

**Final Year Project Report
Integrated Business Model**



Project Advisor: Dr. Asif Subhani

Submitted By:

Abdul Raheem (F2018266275)

Haris Siddiqui (F2018266277)

Zeeshan Fiaz (F2018266286)

Abu Bakar (S2020266073)

Spring 2022

**University of Management and Technology
C-II Johar Town Lahore Pakistan**

Dedication

Final Approval

Panel of Examiners

- **Head of Department**
Department of Software Engineering
University of Management and Technology
Lahore _____
- **Program Director (Final Year Projects)**
Department of Software Engineering
University of Management and Technology
Lahore _____
- **Supervisor**
Department of Software Engineering
University of Management and Technology
Lahore _____
- **Co-Supervisor**
(If any) _____

Acknowledgment

Project Title: Integrated Business Model

Objective: The goal of our project is to develop an efficient system that would be able to manage the business in more reliable and smart way. The users won't need to depend on other departments to stay updated about any transaction.

Undertaken by: Afifa Wajid

Supervised by: Dr. Asif Subhani

Starting Date: Dec 12, 2021

Completion Date: Dec 31, 2021

Tools Used: Visual Studio Code, ReactJs, MongoDB, ExpressJs, NodeJs, Postman

Operating System: Kali Linux, Windows 10, Android 10.1

Documentation: Abdul Raheem, Muhammed Zeeshan Fiaz, Abu Bakar Rehan, Haris Siddiqui

Plagairism Report

Declaration Form

I have carefully examined the documentation of the Final Year Project titled “*Project title*”; and I endorse that this documentation complies with the standards of an undergraduate level Final Year Project report.

The document has been checked for plagiarism through Turnitin software available in UMT Library. The similarities of the document are within acceptable range.

Moreover, the accompanying CDs contain PDF of the documentation, as well as the source code and binaries with user manual and installation guide.

FYP Advisor Name: Dr. Asif Subhani

Signature: _____

Date: _____

Abstract

Currently, the main problem with traditional business models is no direct access to data. Some other problems are:

- Inefficiency
- Data Reporting
- Maintaining Records Manually

The goal of our project is to develop an efficient system that would be able to manage the business in more reliable and smart way. The users won't need to depend on other departments to stay updated about any transaction. We are actually developing:

1. Customer Relation Management
2. Book Keeping
3. Human Resource Management
4. Inventory Management
5. Point of Sale
6. Supply Chain Management

And then integrate these systems to form a single **Integrated Business Model**.

REVISION CHART

Version	Primary Author(s)	Description of Version	Date Completed
Draft	Abdul Raheem	Initial draft created for distribution and review comments	Dec 14, 2021
Preliminary	Abdul Raheem	Second draft incorporating initial review comments, distributed for final review	Feb 19, 2022
Final	Abdul Raheem	First complete draft, which is placed under change control	Sep 12, 2022
Revision	Abdul Raheem	Revised draft, revised according to the change control process and maintained under change control	Jan 1, 2023

CONTENTS

CONTENTS	1
DEFINITIONS AND ACRONYMS.....	3
LIST OF FIGURES	4
LIST OF TABLES.....	7
1. INTRODUCTION.....	9
1.1 MOTIVATIONS	9
1.2 PROJECT OVERVIEW.....	9
1.3 PROBLEM STATEMENT	9
1.4 OBJECTIVES.....	9
2. DOMAIN ANALYSIS	11
2.1 CUSTOMER	11
2.2 STAKEHOLDERS.....	11
2.3 AFFECTED GROUPS WITH SOCIAL OR ECONOMIC IMPACT	11
2.4 DEPENDENCIES/ EXTERNAL SYSTEMS.....	11
2.5 REFERENCE DOCUMENTS	12
2.5.1 <i>Related Projects</i>	12
2.5.2 <i>Feature Comparison</i>	12
3. REQUIREMENTS ANALYSIS.....	14
3.1 REQUIREMENTS.....	14
3.2 LIST OF ACTORS	17
3.3 LIST OF USE CASES	17
3.4 SYSTEM USE CASE DIAGRAM	18
3.5 EXTENDED USE CASES	19
3.6 USER INTERFACES (MOCK SCREENS).....	64
4. DATA FLOW DIAGRAM (OPTIONAL)	74
4.1 DATA FLOW DIAGRAM LEVEL 0	74
4.2 DATA FLOW DIAGRAM LEVEL 1	75
5. SYSTEM DESIGN.....	76
5.1 SYSTEM ARCHITECTURE DIAGRAM.....	76
5.2 CLASS DIAGRAM	77
5.3 SEQUENCE DIAGRAMS.....	78
5.4 COLLABORATION DIAGRAMS	88
5.5 ERD	98
5.6 DATA DICTIONARY	99
6. IMPLEMENTATION DETAILS.....	101
6.1 DEVELOPMENT SETUP.....	101
6.2 DEPLOYMENT SETUP	101
6.3 ALGORITHMS.....	101
6.3.1 <i>Add Account</i>	101

6.3.2	<i>Generate Salaries</i>	104
6.3.3	<i>Add Raw Material</i>	105
6.3.4	<i>Add Purchase</i>	106
6.3.5	<i>Add Employee</i>	107
6.3.6	<i>Capital Management</i>	111
6.3.7	<i>Sales Management</i>	113
6.4	CONSTRAINTS.....	117
6.4.1	<i>Assumptions</i>	117
6.4.2	<i>System constraints</i>	117
6.4.3	<i>Restrictions</i>	117
6.4.4	<i>Limitations</i>	117
7.	TESTING	118
7.1	EXTENDED TEST CASES	118
7.2	DECISION TABLE	153
7.2.1	<i>Code snippet</i>	153
7.2.2	<i>Decision coverage table</i>	153
7.3	TRACEABILITY MATRIX	153
7.3.1	<i>RID vs UCID (requirements vs use cases)</i>	153
7.3.2	<i>Prototypes (RID vs PID)</i>	156
7.3.3	<i>Test Cases (RID vs TID)</i>	155
7.3.4	<i>Coverage (UCID vs TID)</i>	157
8.	RESULTS/OUTPUT/STATISTICS	158
8.1	%COMPLETION	158
8.2	%ACCURACY	158
8.3	%CORRECTNESS	158
9.	CONCLUSION	159
10.	FUTURE WORK	160
11.	BIBLIOGRAPHY	161
11.1	BOOKS	161
11.2	JOURNALS	161
11.3	ARTICLES	161
11.4	RESEARCH PAPERS.....	161
11.5	OTHER REFERENCES.....	161
12.	APPENDIX	162
12.1	GLOSSARY OF TERMS	162
12.2	PRE-REQUISITES	162

