();
if (!id || !eName || !eAge || !eSalary || !eWhours || !eDescription) {
alert("All fields required!!!");
} else {
updateHrm(id, data);
alert("Updated!");
}
};
const data = {
_id: eId,
name: eName,
age: eAge,
salary: eSalary,
whours: eWhours,
description: eDescription,
};

HrmForm.js | HrmList.js | HrmUpdateForm.js

src > Components > Hrm > HrmList.js > ...

```
import React, { useEffect } from "react";
import { useGlobalContext } from "../../context/globalContext";
import Display from "./Display";

function HrmList() {
  const { getHrm, hrms } = useGlobalContext();

  useEffect(() => {
    getHrm();
  }, [getHrm]);

  return (
    <div>
      <h2>All Employees</h2>
      <Display hrms={hrms} />
    </div>
  );
}

export default HrmList;
```

HrmForm.js

src > Components > Hrm > HrmForm.js > ...

```
            color: rgba(34, 34, 96, 1);
          }
        }
      }

      .submit-btn {
        button {
          box-shadow: 0px 1px 15px rgba(0, 0, 0, 0.06);
          &:hover {
            background: var(--color-green) !important;
          }
        }
      }
      .hrm-container {
        display: flex;
        flex-direction: row;
      }

      .hrm-list {
        width: 40%;
        padding-right: 20px;
      }

      .hrm-form {
        width: 60%;
      }
    `;
    export default HrmForm;
```

HrmForm.js

rc > Components > Hrm > HrmForm.js > ...

```
const HrmFormStyled = styled.form`
  display: flex;
  flex-direction: column;
  gap: 2rem;
  input,
  textarea,
  select {
    font-family: inherit;
    font-size: inherit;
    outline: none;
    border: none;
    padding: 0.5rem 1rem;
    border-radius: 5px;
    border: 2px solid #fff;
    background: transparent;
    resize: none;
    box-shadow: 0px 1px 15px rgba(0, 0, 0, 0.06);
    color: rgba(34, 34, 96, 0.9);
    &::placeholder {
      color: rgba(34, 34, 96, 0.4);
    }
  }
  .input-control {
    input {
      width: 100%;
    }
  }

  .selects {
    display: flex;
    justify-content: flex-end;
    select {
      color: rgba(34, 34, 96, 0.4);
      &:focus,
      &:active {
        color: rgba(34, 34, 96, 1);
      }
```

HrmForm.js

src > Components > Hrm > HrmForm.js > ...

```
        />
      </div>
      <div className="selects input-control">
        <select
          required
          value={jobcategory}
          name="jobcategory"
          id="jobcategory"
          onChange={handleInput("jobcategory")}
        >
          <option value="" disabled>
            Select Option
          </option>
          <option value="inventry">Inventry Manager</option>
          <option value="it">IT Manager</option>
          <option value="financial">Financial Manager</option>
          <option value="supply">Supply Manager</option>
          <option value="accountant">Accountant</option>
        </select>
      </div>
      <div className="input-control">
        <input
          type="text"
          value={description}
          name={"description"}
          placeholder="Enter the descrition"
          onChange={handleInput("description")}
        />
      </div>
      <div className="button">
        <button type="submit">Submit</button>
      </div>
    </HrmFormStyled>
  );
}

const HrmFormStyled = styled.form`
```

HrmForm.js

rc > Components > Hrm > HrmForm.js > ...

```
      <div className="input-control">
        <input
          type="text"
          value={age}
          name={"age"}
          placeholder="Enter the age"
          onChange={handleInput("age")}
        />
      </div>
      <div className="input-control">
        <input
          type="text"
          value={salary}
          name={"salary"}
          placeholder="Enter the salary"
          onChange={handleInput("salary")}
        />
      </div>
      <div className="input-control">
        <input
          type="text"
          value={whours}
          name={"whours"}
          placeholder="Enter the hours you worked"
          onChange={handleInput("whours")}
        />
      </div>
      <div className="input-control">
        <DatePicker
          id="date"
          placeholderText="Enter the Date"
          selected={date}
          dateFormat="dd/MM/yyyy"
          onChange={(date) => {
            setInputState({ ...inputState, date: date });
          }}
        />
```

HrmForm.js

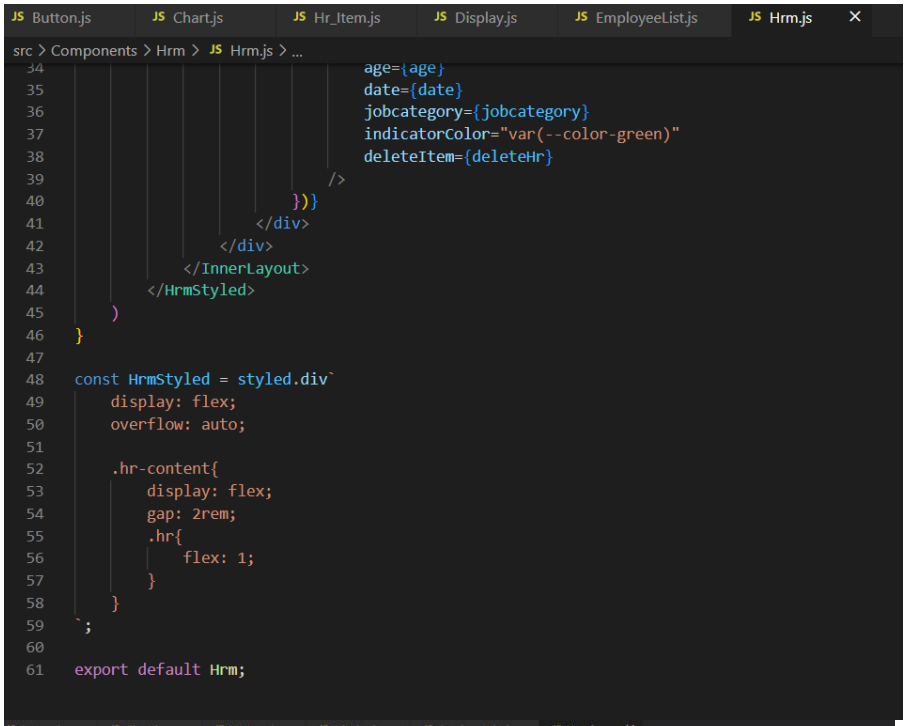
c > Components > Hrm > JS HrmForm.js > ...

```
31
32      setInputState({
33        id: "",
34        name: "",
35        age: "",
36        salary: "",
37        whours: "",
38        date: "",
39        jobcategory: "",
40        description: "",
41      });
42      addHr(inputState);
43    };
44
45    return (
46      <HrmFormStyled onSubmit={handleSubmit}>
47        {error && <p className="error">{error}</p>}
48        <div className="input-control">
49          <input
50            type="text"
51            value={name}
52            name={"name"}
53            placeholder="Enter the name"
54            onChange={handleInput("name")}
55          />
56        </div>
57        <div className="input-control">
58          <input
59            type="text"
60            value={id}
61            name={"id"}
62            placeholder="Enter the id"
63            onChange={handleInput("id")}
64          />
```