DEFINITIONS AND ACRONYMS

Acronym	Definition
UMT	University of Management and Technology
POS	Point of Sale

Table 1: Table of acronyms and definitions

LIST OF FIGURES

Figure 1(3.4): Use Case Diagram.....	18
Figure 2(3.6): Manage User	64
Figure 3(3.6): Manage Account	64
Figure 4(3.6): Account Details.....	65
Figure 5(3.6): Manage Capital	65
Figure 6(3.6): Generate Report	66
Figure 7(3.6): Manage Payments	66
Figure 8(3.6): Manage Employee Request.....	67
Figure 9(3.6): Employee Request	67
Figure 10(3.6): Manage Purchase	68
Figure 11(3.6): Manage Department.....	68
Figure 12(3.6): Manage Attendance.....	69
Figure 13(3.6): Manage Backup	69
Figure 14(3.6): Manage Employee	71
Figure 15(3.6): Employee Details	71
Figure 16(3.6): Manage Production	72
Figure 17(3.6): View Production	72
Figure 18(3.6): Dashboard	73
Figure 19(4.1): DFD Level 0 Diagram	74
Figure 20(4.2): DFD Level 1 Diagram	75
Figure 21(5.1): System Architecture.....	76
Figure 22(5.2): Class Diagram.....	77
Figure 23(5.3): Authenticate User.....	78
Figure 24(5.3): Manage User	78
Figure 25(5.3): Manage Accounts.....	79
Figure 26(5.3): Manage Capital	79
Figure 27(5.3): Generate Report	80
Figure 28(5.3): Manage Expense	80
Figure 29(5.3): Manage Payments	81
Figure 30(5.3): Manage Employee Requests	81
Figure 31(5.3): Manage Purchases.....	82
Figure 32(5.3): Manage Production Requests.....	82
Figure 33(5.3): Manage Sales	83

Figure 34(5.3): Manage Products.....	83
Figure 35(5.3): Manage Raw Material.....	84
Figure 36(5.3): Manage Departments	84
Figure 37(5.3): Manage Backup	85
Figure 38(5.3): Manage Attendance.....	85
Figure 39(5.3): Manage Employees.....	86
Figure 40(5.3): Manage Salaries	86
Figure 41(5.3): Manage Production	87
Figure 42(5.4): Authenticate User.....	88
Figure 43(5.4): Manage User	88
Figure 44(5.4): Manage Accounts.....	89
Figure 45(5.4): Manage Capital	89
Figure 46(5.4): Generate Report	90
Figure 47(5.4): Manage Expense	90
Figure 48(5.4): Manage Payments	91
Figure 49(5.4): Manage Employee Request.....	91
Figure 50(5.4): Manage Purchases.....	92
Figure 51(5.4): Manage Production Request	92
Figure 52(5.4): Manage Sales	93
Figure 53(5.4): Manage Products.....	93
Figure 54(5.4): Manage Raw Material.....	94
Figure 55(5.4): Manage Department.....	94
Figure 56(5.4): Manage Backup	95
Figure 57(5.4): Manage Attendance.....	95
Figure 58(5.4): Manage Employee	96
Figure 59(5.4): Manage Salaries	96
Figure 60(5.4): Manage Production	97
Figure 61(5.5): ERD.....	98

LIST OF TABLES

Table 1: table of acronyms and definitions	3
Table 2(2.5): Feature Comparison	12
Table 3(3.1): Requirement Categories	14
Table 4(3.1): List of Requirements	16
Table 5(3.5): UC-1 Authenticate.....	19
Table 6(3.5): UC-2 Manage Users	21
Table 7(3.5): UC-3 Manage Accounts	23
Table 8(3.5): UC-4 Manage Capital.....	28
Table 9(3.5): UC-5 Generate Reports	30
Table 10(3.5): UC-6 Manage Expense.....	31
Table 11(3.5): UC-7 Manage Payments.....	33
Table 12(3.5): UC-8 Manage Employee Requests.....	36
Table 13(3.5): UC-9 Manage Purchases	37
Table 14(3.5): UC-10 Manage Production Requests	42
Table 15(3.5): UC-11 Manage Sales.....	43
Table 16(3.5): UC-12 Manage Products	48
Table 17(3.5): UC-13 Manage Raw Material	49
Table 18(3.5): UC-14 Manage Departments.....	50
Table 19(3.5): UC-15 Manage Backup.....	51
Table 20(3.5): UC-16 Manage Attendance	52
Table 21(3.5): UC-17 Manage Employees	54
Table 22(3.5): UC-18 Manage Salaries.....	58
Table 23(3.5): UC-19 Manage Production.....	60
Table 24(5.6): Data Dictionary	99
Table 25(6.1): Development Setup	101
Table 26(7.1): Test Case 1 Login.....	118
Table 27(7.1): Test Case 2 Forget Password	120
Table 28(7.1): Test Case Manage Account.....	122
Table 29(7.1): Test Case 4 Manage Capital.....	126
Table 30(7.1): Test Case 6 Manage Expense.....	127
Table 31(7.1): Test Case 7 Manage Payments (Payments In)	129
Table 32(): Test Case 8 Manage Payments (Payments Out).....	131
Table 33(7.1): Test Case 9 Manage Purchases	133

Table 34(7.1): Test Case 10 Add sale	136
Table 35(7.1): Test Case 11 Return Sale.....	139
Table 36(7.1): Test Case 12 Manage Products	141
Table 37(7.1): Test Case 13 Manage Raw Material.....	142
Table 38(7.1): Test Case 14 Manage Departments	143
Table 39(7.1): Test Case 15 Search Attendance	144
Table 40(7.1): Test Case 16 Marked Attendance.....	145
Table 41(7.1): Test Case 17 Add/Update Employees	146
Table 42(7.1): Test Case 18 Add/Update Production	150
Table 43(7.2): Decision Coverage Table	146
Table 44(7.3): RID vs UCID	153
Table 45(7.3): RID vs PID	148
Table 46(7.3): RID vs TID	155
Table 47(7.3): UCID vs TID	157

1. INTRODUCTION

1.1 Motivations

In this era of technology, almost everything is going online no matter if it is a student attendance system or an ultimate business solution. Everyone wants access to their data every minute. They want to stay updated about everything. So, what if we have an Integrated Business Solution that is accessible anytime anywhere? What if user is able to manage their business on the go?

So, if we have access to our business data anytime anywhere, we may work more efficiently and stay updated to current business status. We won't have to wait for our finance department to confirm if we can afford the product or not. We can automate most of our business load and keep track of every bit. We may reduce extra expenses and improve our business model by reading the stats of current business.

1.2 Project Overview

Currently, the main problem with traditional business models is no direct access to data. Some other problems are:

- Inefficiency
- Data Reporting
- Maintaining Records Manually

The goal of our project is to develop an efficient system that would be able to manage the business in more reliable and smart way. The users won't need to depend on other departments to stay updated about any transaction. We are actually developing:

7. Customer Relation Management
8. Book Keeping
9. Human Resource Management
10. Inventory Management
11. Point of Sale
12. Supply Chain Management

And then integrate these systems to form a single **Integrated Business Model**.

1.3 Problem Statement

In our traditional business scenario, we don't have a direct access to every data that we need to know. We need different departments for managing different accounts. Risk for mistakes is very high because the workflow is totally manual and time consuming.

- We don't know how much stock is left in our inventory without confirming from our Inventory Manager.
- We don't know what our financial condition is without our finance department.
- If there is even a single wrong entry in any transaction, the balance sheet would not be balanced anymore and you would have to find the mistake manually that could take hours, days or maybe months. Human may make mistake but Computer won't.
- You have to write every transaction manually, whether it is a product purchase invoice, or maintaining a customer's account.

1.4 Objectives

The main objective of this project is developing a system that would:

- Enhance the operational efficiency of business resources.
- Assist company owners to achieve their goals at a faster rate.
- Assist in knowing real time information about business field strategies.
- Enhance customer relation management.
- Increase quality of services.
- Shorten delivery time.
- Control Errors in transactions.
- Helps in better planning and coordination of business resources so as to achieve maximum profit.
- Improve accuracy rate of results.

2. DOMAIN ANALYSIS

2.1 Customer

RANKOLI is a Vanity manufacturing company in Johar Town, Lahore. They manufacture their products and sell them to dealers on credit. They have a production department, finance department, accountants and sales department. They need a software that can automate most of their transaction entries and that can generate some of the accounting reports like Balance Sheet, Profit Loss Statement etc.

2.2 Stakeholders

Stakeholder	Role in System
HR	Use human resource management tool for accessing data for employs and salaries. HR may insert new employs. HR can also update expenses.
Sales	This department is concerned with making sales and forecasting sales. They may use CRM data to approach the right customer with right product. Update or Insert entries in book keeping.
Account Management	Recording information directly related to specific customers. E.g. Credit Amount, Debit Amount, Contact Info
Finance	This department often uses CRM for invoicing purpose. They may check data from book keeping management for any debts or debit amount. Pay for the purchases. Keep record of capital/equity.
Business Owner/President	He can view stats and current status of company
Production	They may use the inventory management system to check for available raw material and stocks. Keep record or work in progress.
Delivery Team	They may use the system to retrieve customer information like address and contact information
Customer	They may use the system to fetch their records. E.g. Remaining Balance.

Table 1(2.2): List of stakeholders

2.3 Affected Groups with social or economic impact

- Business Owner/President
- Production Department
- Labor
- Sales Team
- Customers
- Finance Department
- Human Resource Department

2.4 Dependencies/ External Systems

- Internet
- Computer or Mobile Phone

- Web hosting or Dedicated Server
- Online Database
- Printe

2.5 Reference Documents

2.5.1 Related Projects

In order to develop RANKOLI IBM, we looked up several similar systems. Their details are given below:

1. Odoo
Odoo is a suit of open source business apps that includes CRM, POS, Accounting, Supply Chain, Inventory System etc.
2. ERPNext
ERPNext is free and open source integrated ERP software developed by Frappe Technologies PVT.

2.5.2 Feature Comparison

SR No.	Feature	Odoo	ERPNext	Remarks
1	Database	Postgresql	MariaDb	Postgresql offers more complex data-types and allows objects to inherit properties.
2	Invoicing	Yes	Yes	Available for both
3	Online Payment	Yes	No	Odoo Support VISA, American Express, UnionPay, Discover and many more online payment methods.
4	Inventory Module	Available	Available	Odoo have additional feature of QR Code Support
5	QR Code Support	Yes	No	QR Code Support is beneficial in many inventory management

Table 2(2.5): Feature Comparison

3. REQUIREMENTS ANALYSIS

3.1 Requirements

The Requirements are based on the performance and functionalities the system must uphold in order to achieve objectives.

Table 4 contains the list of categories to which requirements belong. The requirements type given in table categorizes the requirements. So that it is easier for the student to identify the requirements by unique id given to each requirement.

RID	Requirement Type	Description
FR_1.x	Functional Requirement	Core requirements of business operations.
NFR_2.x	Non-Functional Requirement	Supporting requirements for functional requirements.
PR_3.x	Programing Requirement	Tools used to develop the web application.
DR_4.x	Data Requirement	Data storage and resources.