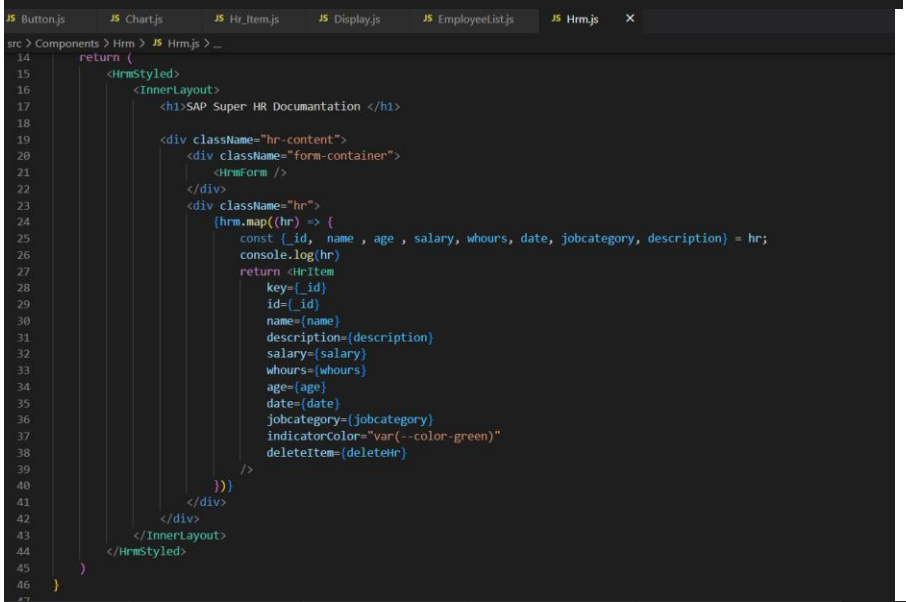
HrmForm.js

src > Components > Hrm > JS HrmForm.js > ...

```
1   import React, { useState } from "react";
2   import styled from "styled-components";
3   import DatePicker from "react-datepicker";
4   import "react-datepicker/dist/react-datepicker.css";
5   import { useGlobalContext } from "../../context/globalContext";
6   import Button from "../Button/Button";
7
8   function HrmForm() {
9     const { addHr, error, setError, Advertisements } = useGlobalContext();
10    const [inputState, setInputState] = useState({
11      id: "",
12      name: "",
13      age: "",
14      salary: "",
15      whours: "",
16      date: "",
17      jobcategory: "",
18      description: "",
19    });
20
21    const { id, name, age, salary, whours, date, jobcategory, description } =
22      inputState;
23
24    const handleInput = (name) => (e) => {
25      setInputState({ ...inputState, [name]: e.target.value });
26      setError("");
27    };
28
29    const handleSubmit = (e) => {
30      e.preventDefault();
```


Button.js
Chart.js
Hr_Item.js
Display.js
EmployeeList.js
Hrm.js
src > Components > Hrm > Hrm.js > ...
age={age}
date={date}
jobcategory={jobcategory}
indicatorColor="var(--color-green)"
deleteItem={deleteHr}
/>
})}

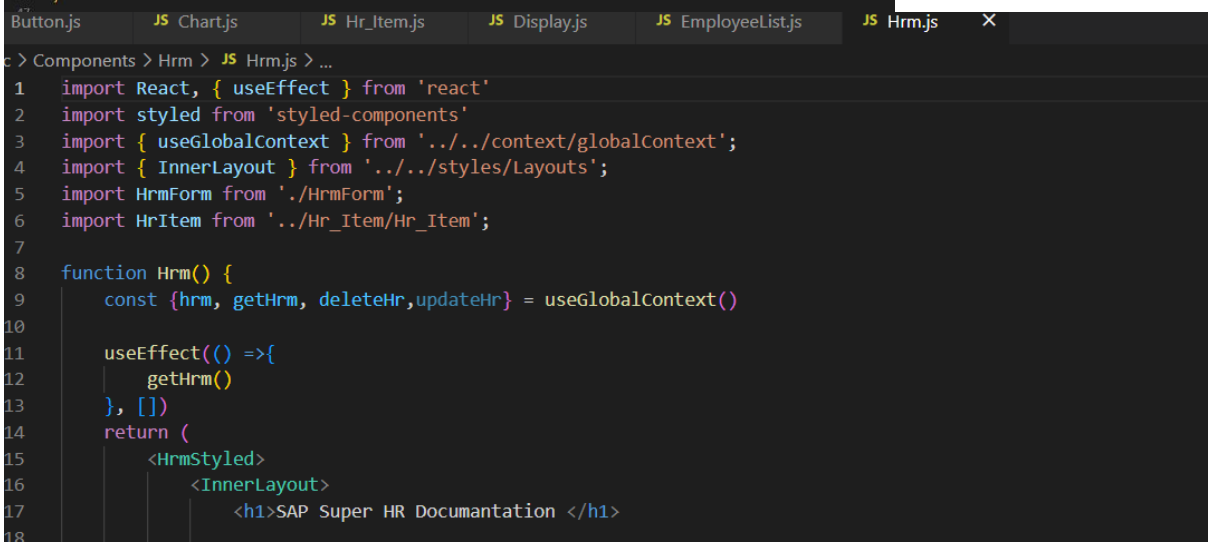
</InnerLayout>
</HrmStyled>
)
}
const HrmStyled = styled.div`
display: flex;
overflow: auto;
.hr-content{
display: flex;
gap: 2rem;
.hr{
flex: 1;
}
}
`;
export default Hrm;


Button.js
Chart.js
Hr_Item.js
Display.js
EmployeeList.js
Hrm.js
src > Components > Hrm > Hrm.js > ...
return (
<HrmStyled>
<InnerLayout>
<h1>SAP Super HR Documantation </h1>

<HrmForm />

{hrm.map((hr) => {
const {_id, name , age , salary, whours, date, jobcategory, description} = hr;
console.log(hr)
return <HrItem
key={_id}
id={_id}
name={name}
description={description}
salary={salary}
whours={whours}
age={age}
date={date}
jobcategory={jobcategory}
indicatorColor="var(--color-green)"
deleteItem={deleteHr}
/>
})}

</InnerLayout>
</HrmStyled>
)
}


Button.js
Chart.js
Hr_Item.js
Display.js
EmployeeList.js
Hrm.js
Components > Hrm > Hrm.js > ...
import React, { useEffect } from 'react'
import styled from 'styled-components'
import { useGlobalContext } from '../../context/globalContext';
import { InnerLayout } from '../../styles/Layouts';
import HrmForm from './HrmForm';
import HrItem from '../Hr_Item/Hr_Item';
function Hrm() {
const {hrm, getHrm, deleteHr,updateHr} = useGlobalContext()
useEffect(() =>{
getHrm()
}, [])
return (
<HrmStyled>
<InnerLayout>
<h1>SAP Super HR Documantation </h1>

Button.js | Chart.js | Hr_Item.js | Display.js | EmployeeList.js

src > Components > Hrm > EmployeeList.js > ...

```
const EmployeeListStyled = styled.div`
  h2 {
    margin-bottom: 1rem;
  }
  .employee-table {
    display: grid;
    grid-template-columns: repeat(8, 1fr);
    gap: 1rem;
    background: #fff;
    border-radius: 5px;
    overflow: hidden;
  }
  .employee-row {
    display: grid;
    grid-template-columns: repeat(8, 1fr);
    padding: 1rem;
    border-bottom: 1px solid #ccc;
    align-items: center;
    font-size: 1rem;
    font-weight: 500;
    color: #222;
    &:hover {
      background: #f9f9f9;
    }
    &.employee-header {
      background: #f2f2f2;
      font-size: 1.2rem;
      font-weight: 600;
      color: #333;
      border-bottom: 2px solid #ccc;
    }
  }
`;

export default EmployeeList;
```