Table 3(3.1): Requirement Categories

RID	Description	Category	Attribute	Details & Boundary Constraints
FR_1.1	Authenticate the user to the system	Functional Requirement	Authenticate	
FR_1.2	Activate/Deactivate user and add role based access	Functional Requirement	User Management	
FR_1.3	Add, remove, update account and view account ledgers	Functional Requirement	Account Management	
FR_1.4	Add, remove and update capital	Functional Requirement	Capital Management	
FR_1.5	Generate various types of reports	Functional Requirement	Generate Report	
FR_1.6	Add, remove and update expense	Functional Requirement	Expense Management	

FR_1.7	Payment in/out	Functional Requirement	Payments Management	
FR_1.8	Accept/Decline employee creation request	Functional Requirement	Employee Request Management	
FR_1.9	Add, remove and update purchase/purchase returns	Functional Requirement	Purchases Management	
FR_1.10	Accept/Decline production add/update request	Functional Requirement	Production Request Management	
FR_1.11	Add, remove and update sale/sale return requests.	Functional Requirement	Sales Management	
FR_1.12	Add, remove and update products	Functional Requirement	Products Management	
FR_1.13	Add, remove and update raw material	Functional Requirement	Raw Material Management	
FR_1.14	Add, remove and update departments	Functional Requirement	Departments Management	
FR_1.15	Create/Restore Backup	Functional Requirement	Backup Management	
FR_1.16	Search attendance and keep record of check-in/check-out time	Functional Requirement	Attendance Management	
FR_1.17	Add, remove and update employee	Functional Requirement	Employee Management	
FR_1.18	Generate salary or pay salary to all/one	Functional Requirement	Salary Management	
FR_1.19	Start, update and finish production	Functional Requirement	Production Management	

NFR_2.1	Adjust component size according to screen size	Non Functional Requirement	Responsive Design	
NFR_2.2	Access system from anywhere over the internet	Non Functional Requirement	Online Access	
NFR_2.3	Passwords are hashed	Non Function Requirement	Credential Security	
NFR_2.4	Application don't need to be loaded multiple time on every submission	Non Function Requirement	Single Page Application	
NFR_2.5	User should be able to use application easily	Non Function Requirement	Easy Interface	
PR_3.1	MERN Stack	Programing Requirement	Technologies	
DR_4.1	Online MongoDB NoSQL database	Data Requirement	Storage Resource	

Table 4(3.1): List of Requirements

3.2 List of Actors

- **Administrator:** Manage system users and role based access.
- **Accounts Manager:** Manage accounts, payments and expense, capital and employee requests. Also responsible for generating financial reports.
- **HR Manager:** Manage attendance, employees and salaries.
- **Procurement Manager:** Manage purchases and production requests.
- **Production Manager:** Manage production of products.
- **Sales Manager:** Responsible for managing sales.
- **System Manager:** Manage products, raw material, departments and backup.

3.3 List of use cases

- **UC-1 Authenticate:** Authenticate the user to the system
- **UC-2 Manage Users:** Activate/Deactivate user and add role based access
- **UC-3 Manage Accounts:** Add, remove, update account and view account ledgers
- **UC-4 Manage Capital:** Add, remove and update capital
- **UC-5 Generate Reports:** Generate various types of reports
- **UC-6 Manage Expense:** Add, remove and update expense
- **UC-7 Manage Payments:** Payment in, Payment out
- **UC-8 Manage Employee Requests:** Accept/Decline employee creation request
- **UC-9 Manage Purchases:** Add, remove and update purchase/purchase returns
- **UC-10 Manage Production Requests:** Accept/Decline production add/update request
- **UC-11 Manage Sales:** Add, remove and update sale/sale return requests.
- **UC-12 Manage Products:** Add, remove and update products
- **UC-13 Manage Raw Material:** Add, remove and update raw material
- **UC-14 Manage Departments:** Add, remove and update departments
- **UC-15 Manage Backup:** Create/Restore Backup
- **UC-16 Manage Attendance:** Search attendance and keep record of check-in/check-out time
- **UC-17 Manage Employees:** Add, remove and update employee
- **UC-18 Manage Salaries:** Generate salary or pay salary to all/one
- **UC-19 Manage Production:** Start, update and finish production