Button.js | Chart.js | Hr_Item.js | Display.js | EmployeeList.js

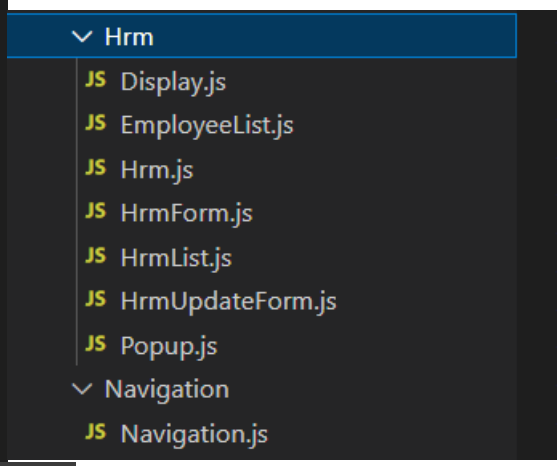
src > Components > Hrm > EmployeeList.js > ...

```
import React from "react";
import styled from "styled-components";

function EmployeeList({ employees }) {
  return (
    <EmployeeListStyled>
      <h2>Employee List</h2>
      <div className="employee-table">
        <div className="employee-row employee-header">
          <div>ID</div>
          <div>Name</div>
          <div>Age</div>
          <div>Salary</div>
          <div>Hours Worked</div>
          <div>Date</div>
          <div>Job Category</div>
          <div>Description</div>
        </div>
        {employees.map((employee) => (
          <div className="employee-row" key={employee.id}>
            <div>{employee.id}</div>
            <div>{employee.name}</div>
            <div>{employee.age}</div>
            <div>{employee.salary}</div>
            <div>{employee.whours}</div>
            <div>{employee.date}</div>
            <div>{employee.jobcategory}</div>
            <div>{employee.description}</div>
          </div>
        ))}
      </div>
    </EmployeeListStyled>
  );
}
```

Button.js | Chart.js | Hr_Item.js | Display.js

src > Components > Hrm > Display.js > ...

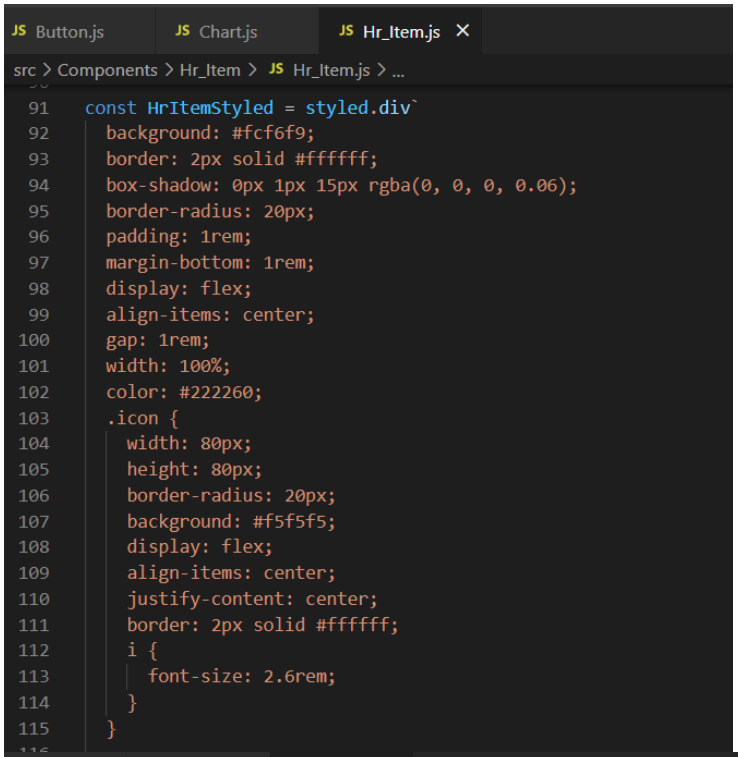
```jsx
import React from "react";

function Display(props) {
  const { hrms } = props;

  return (
    <ul>
      {hrms.map((hrm) => {
        return (
          <li key={hrm._id}>
            <h3>{hrm.name}</h3>
            <p>ID: {hrm.id}</p>
            <p>Age: {hrm.age}</p>
            <p>Salary: {hrm.salary}</p>
            <p>Working Hours: {hrm.whours}</p>
            <p>Job Category: {hrm.jobcategory}</p>
            <p>Description: {hrm.description}</p>
            <p>Date: {hrm.date}</p>
          </li>
        );
      })}
    </ul>
  );
}

export default Display;
```

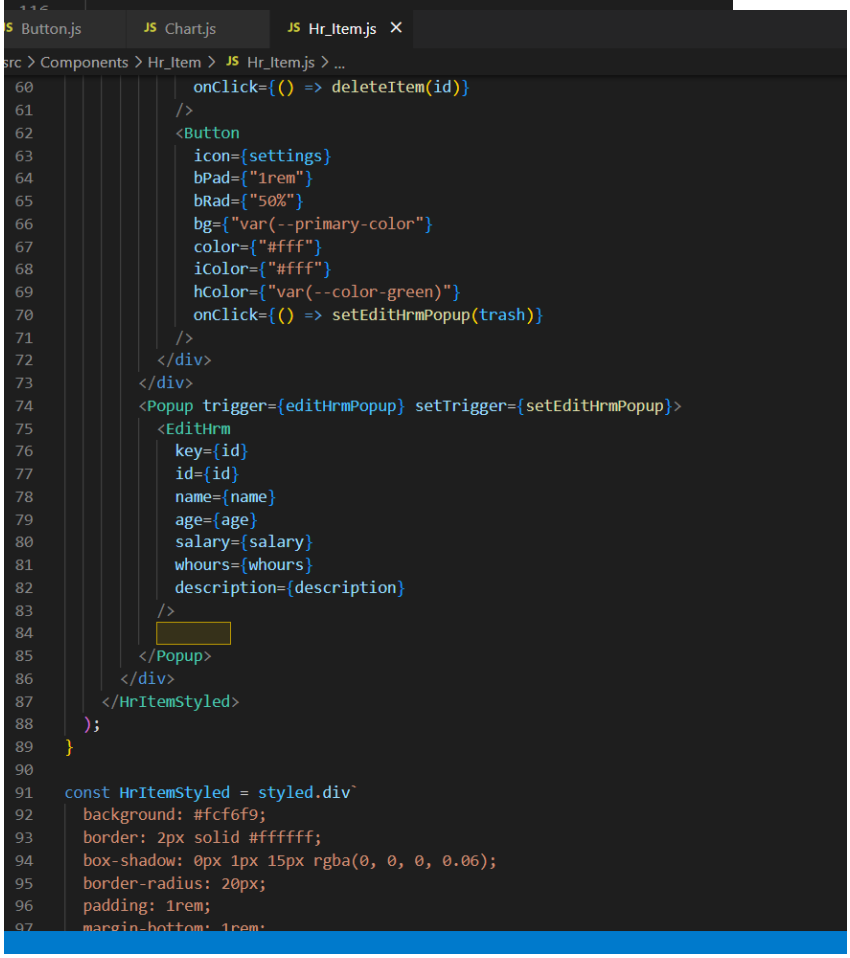


Button.js | Chart.js | Hr_Item.js

src > Components > Hr_Item > Hr_Item.js > ...

```scss
.content {
  flex: 1;
  display: flex;
  flex-direction: column;
  gap: 0.2rem;
  h5 {
    font-size: 1.3rem;
    padding-left: 2rem;
    position: relative;
    &::before {
      content: "";
      position: absolute;
      left: 0;
      top: 50%;
      transform: translateY(-50%);
      width: 0.8rem;
      height: 0.8rem;
      border-radius: 50%;
      background: ${(props) => props.indicator};
    }
  }

  .inner-content {
    display: flex;
    justify-content: space-between;
    align-items: center;
    .text {
      display: flex;
      align-items: center;
      gap: 1.5rem;
      p {
        display: flex;
        align-items: center;
        gap: 0.5rem;
        color: var(--primary-color);
        opacity: 0.8;
      }
```