3.4 System use case diagram

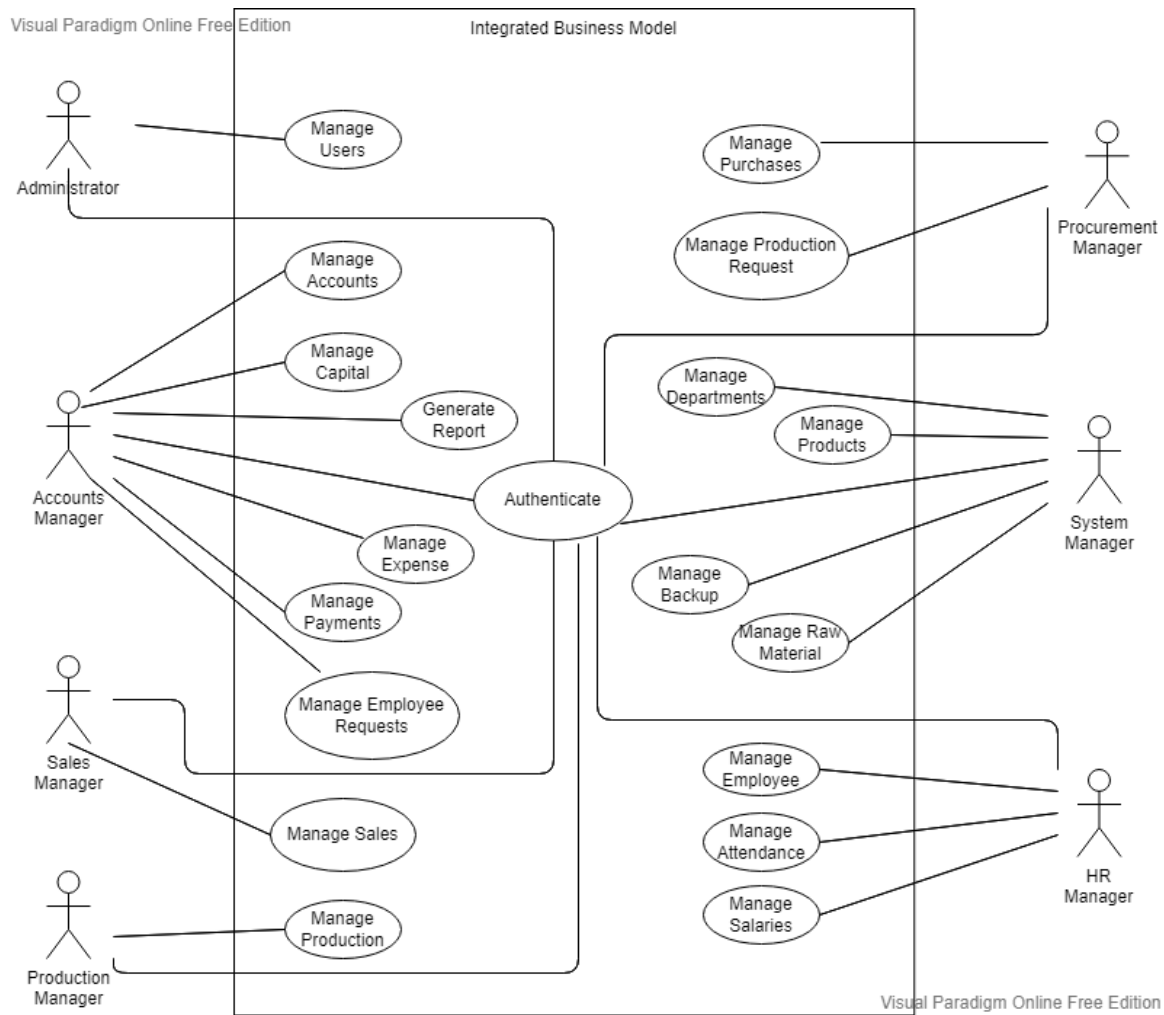


Figure 1(3.4): Use Case Diagram

3.5 Extended use cases

Table 5(3.5): UC-1 Authenticate

Use Case ID:	UC-1		
Use Case Name:	Authenticate		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	Accounts Manager, Administrator, HR Manager, Procurement Manager, Production Manager, Sales Manager, System Manager		
Description:	Actor would be able to login to the system or reset login password.		
Trigger:	Actor starts the application.		
Preconditions:	N.A		
Post conditions:	Login <ol style="list-style-type: none"> 1. System displays dashboard page. 2. System displays message “Login Successful”. Forget Password <ol style="list-style-type: none"> 1. System send password reset link to actor’s email address. 2. System displays message “Password reset link sent to your email address”. 		
Normal Flow:	Login <ol style="list-style-type: none"> 1. Actor enters employee id. 2. Actor enters password. 3. Actor sends request to sign into the system. 4. System displays dashboard page. 5. System displays message “Login Successful”. Forget Password <ol style="list-style-type: none"> 1. Actor selects forget password option. 2. System displays reset account password page. 3. Actor enters employee id. 4. Actor sends request to reset password. 5. System sends password link to actor’s email address. 6. System displays message “Password reset link sent to your email address”. 		
Alternative Flows:	N.A		
Exceptions:	Login <ol style="list-style-type: none"> 1a. Employee id field is empty, <ol style="list-style-type: none"> 1. System displays message “Employee id is required”. 2. Actor enters employee id. 3. Use case resumes on step 2. 1b. Employee id field contains alphabet, <ol style="list-style-type: none"> 1. System displays message “Invalid employee id”. 2. Actor enters valid employee id. 3. Use case resumes on step 2. 1c. Employee id length is not equal to 13, <ol style="list-style-type: none"> 1. System displays message “Invalid employee id”. 2. Actor enters valid employee id. 		

	<p>3. Use case resumes on step 2.</p> <p>2a. Password field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Password field is required”. 2. Actor enters password. 3. Use case resumes on step 3. <p>2b. Password length is smaller than 8 characters,</p> <ol style="list-style-type: none"> 1. System displays message “Password length can’t be less than 8 characters”. 2. Actor enters valid password. 3. Use case resumes on step 3. <p>Forget Password</p> <p>3a. Employee id field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Employee id is required”. 2. Actor enters employee id. 3. Use case resumes on step 4. <p>3b. Employee id field contains alphabet,</p> <ol style="list-style-type: none"> 1. System displays message “Invalid employee id”. 2. Actor enters valid employee id. 3. Use case resumes on step 4. <p>3c. Employee id length is not equal to 13,</p> <ol style="list-style-type: none"> 1. System displays message “Invalid employee id”. 2. Actor enters valid employee id. 3. Use case resumes on step 4.
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	N.A
Assumptions:	N.A
Notes and Issues:	N.A

Table 6(3.5): UC-2 Manage Users

Use Case ID:	UC-2		
Use Case Name:	Manage Users		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	Administrator		
Description:	Actor would be able to activate/deactivate user account and grant role based access.		
Trigger:	Actor navigates to user management		
Preconditions:	N.A		
Post conditions:	<p>Activate User</p> <ol style="list-style-type: none"> 1. System updates employee account's active status. 2. System sends login credentials to employee's email address. 3. System displays message "User activated". <p>Deactivate User</p> <ol style="list-style-type: none"> 1. System updates employee account's active status. 2. System displays message "User deactivated". <p>Grant Access</p> <ol style="list-style-type: none"> 1. System updates employee's access rights. 2. System display message "Access Granted". 		
Normal Flow:	<p>Activate User</p> <ol style="list-style-type: none"> 1. Actor navigates to user management. 2. System displays list of all employees. 3. Actor sends request to activate employee's account. 4. System updates employee account's active status. 5. System sends login credentials to employee's email address. 6. System displays message "User activated". <p>Deactivate User</p> <ol style="list-style-type: none"> 1. Actor navigates to user management. 2. System displays list of all employees. 3. Actor sends request to deactivate employee's account. 4. System updates employee account's active status. 5. System displays message "User deactivated". <p>Grant Access</p> <ol style="list-style-type: none"> 1. Actor navigates to user management. 2. System displays list of all employees. 3. Actor selects manage access for an employee. 4. System displays list of modules. 5. Actor selects required access roles. 6. Actor sends request to grant request. 7. System updates employee's access rights. 8. System display message "Access Granted". 		
Alternative Flows:	N.A		
Exceptions:	<p>Grant Access</p> <ol style="list-style-type: none"> 5. Actor didn't selected any access role, 		

	<ol style="list-style-type: none"> 1. System displays message “At least 1 access role is required”. 2. Actor selects required access roles. 3. Use case resumes on step 6.
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	N.A
Assumptions:	N.A
Notes and Issues:	N.A