Button.js
Chart.js
Hr_Item.js
src > Components > Hr_Item > Hr_Item.js > ...
const HrItemStyled = styled.div`
background: #fcf6f9;
border: 2px solid #ffffff;
box-shadow: 0px 1px 15px rgba(0, 0, 0, 0.06);
border-radius: 20px;
padding: 1rem;
margin-bottom: 1rem;
display: flex;
align-items: center;
gap: 1rem;
width: 100%;
color: #222260;
.icon {
width: 80px;
height: 80px;
border-radius: 20px;
background: #f5f5f5;
display: flex;
align-items: center;
justify-content: center;
border: 2px solid #ffffff;
i {
font-size: 2.6rem;
}
}

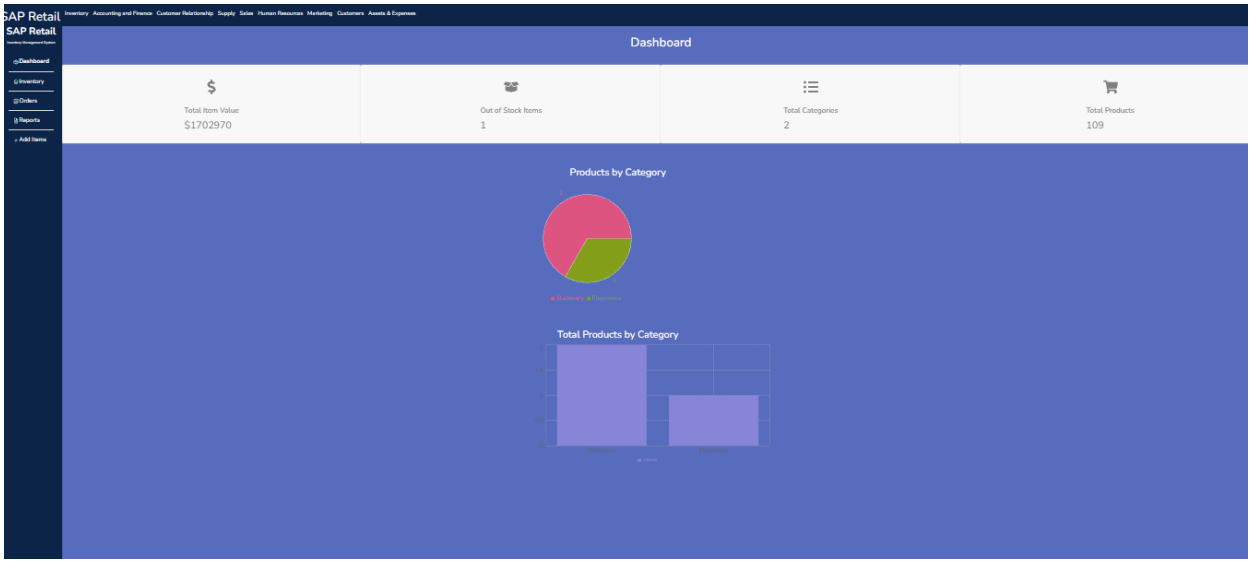

Button.js
Chart.js
Hr_Item.js
src > Components > Hr_Item > Hr_Item.js > ...
onClick={() => deleteItem(id)}
/>
<Button
icon={settings}
bPad={"1rem"}
bRad={"50%"}
bg={"var(--primary-color"}
color={"#fff"}
iColor={"#fff"}
hColor={"var(--color-green)"}
onClick={() => setEditHrmPopup(trash)}
/>

<Popup trigger={editHrmPopup} setTrigger={setEditHrmPopup}>
<EditHrm
key={id}
id={id}
name={name}
age={age}
salary={salary}
whours={whours}
description={description}
/>
</Popup>

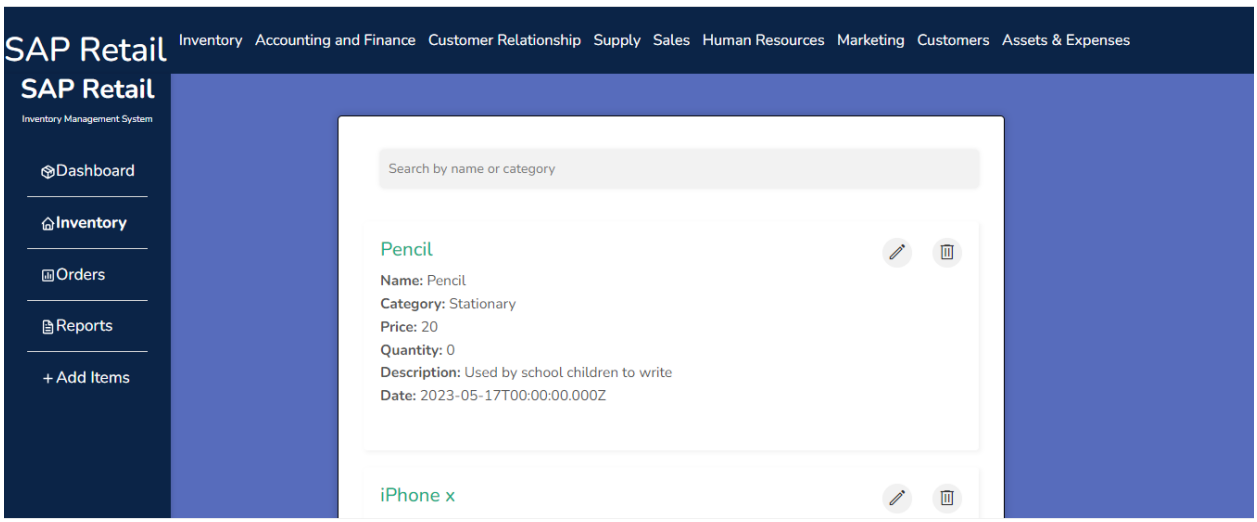
</HrItemStyled>
);
}
const HrItemStyled = styled.div`
background: #fcf6f9;
border: 2px solid #ffffff;
box-shadow: 0px 1px 15px rgba(0, 0, 0, 0.06);
border-radius: 20px;
padding: 1rem;

Inventory Management System

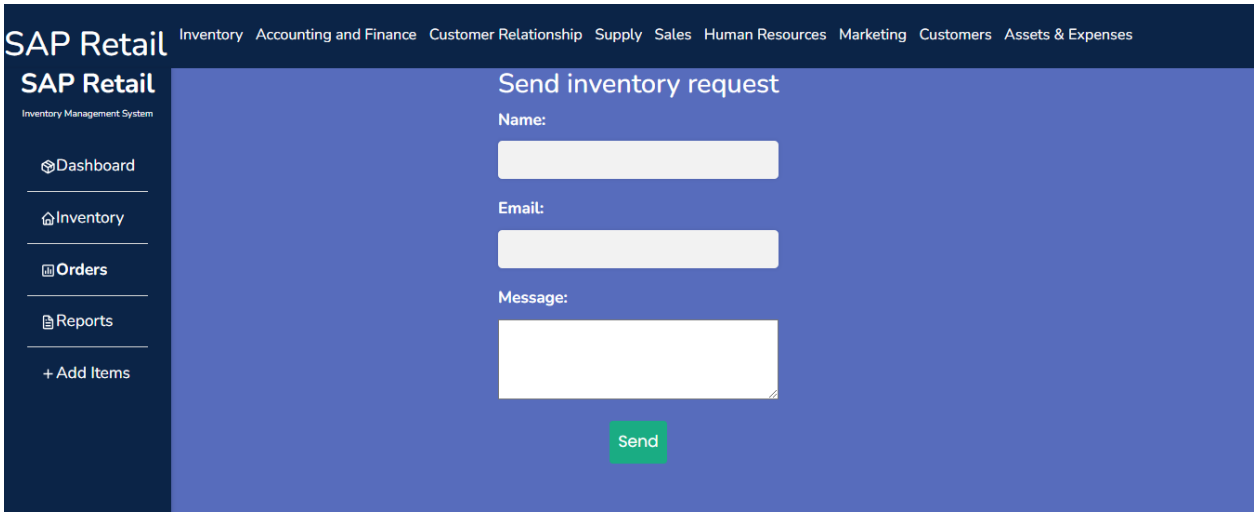
Inventory dashboard



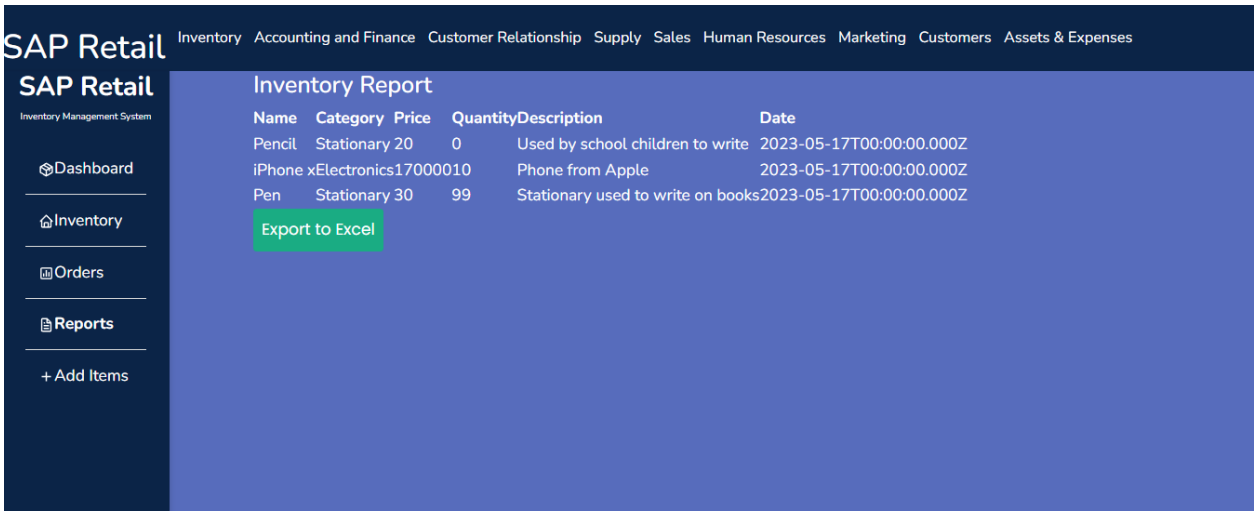
Inventory UI



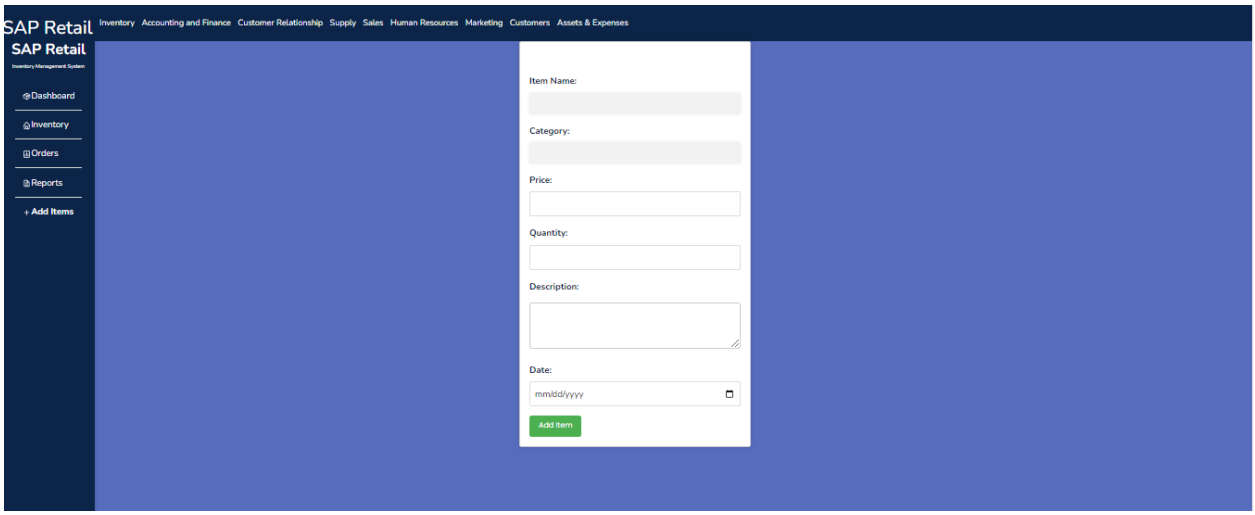
Inventory Order UI



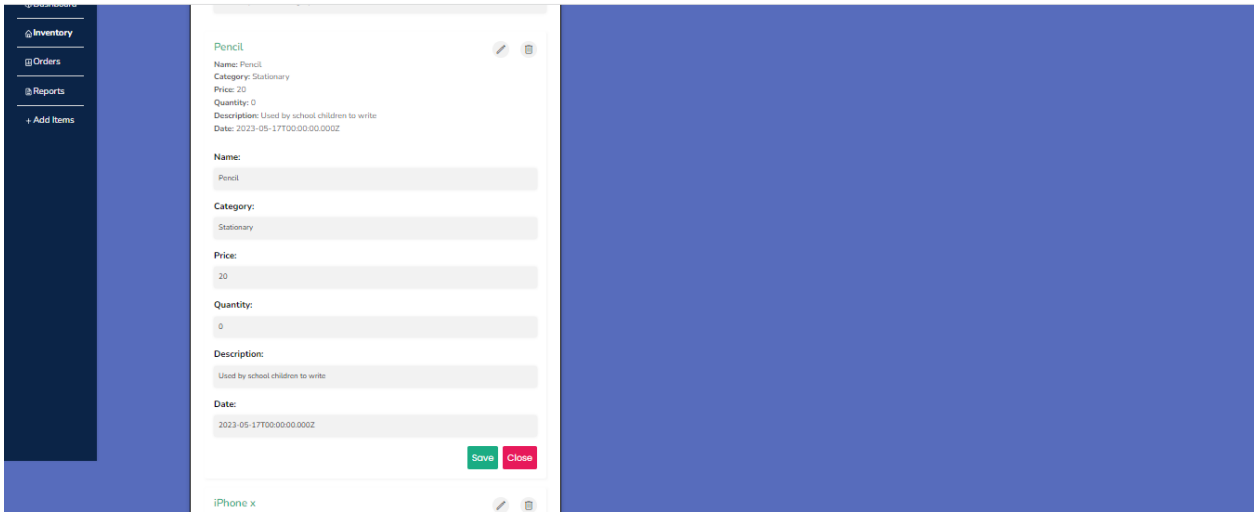
Inventory Report UI



Item Form



Edit Item Form



Item Form Validations

FrontEnd

```
const fieldsToCheck = ['name', 'category', 'price', 'quantity', 'description',
'date'];
    const emptyFields = [];
    fieldsToCheck.forEach((field) => {
      if (!eval(field)) {
        emptyFields.push(field);
      }
    });
    setEmptyFields(emptyFields);
```