Table 7(3.5): UC-3 Manage Accounts

Use Case ID:	UC-3		
Use Case Name:	Manage Account		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	Accounts Manager		
Description:	Actor would be able to add, remove and update accounts and view account ledgers.		
Trigger:	Actor navigates to account management		
Preconditions:	N.A		
Post conditions:	Add Account <ol style="list-style-type: none"> 1. System adds new account record. 2. System displays message “Account Added”. Update Account <ol style="list-style-type: none"> 1. System adds new account record. 2. System displays message “Account Updated”. Remove Account <ol style="list-style-type: none"> 1. System deletes account from records. 2. System display message “Account Deleted”. View Account Ledger <ol style="list-style-type: none"> 1. System display account ledgers. 		
Normal Flow:	Add Account <ol style="list-style-type: none"> 1. Actor navigates to account management. 2. System displays list of all accounts 3. Actor selects add account option. 4. System displays add account form. 5. Actor enters first name. 6. Actor enters last name. 7. Actor enters cnic number. 8. Actor enters mobile number. 9. Actor enters address. 10. Actor enters email address. 11. Actor enters emergency contact number. 12. Actor selects account type. 13. Actor enters opening balance. 14. Actor selects opening balance type. 15. Actor sends request to add account. 16. System adds new account record. 17. System displays message “Account Added”. Update Account <ol style="list-style-type: none"> 1. Actor navigates to account management. 2. System displays list of all accounts 3. Actor selects edit account option from account list. 4. System displays edit account form. 5. Actor enters first name. 6. Actor enters last name. 7. Actor enters cnic number. 8. Actor enters mobile number. 9. Actor enters address. 10. Actor enters email address. 11. Actor enters emergency contact number. 12. Actor selects account type. 		

	<ol style="list-style-type: none"> 13. Actor enters opening balance. 14. Actor selects opening balance type. 15. Actor sends request to update account. 16. System updates account record. 17. System displays message “Account Updated”. <p>Remove Account</p> <ol style="list-style-type: none"> 1. Actor navigates to account management. 2. System displays list of all accounts 3. Actor sends request to delete account from account list. 4. System deletes account from records. 5. System display message “Account Deleted”. <p>View Account Ledger</p> <ol style="list-style-type: none"> 1. Actor navigates to account management. 2. System displays list of all accounts 3. Actor selects account from the list. 4. System display account information. 5. Actor selects view ledger option. 6. System display account ledgers.
Alternative Flows:	N.A
Exceptions:	<p>Add Account</p> <p>5a. First name field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “First name is required”. 2. Actor enters first name. 3. Use case resumes on step 6. <p>5b. First name field contains number or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Invalid first name”. 2. Actor enters first name. 3. Use case resumes on step 6. <p>6a. Last name field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Last name is required”. 2. Actor enters last name. 3. Use case resumes on step 7. <p>6b. Last name field contains number or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Last name is required”. 2. Actor enters last name. 3. Use case resumes on step 7. <p>7a. CNIC field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “CNIC is required”. 2. Actor enters CNIC. 3. Use case resumes on step 8. <p>7b. CNIC field contains alphabet or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “CNIC is invalid”. 2. Actor enters CNIC. 3. Use case resumes on step 8. <p>7c. CNIC length is not 13,</p> <ol style="list-style-type: none"> 1. System displays message “CNIC is invalid”. 2. Actor enters CNIC. 3. Use case resumes on step 8. <p>8a. Mobile number field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Mobile number is required”. 2. Actor enters mobile number. 3. Use case resumes on step 9.

	<p>8b. Mobile number field contains alphabet or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Mobile number is invalid”. 2. Actor enters mobile number. 3. Use case resumes on step 9. <p>8c. Mobile number length is not 11,</p> <ol style="list-style-type: none"> 1. System displays message “Mobile number is invalid”. 2. Actor enters mobile number. 3. Use case resumes on step 9. <p>9. Address field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Address is required”. 2. Actor enters address. 3. Use case resumes on step 10. <p>10. Email address format is invalid,</p> <ol style="list-style-type: none"> 1. System displays message “Email address is invalid”. 2. Actor enters email address. 3. Use case resumes on step 11. <p>11a. Emergency number field contains alphabet or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Emergency number is invalid”. 2. Actor enters emergency number. 3. Use case resumes on step 12. <p>11b. Emergency number length is not 11,</p> <ol style="list-style-type: none"> 1. System displays message “Emergency number is invalid”. 2. Actor enters emergency number. 3. Use case resumes on step 12. <p>12. Account type is not selected,</p> <ol style="list-style-type: none"> 1. System displays message “Account type is required”. 2. Actor enters account type. 3. Use case resumes on step 13. <p>13a. Opening balance field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Opening balance is required”. 2. Actor enters opening balance. 3. Use case resumes on step 14. <p>13b. Opening balance field contains alphabet or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Opening balance is invalid”. 2. Actor enters opening balance. 3. Use case resumes on step 14. <p>14. Opening balance type is not selected,</p> <ol style="list-style-type: none"> 1. System displays message “Balance type is required”. 2. Actor enters balance type. 3. Use case resumes on step 15. <p>Update Account</p> <p>5a. First name field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “First name is required”. 2. Actor enters first name. 3. Use case resumes on step 6. <p>5b. First name field contains number or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Invalid first name”. 2. Actor enters first name. 3. Use case resumes on step 6. <p>6a. Last name field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Last name is required”. 2. Actor enters last name.
--	---

	<ul style="list-style-type: none"> 3. Use case resumes on step 7. 6b. Last name field contains number or special characters, <ul style="list-style-type: none"> 1. System displays message “Last name is required”. 2. Actor enters last name. 3. Use case resumes on step 7. 7a. CNIC field is empty, <ul style="list-style-type: none"> 1. System displays message “CNIC is required”. 2. Actor enters cnic. 3. Use case resumes on step 8. 7b. CNIC field contains alphabet or special characters, <ul style="list-style-type: none"> 1. System displays message “CNIC is invalid”. 2. Actor enters cnic. 3. Use case resumes on step 8. 7c. CNIC length is not 13, <ul style="list-style-type: none"> 1. System displays message “CNIC is invalid”. 2. Actor enters cnic. 3. Use case resumes on step 8. 8a. Mobile number field is empty, <ul style="list-style-type: none"> 1. System displays message “Mobile number is required”. 2. Actor enters mobile number. 3. Use case resumes on step 9. 8b. Mobile number field contains alphabet or special characters, <ul style="list-style-type: none"> 1. System displays message “Mobile number is invalid”. 2. Actor enters mobile number. 3. Use case resumes on step 9. 8c. Mobile number length is not 11, <ul style="list-style-type: none"> 1. System displays message “Mobile number is invalid”. 2. Actor enters mobile number. 3. Use case resumes on step 9. 9. Address field is empty, <ul style="list-style-type: none"> 1. System displays message “Address is required”. 2. Actor enters address. 3. Use case resumes on step 10. 10. Email address format is invalid, <ul style="list-style-type: none"> 1. System displays message “Email address is invalid”. 2. Actor enters email address. 3. Use case resumes on step 11. 11a. Emergency number field contains alphabet or special characters, <ul style="list-style-type: none"> 1. System displays message “Emergency number is invalid”. 2. Actor enters emergency number. 3. Use case resumes on step 12. 11b. Emergency number length is not 11, <ul style="list-style-type: none"> 1. System displays message “Emergency number is invalid”. 2. Actor enters emergency number. 3. Use case resumes on step 12. 12. Account type is not selected, <ul style="list-style-type: none"> 1. System displays message “Account type is required”. 2. Actor enters account type. 3. Use case resumes on step 13. 13a. Opening balance field is empty, <ul style="list-style-type: none"> 1. System displays message “Opening balance is required”. 2. Actor enters opening balance. 3. Use case resumes on step 14.
--	---

	<p>13b. Opening balance field contains alphabet or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Opening balance is invalid”. 2. Actor enters opening balance. 3. Use case resumes on step 14. <p>14. Opening balance type is not selected,</p> <ol style="list-style-type: none"> 1. System displays message “Balance type is required”. 2. Actor enters balance type. 3. Use case resumes on step 15. <p>Remove Account</p> <p>3. Selected account has entries in ledger,</p> <ol style="list-style-type: none"> 1. System displays message “This account can’t be deleted”. 2. Use case ends here.
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	N.A
Assumptions:	N.A
Notes and Issues:	N.A

Table 8(3.5): UC-4 Manage Capital

Use Case ID:	UC-4		
Use Case Name:	Manage Capital		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	Accounts Manager		
Description:	Actor would be add, remove and update capital.		
Trigger:	Actor navigates to capital management.		
Preconditions:	N.A		
Post conditions:	<p>Add Capital</p> <ol style="list-style-type: none"> 1. System adds a new capital record. 2. System adds to entry to cash ledger debit. 3. System adds to entry to account ledger credit. 4. System displays message “Capital Added”. <p>Update Capital</p> <ol style="list-style-type: none"> 1. System updates capital record. 2. System updates entry to cash ledger debit. 3. System updates entry to account ledger credit. 4. System displays message “Capital Updated”. <p>Remove Capital</p> <ol style="list-style-type: none"> 1. System deletes capital record. 2. System displays message “Capital Deleted”. 		
Normal Flow:	<p>Add Capital</p> <ol style="list-style-type: none"> 1. Actor navigates to capital management. 2. System displays list of investors. 3. Actor selects add capital option. 4. System displays add capital form. 5. Actor selects investor. 6. Actor enters capital amount. 7. Actor sends a request to add capital. 8. System adds a new capital record. 9. System adds to entry to cash ledger debit. 10. System adds to entry to account ledger credit. 11. System displays message “Capital Added”. <p>Update Capital</p> <ol style="list-style-type: none"> 1. Actor navigates to capital management. 2. System displays list of investors. 3. Actor selects edit capital option from the list. 4. System displays edit capital form. 5. Actor selects investor. 6. Actor enters capital amount. 7. Actor sends a request to edit capital. 8. System updates capital record. 9. System updates entry to cash ledger debit. 10. System updates entry to account ledger credit. 11. System displays message “Capital Updated”. <p>Remove Capital</p> <ol style="list-style-type: none"> 1. Actor navigates to capital management. 2. System displays list of investors. 3. Actor sends a request to delete capital from list. 4. System deletes capital record. 5. System displays message “Capital Deleted”. 		

Alternative Flows:	N.A
Exceptions:	<p>Add Capital</p> <p>5. Account is not selected,</p> <ol style="list-style-type: none"> 1. System displays message “Account is required”. 2. Actor selects account. 3. Use case resumes on step 6. <p>6a. Amount field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Amount is required”. 2. Actor enters amount. 3. Use case resumes on step 7. <p>6b. Amount field contains special character or alphabet,</p> <ol style="list-style-type: none"> 1. System displays message “Amount is invalid”. 2. Actor enters amount. 3. Use case resumes on step 7. <p>Update Capital</p> <p>5. Account is not selected,</p> <ol style="list-style-type: none"> 1. System displays message “Account is required”. 2. Actor selects account. 3. Use case resumes on step 6. <p>6a. Amount field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Amount is required”. 2. Actor enters amount. 3. Use case resumes on step 7. <p>6b. Amount field contains special character or alphabet,</p> <ol style="list-style-type: none"> 1. System displays message “Amount is invalid”. 2. Actor enters amount. 3. Use case resumes on step 7.
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	N.A
Assumptions:	N.A
Notes and Issues:	N.A

Table 9(3.5): UC-5 Generate Reports

Use Case ID:	UC-5		
Use Case Name:	Generate Reports		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	Accounts Manager		
Description:	Actor would be able to generate reports.		
Trigger:	Actor navigates to reports.		
Preconditions:	Actor is authenticated to application.		
Post conditions:	<ol style="list-style-type: none"> 1. System generates report. 2. Report is downloaded to system storage. 		
Normal Flow:	<ol style="list-style-type: none"> 1. Actor navigates to reports. 2. System displays generate report form. 3. Actor selects report type. 4. Actor selects start date. 5. Actor selects end date. 6. Actor sends request to generate report. 7. System generates the report. 8. Report is downloaded to system storage. 		
Alternative Flows:	N.A		
Exceptions:	<ol style="list-style-type: none"> 3. Report date is not selected, <ol style="list-style-type: none"> 1. System displays message “Report type is required”. 2. Actor selects report type. 3. Use case resumes on step 4. 4. Start date is not selected, <ol style="list-style-type: none"> 1. System displays message “Start date is required”. 2. Actor selects start date. 3. Use case resumes on step 5. 5. End date is not selected, <ol style="list-style-type: none"> 1. System displays message “End date is required”. 2. Actor selects end date. 3. Use case resumes on step 6. 		
Includes:	N.A		
Frequency of Use:	Could be nearly continuous.		
Special Requirements:	N.A		
Assumptions:	N.A		
Notes and Issues:	N.A		