Backend

```
try {
        if(!name || !category || !price || !quantity|| !description || !date){
            return res.status(400).json({message: 'All fields are required!'})
        }

        if(price <= 0 ){
            return res.status(400).json({message: 'Valid price required'})
        }
        //saving data into the database
        await item.save()
        res.status(200).json({message: 'Item was added'})
    } catch (error) {
        res.status(500).json({message: 'Server Error'})
    }
```

Testing

Accounting and Finance management system

Test ID	Test Input	Expected Output	Actual Output	Result (Pass/ Fail)	Description

1	Income:
	{
	Title: Damaged Goods
	Amount:5000
	Category: Refund
	Description: Goods were already Damaged on Arrival
	Type: Cash
	Date: 15/05/2023
	}
		{
		Title: Damaged Goods
		Amount:5000
		Category: Refund
		Description: Goods were already Damaged on Arrival
		Type: Cash
		Date: 15/05/2023
		}
			{
			Title: Damaged Goods
			Amount:5000
			Category: Refund
			Description: Goods were already Damaged on Arrival
			Type: Cash
			Date: 15/05/2023
			}
				Pass	Income is added as Intended.

2	Income:
	{
	Title:
	Amount:5000
	Category: Refund
	Description: Goods were already Damaged on Arrival
	Type: Cash
	Date: 16/05/2023
	}
		Alert : All fields Required	Alert : All fields Required	Pass	Title validation works as intended

3	{
	Title: Damaged Goods
	Amount:
	Category: Refund
	Description: Goods were already Damaged on Arrival
	Type: Cash
	Date: 16/05/2023
	}
		Alert : Invalid Income	Alert: Invalid Income	Pass	Blank Amount validation works as intended

4	{
	Title: Damaged Goods
	Amount: 0
	Category: Refund
	Description: Goods were already Damaged on Arrival
	Type: Cash
	Date: 16/05/2023
	}
		Alert : Invalid Income	Alert : Invalid Income	Pass	0 Amount validation works as intended

5	{
	Title: Damaged Goods
	Amount: -1
	Category: Refund
	Description: Goods were already Damaged on Arrival
	Type: Cash
	Date: 16/05/2023
	}
		Alert: Invalid Amount	Alert: Invalid Amount	Pass	Negative Amount validation works as intended

6	{
	Title: Damaged Goods
	Amount: 5000
	Category:
	Description: Goods were already Damaged on Arrival
	Type: Cash
	Date: 16/05/2023
	}
		Alert : All fields Required	Alert : All fields Required	Pass	Empty Category Validation works as intended

7	{
	Title: Damaged Goods
	Amount: -1
	Category: Refund
	Description:
	Type: Cash
	Date: 16/05/2023
	}
		Alert : All fields Required	Alert : All fields Required	Pass	Empty Description Validation works as intended

8	{
	Title: Damaged Goods
	Amount: -1
	Category: Refund
	Description: Goods were already Damaged on Arrival
	Type:
	Date: 16/05/2023
	}
		Alert : All fields Required	Alert : All fields Required	Pass	Empty Type validation works as intended

9	{
	Title: Damaged Goods
	Amount: -1
	Category: Refund
	Description: Goods were already Damaged on Arrival
	Type:
	Date:
	}
		Alert: All fields Required	Alert: All fields Required	Pass	Date Validation works as intended

10	Internal Method to Calculate total Income	522168	522168	Pass	Accurate Calculation

11	Internal Method to Calculate total Expense	281000	281000	Pass	Accurate Calculation
12	Internal Method to Calculate Total Purchase	604290	604290	Pass	Accurate Calculation
13	Internal Method to Calculate total Cash outflow	885290	885290	Pass	Accurate Calculation
14	Internal Method to calculate Gross Profit	-110222 + Closing stock	Null value	Fail	Invalid Closing stock value causes type error in gross profit calculation

Supply Chain management system

Test ID	Test Input	Expected Output	Actual Output	Result (Pass/Fail)	Description
1	Add supplier to database	Pop up for task complete	Pop up for task complete	Pass	Correctly added data to the database
2	Add supplier without filling all the data.	Show empty field message	Show empty field message	pass	Empty field validation works correctly
3	Add a 6-digit number as phone number	Shows incomplete phone number message	Shows incomplete phone number message	pass	Phone number validation works correctly
4	Try to Update supplier details	Shows update complete message and update data	Shows update complete message and update data	pass	Update function works properly
5	Try to delete a supplier	Shows delete complete message and delete data from database	Shows delete complete message and delete data from database	pass	Delete function works correctly
6	Press generates PDF button	Generate a PDF with all the supplier data	Generate a PDF with all the supplier data	pass	Generate pdf function works correctly
7	Add negative value for price in add new supply order	Show invalid price message	Show invalid price message	pass	Price validation is working correctly
8	Add negative value for Amount in add new supply order	Show invalid Amount message	Show invalid Amount message	Amount	Price validation is working correctly

Customer Relationship Management System.

Test ID	Test Input	Expected Output	Actual Output	Result (Pass/Fail)	Description

1	customerData:
	{
	name:"Shehan Perera",
	email:"shehan@gmail.com"
	address:"391/31,Malwatta Road, Gampaha.",
	contact:"0711758782",
	password:"12345$"
	}
		Alert: Customer Registration Successful	**Alert:** Customer Registration Successful	Pass	Validating customer registration data is successful

2	customerData:
	{
	name:"Shehan Perera",
	email:"shehan@gmail.com"
	address: "391/31,Malwatta Road, Gampaha.",
	contact: null,
	password: "12345$"
	}
		Alert: All Fields are Required	**Alert:** All Fields are Required	Pass	Validating blank input fields is successful

3	customerData:
	{
	name:"Shehan Perera",
	email:"shehan@gmail.com"
	address:"391/31,Malwatta Road, Gampaha.",
	contact:"0711758782796",
	password:"12345$"
	}
		Alert: Invalid Contact Number. (Contact number should have only 10 digits)	**Alert:** Invalid Contact Number. (Contact number should have only 10 digits)	Pass	validating contact number is successful

4	customerData:
	{
	name:"Shehan Perera",
	email:"shehan@gmail.com"
	address:"391/31,Malwatta Road, Gampaha.",
	contact:"0711758782",
	password:"1234"
	}
		Alert: Invalid Password. (Password should be contain more than 5 characters with at least 1 special character)	**Alert:** Invalid Password. (Password should be contain more than 5 characters with at least 1 special character)	Pass	Validating Password is successful

5	customerData:
	{
	name:”Shehan 123”,
	email:”shehan@gmail.com”
	address:”391/31,Malwatta Road, Gampaha.”,
	contact:”0711758782”,
	password:”12345$”
	}
		Alert:
		Invalid Name.
		(Name should have only characters, Name should contain more than 5 characters.)
			Alert:
			Invalid Name.
			(Name should have only characters, Name should contain more than 5 characters.)
				Pass	Validating Name is successful

6	customerData:
	{
	name:”Shehan Perera”,
	email:”shehan#gmail.com”
	address:”391/31,Malwatta Road, Gampaha.”,
	contact:”0711758782”,
	password:”12345$”
	}
		Alert:
		Invalid Email Address.
		(Invalid format of email address.)
			Alert:
			Invalid Email Address.
			(Invalid format of email address.)
				Pass	Validating Email Address is successful

7	customerData:
	{
	name:”Shehan Perera”,
	email:”shehan@gmail.com”
	address:”391@31,Malwatta Road, Gampaha.”,
	contact:”0711758782”,
	password:”12345$”
	}
		Alert:
		Invalid Address.
		(Remove special characters on address)
			Alert:
			Invalid Address.
			(Remove special characters on address)
				Pass	Validating Address is successful

Assets and expenses tracking system

Test ID	Test Input	Expected Output	Actual Output	Result (Pass/ Fail)	Description

1	Asset:
	{
	Item Code: A001
	Item Name : Computer
	Asset Amount:55000
	Date: 27/05/2023
	Residual Value:2500
	Useful Life Years: 3
	}
		{
		Item Code: A001
		Item Name : Computer
		Asset Amount:55000
		Date: 27/05/2023
		Residual Value:2500
		Useful Life Years: 3
		}
			{
			Item Code: A001
			Item Name : Computer
			Asset Amount:55000
			Date: 27/05/2023
			Residual Value:2500 Useful Life Years: 3
			}
				Pass	Asset is added

2	Asset:
	{
	Item Code: A001
	Item Name : Computer
	Asset Amount:
	Date: 27/05/2023
	Residual Value:2500
	Useful Life Years: 3
	}
		Alert : Invalid Amount	Alert: Invalid Amount	Pass	Blank Amount validation work

3	Asset:
	{
	Item Code: A001
	Item Name : Computer
	Asset Amount:0
	Date: 27/05/2023
	Residual Value:2500
	Useful Life Years: 3
	}
		Alert: Invalid Amount	Alert: Invalid Amount	Pass	0 Amount validation work

4	Asset:
	{
	Item Code: A001
	Item Name : Computer
	Asset Amount:-55000
	Date: 27/05/2023
	Residual Value:2500
	Useful Life Years: 3
	}
		Alert: Invalid Amount	Alert: Invalid Amount	Pass	Negative Amount validation work

5	Asset:
	{
	Item Code: A001
	Item Name :
	Asset Amount:55000
	Date: 27/05/2023
	Residual Value:2500
	Useful Life Years: 3
	}
		Alert : All fields Required	Alert : All fields Required	Pass	Empty Item Name validation work

6	Asset:
	{
	Item Code: A001
	Item Name : Computer
	Asset Amount:55000
	Date:
	Residual Value:2500
	Useful Life Years: 3
	}
		Alert : All fields Required	Alert : All fields Required	Pass	Empty date Validation work

7	Asset:
	{
	Item Code: A001
	Item Name : Computer
	Asset Amount:55000
	Date: 27/05/2023
	Residual Value:
	Useful Life Years: 3
	}
		Alert : All fields Required	Alert : All fields Required	Pass	Empty Residual Value Validation work

8	Liability:
	{
	Item Code: E002
	Item Name : BOC bank Loan
	Asset Amount:50000
	Date: 02/05/2023
	Ratio:7
	Keep Years: 3
	}
		{
		Item Code: E002
		Item Name : BOC bank Loan
		Asset Amount:50000
		Date: 02/05/2023
		Ratio:7
		Keep Years: 3
		}
			{
			Item Code: E002
			Item Name : BOC bank Loan
			Asset Amount:50000
			Date: 02/05/2023
			Ratio:7
			Keep Years: 3
			}
				Pass	Liability (expenses) is added

9	Liability:
	{
	Item Code: E002
	Item Name : BOC bank Loan
	Asset Amount:
	Date: 02/05/2023
	Ratio:7
	Keep Years: 3
	}
		Alert : Invalid Amount	Alert: Invalid Amount	Pass	Blank Amount validation work

10	Liability:
	{
	Item Code: E002
	Item Name : BOC bank Loan
	Asset Amount: 0
	Date: 02/05/2023
	Ratio:7
	Keep Years: 3
	}
		Alert: Invalid Amount	Alert: Invalid Amount	Pass	0 Amount validation work

11	Liability:
	{
	Item Code: E002
	Item Name : BOC bank Loan
	Asset Amount:-50000
	Date: 02/05/2023
	Ratio:7
	Keep Years: 3
	}
		Alert: Invalid Amount	Alert: Invalid Amount	Pass	Negative Amount validation work

12	Liability:
	{
	Item Code: E002
	Item Name :
	Asset Amount:50000
	Date: 02/05/2023
	Ratio:7
	Keep Years: 3
	}
		Alert : All fields Required	Alert : All fields Required	Pass	Empty Item Name validation work

13	Liability:
	{
	Item Code: E002
	Item Name : BOC bank Loan
	Asset Amount:50000
	Date:
	Ratio:7
	Keep Years: 3
	}
		Alert : All fields Required	Alert : All fields Required	Pass	Empty date Validation work

14	Liability:
	{
	Item Code: E002
	Item Name : BOC bank Loan
	Asset Amount:50000
	Date: 02/05/2023
	Ratio:
	Keep Years: 3
	}
		Alert : All fields Required	Alert : All fields Required	Pass	Empty Ratio Validation work

15	Calculate total Assets (Internal Method)	213500.5	213500.5	Pass	Accurate Calculation
16	Calculate total Liability (Expense) (Internal Method)	159000.5	159000.5	Pass	Accurate Calculation
17	Calculate Total Annual Depreciation (Internal Method)	58450.17	58450.17	Pass	Accurate Calculation
18	Calculate total Annual Interest (Internal Method)	29900.1	29900.1	Pass	Accurate Calculation

Marketing and sales management system

Test ID	Test Input	Expected Output	Actual Output	Result (Pass/Fail)	Description
1	Add Campaign to database	Show the pop-up box as added successfully.	Show the pop-up box as added successfully.	Pass	Correctly added data to the database
2	Add Campaign without filling all the data.	Show empty field message	Show empty field message	pass	Empty field validation works correctly
3	Add negative values to the price fields.	Show the negative input validations.	Show the negative input validations.	pass	Campaign add form shows the negative input validations.
4	Try to Update Campaign details	Shows the pop-up box as Edit successfully and update data	Shows the pop-up box as Edit successfully and update data	pass	Update function works properly
5	Try to delete a Campaign	Shows the pop-up box as Delete successfully and delete data from database	Shows the pop-up box as Delete successfully and delete data from database	pass	Delete function works correctly
6	Press generates PDF button	Generate a PDF with all the Campaign data	Generate a PDF with all the Campaign data	pass	Generate pdf function works correctly

Point of Sales system

Test ID	Test Input	Expected Output	Actual Output	Result(Pass/Fail)	Description

1	Bills:
	{
	“customername”:”tharain,
	“customernumber:”0759687452”,
	“totalamouts”:3000,
	“reason”:”cash”,
	“cartitems”:[]
	}
		{
		“customername”:
		”tharain,
		“customernumber:”
		0759687452”,
		“totalamouts”:3000,
		“reason”:”cash”,
		“cartitems”:[]
		}
			{
			“customername”:
			”tharain,
			“customernumber
			:”0759687452”,
			“totalamouts”:300
			0,
			“reason”:”cash”,
			“cartitems”:[]
			}
				Pass	Bill is added

2	Bills:	Alert : All fields	Alert : All fields	Pass	cartitems

	{ “customername”:”tharain, “customernumber:”07596 87452”, “totalamouts”:3000, “reason”:”cash”, }	Required	Required		validation works
3	Bills: { “customername”:”tharain, “customernumber:”07596 87452”, “reason”:”cash”, “cartitems”:[] }	Alert : Invalid Amount	Alert: Invalid Income	Pass	Amount validation works
4	Bills: { “customername”:”tharain, “customernumber:”07596 87452”, “totalamouts”:3000, “reason”:”cash”, “cartitems”: }	Alert : All fields Required	Alert : All fields Required	Pass	Empty Type validation works as intended
5	Bills: { “customername”:”tharain, “totalamouts”:3000, “reason”:”cash”, “cartitems”:[] }	Alert: All fields Required	Alert: All fields Required	Pass	Mobile number Validation works as intended
6	Internal Method to Calculate total bill amount	5700	5700	Pass	Accurate Calculation
7	Internal Method to Calculate total quantity	9	9	Pass	Accurate Calculation
8	Internal Method to Calculate total bill amount	1300	1300	Pass	Accurate Calculation
9	Internal Method to Calculate total quantity	3	3	Pass	Accurate Calculation
10	Items: { “name”:”maliban biscut”, “category”:snaks, “price”:”100”, “quantiy”:”5”, “image”:” j/4A” }	Item added Successfully	Item added Successfully	pass	Item added Successfully

Human resource management system

Test ID	Test Inputs	Expected Outputs	Actual Output	Result (Yes/No)	Description
01	Add to the employee details.	Show the positive details.	Show the positive details.	Yes	HR form show the positive attributes
02	Add wrong to the employee details.	Show the wrong details.	Show the wrong details.(Emp_id, Emp_name)	Yes	HR form show the negative attributes
03	Try to change the date	Show the date	Show the date	Yes	HR form show the changed date
04	Try to change the future date	Show the date	Validate today date Show the date(Avoid put in)	Yes	HR form rejects show the changed date
05	Delete the entire one employee details	Delete the details from DB	Delete the details from DB	Yes	HR form show deleted details and show other details(expect the delete one)
06	Edit the entire one employee details	Edit the details from DB	Edit the details from DB	Yes	HR form show Edited details and show other details(expect the edit one)
07	Add employee age as 15	Reject the age because you should at least 18 old.	Reject the age because you should at least 18 old.	Yes	HR form dosent require the age below 18 old.
08	Add empid and empname wrongly	Reject them because System have own validation.	Reject the them because System have own validation.	Yes	HR form dosent require the validation because they have own validation.

Inventory Management system

Test ID	Test Input	Expected Output	Actual Output	Result(Pass /Fail)	Description

1	Item:
	{
	Name: pen
	Category: Stationary
	Price: 30
	Quantity: 100
	Description: Stationary used to write
	Date: 16/05/2023
	}
		{
		Name: pen
		Category: Stationary
		Price: 30
		Quantity: 100
		Description: Stationary used to write
		Date: 16/05/2023
		}
			{
			Name: pen
			Category: Stationary
			Price: 30
			Quantity: 100
			Description: Stationary used to write
			Date: 16/05/2023
			}
				Pass	Item is added as Intended.

2	Item:
	{
	Name:
	Category: Electronics
	Price: 70000
	Quantity: 20
	Description: Phone from Samsung
	Date: 16/05/2023
	}
		Alert : All fields Required	Alert : All fields Required	Pass	Empty field validation works as intended

3	Item:
	{
	Name: iPhone X
	Category: Electronics
	Price: 70000
	Quantity: -1
	Description: Phone from Apple
	Date: 16/05/2023
	}
		Alert : Quantity should be >=0	Alert: Quantity should be >=0	Pass	Quantity validation works as intended

4	Item:
	{
	Name: iPhone X
	Category: Electronics
	Price: 70000e
	Quantity: 10
	Description: Phone from Apple
	Date: 16/05/2023
	}
		Alert : Price should be a number	Alert : Price should be a number	Pass	Price validation works as intended

10	Internal Method to Calculate total item value.	432000	432000	Pass	Accurate Calculation
11	Internal Method to Calculate total no of products out of stock.	Nan	Nan	Pass	Accurate Calculation as there is not records available
12	Internal Method to Calculate total no of categories.	15	15	Pass	Accurate Calculation

. Evaluation and Conclusion

The S.A.P Retail Management System was developed with the primary objective of managing and controlling all the procedures within the company. Based on discussions with the client, eight main functions were identified and implemented using the MERN stack. These functions were designed to be user-friendly, efficient, accurate, and secure. The system is planned to be hosted on HEROKU, a cloud platform that provides scalability and reliability.

Evaluation:

The evaluation of the system can be based on several factors:

Functionality: The system successfully implements the eight main functions identified during the discussions with the client. It effectively handles tasks such as sales data collection, inventory management, restocking, financial management, marketing and sales management, customer relationship management, asset and expense tracking, and human resource management.

User Experience: The system is user-friendly, ensuring that staff members can easily access different tasks and perform their duties efficiently. The intuitive interface and streamlined processes contribute to a positive user experience.

Accuracy and Security: The system ensures the accuracy of data by effectively collecting, updating, and managing information related to sales, inventory, financials, and other aspects of the business. The implementation of security measures ensures that sensitive data is protected from unauthorized access or breaches.

Hosting and Scalability: The decision to host the system on HEROKU indicates a consideration for scalability and reliability. HEROKU's cloud infrastructure allows for easy deployment, management, and scaling of the application as the business grows.

Conclusion:

In conclusion, the implementation of the S.A.P Retail Management System addresses the client's needs by providing a comprehensive solution for managing and controlling various procedures within the company. The system offers user-friendly interfaces, efficient processes, accurate data management, and security measures. By choosing HEROKU for hosting, the system is positioned for scalability and reliability. Overall, the customer's satisfaction with the final product is a positive indicator of the system's success in meeting the objectives and requirements set forth during the development process.

References

[1] J. W. Toomey, Inventory management: principles, concepts and techniques. Boston: Kluwer Academic, 2003.

[2] N. Slack, S. Chambers, and R. Johnston, Operations management. Harlow: Prentice Hall/Financial Times, 2009. 29

[3] T. Arnold and S. N. Chapman, Introduction to materials management. London: Prentice Hall, Cop, 2001.

[4] S. Chopra and P. Meindl, Supply chain management: strategy, planning, and operation, 6th ed. Boston: Pearson, 2016.

[5]P. Kotler and K. L. Keller, Marketing management, 15th ed. Boston: Pearson, 2016.

[6]J. C. Reilly, An Overview of Point of Sale Systems. Createspace Independent Publishing Platform, 2016.

[7]G. Dessler, Human resource management. Harlow, England Pearson, 2020. https://eddy.com/hr-encyclopedia/hr-in-the-retail-industry/

[8] R. H. Peterson and I. Netlibrary, Accounting for fixed assets. New York: J. Wiley, 2002.

[9] F. Buttle, Customer relationship management : concepts and technologies. Oxford: Elsevier/Butterworth-Heinemann, 2009.

[10] F. R. Jacobs and R. B. Chase, Operations and supply chain management : The core. New York, Ny: Mcgraw-Hill Education, 2020.

[11] Draw.io, "Flowchart Maker & Online Diagram Software," app.diagrams.net, 2022. https://app.diagrams.net/

[12] Coursehero.com, 2023. https://www.coursehero.com/study-guides/wmopenretailmanagement/human-resource- (accessed Mar. 09, 2023).

[13] "React Tutorial," www.w3schools.com. https://www.w3schools.com/react (accessed Mar. 09, 2023).
[14] [Online]. Available: https://www.tutorialspoint.com/reactjs/reactjs_overview.htm.

[15] "Node.js Tutorial," W3schools.com, 2019. https://www.w3schools.com/nodejs/ [

16] D. Taylor, "What is MongoDB? Introduction, Architecture, Features & Example," www.guru99.com, Sep. 10, 2022. https://www.guru99.com/what-is-mongodb.html

[17] [Online]. Available: https://www.geeksforgeeks.org/introduction-postman-apidevelopment/ 30

[18] "ReactJS - Overview - Tutorialspoint," www.tutorialspoint.com. https://www.tutorialspoint.com/reactjs/reactjs_over view.htm