Table 10(3.5): UC-6 Manage Expense

Use Case ID:	UC-6		
Use Case Name:	Manage Expense		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	Accounts Manager		
Description:	Actor would be able to add, view and delete expenses.		
Trigger:	Actor navigates to Expense Management.		
Preconditions:	Actor is authorized to the application.		
Post conditions:	<p>Add Expense</p> <ol style="list-style-type: none"> 1. System records new expense. 2. System records new ledger entry for cash credit against amount of expense. 3. System records new ledger entry for expense debit against amount of expense. 4. System displays message “Expense Added”. <p>Delete Expense</p> <ol style="list-style-type: none"> 1. System removes expense record. 2. System removes cash ledger entry recorded against this expense. 3. System removes expense ledger entry recorded against this expense. 4. System displays message “Expense Deleted”. 		
Normal Flow:	<p>Add Expense</p> <ol style="list-style-type: none"> 1. Actor navigates to Expense Management. 2. System display list of expenses. 3. Actor selects add new expense. 4. Actor enters expense title. 5. Actor enters expense amount. 6. Actor sends request to add new expense. 7. System records new expense. 8. System records new ledger entry for cash credit against amount of expense. 9. System records new ledger entry for expense debit against amount of expense. 10. System displays message “Expense Added”. <p>Delete Expense</p> <ol style="list-style-type: none"> 1. Actor navigates to Expense Management. 2. System display list of expenses. 3. Actor deletes expense from the list. 4. System asks for confirmation. 5. Actor confirms expense deletion. 6. System removes expense record. 7. System removes cash ledger entry recorded against this expense. 8. System removes expense ledger entry recorded against this expense. 9. System displays message “Expense Deleted”. 		

Alternative Flows:	N.A
Exceptions:	<p>Add Expense</p> <p>4. Expense title is empty</p> <ol style="list-style-type: none"> 1. System displays message “Expense title is required”. 2. Actor enters expense title. 3. Use case resume on step 5 <p>5a. Expense amount is a negative number</p> <ol style="list-style-type: none"> 1. System display message “Expense amount should be greater than 0”. 2. Actor enters valid expense amount. 3. Use case resume on step 6 <p>5b. Expense amount field is empty</p> <ol style="list-style-type: none"> 1. System displays message “Expense amount is required”. 2. Actor enters expense amount. 3. Use case resume on step 6 <p>5c. Expense amount is not a number</p> <ol style="list-style-type: none"> 1. System displays message “Expense amount should be a number”. 2. Actor enter valid expense amount. 3. Use case resume on step 6 <p>7. System fail to record new expense record</p> <ol style="list-style-type: none"> 1. System tries again to add the record. 2. Use case resume on step 8 <p>8. System fail to record new cash ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add the record. 2. Use case resume on step 9 <p>9. System fail to record new expense ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add the record. 2. Use case resume on step 10 <p>Delete Expense</p> <p>6. System fails to delete the expense record</p> <ol style="list-style-type: none"> 1. System tries again to delete the record. 2. Use case resumes on step 7 <p>7. System fails to delete the cash ledger entry record</p> <ol style="list-style-type: none"> 1. System tries again to delete the record. 2. Use case resumes on step 8 <p>8. System fails to delete the expense ledger entry record</p> <ol style="list-style-type: none"> 1. System tries again to delete the record. 2. Use case resumes on step 9
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	Internet Connection
Assumptions:	N.A
Notes and Issues:	N.A

Table 11(3.5): UC-7 Manage Payments

Use Case ID:	UC-7		
Use Case Name:	Manage Payments		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 14, 2022
Actors:	Accounts Manager		
Description:	Actor would be able to payment in and payment out		
Trigger:	Actor navigates to Payment Management.		
Preconditions:	Account Manager is authorized to the application.		
Post conditions:	<p>Payment In</p> <ol style="list-style-type: none"> 1. System updates payment record. 2. System records new ledger entry for cash account debit against received amount. 3. System records new ledger entry for customer account credit against received amount. 4. System display message "Payment Received". <p>Payment Out</p> <ol style="list-style-type: none"> 1. System updates payment record. 2. System records new ledger entry for cash account credit against received amount. 3. System records new ledger entry for vendor account debit against received amount. 4. System display message "Payment Paid". 		
Normal Flow:	<p>Payment In</p> <ol style="list-style-type: none"> 1. Actor navigates to Payment Management. 2. Actor selects customer account. 3. Actor selects sale invoice. 4. System displays receivable amount. 5. Actor enters amount received. 6. Actor sends request to update receivable payment. 7. System updates payment record. 8. System records new ledger entry for cash account debit against received amount. 9. System records new ledger entry for customer account credit against received amount. 10. System display message "Payment Received". <p>Payment Out</p> <ol style="list-style-type: none"> 1. Actor navigates to Payment Management. 2. Actor selects vendor account. 3. Actor selects purchase invoice. 4. System displays payable amount. 5. Actor enters amount paid. 6. Actor sends request to update payable payment. 7. System updates payment record. 8. System records new ledger entry for cash account credit against received amount. 9. System records new ledger entry for vendor account debit against received amount. 10. System display message "Payment Paid". 		
Alternative Flows:	N.A		

Exceptions:	<p>Payment In</p> <ol style="list-style-type: none"> 2. Customer account not selected. <ol style="list-style-type: none"> 1. System display message “Customer account is required”. 2. Actor selects customer account. 3. Use case resumes on step 3. 3. Sale invoice not selected. <ol style="list-style-type: none"> 1. System display message “Sale invoice is required”. 2. Actor selects sale invoice. 3. Use case resumes on step 4. 5a. Amount field is empty. <ol style="list-style-type: none"> 1. System display message “Amount is required”. 2. Actor enters valid amount. 3. Use case resume on step 6. 5b. Amount field contains alphabet. <ol style="list-style-type: none"> 1. System display message “Please enter amount in numbers only”. 2. Actor enters valid amount. 3. Use case resumes on step 6. 5c. Amount is greater than receivable amount. <ol style="list-style-type: none"> 1. System display message “Amount can’t be greater than receivable amount”. 2. Actor enters valid amount. 3. Use case resumes on step 6. 7. System failed to update payment record. <ol style="list-style-type: none"> 1. System tries again to update payment. 2. Use case resumes on step 8. 8. System failed to record new ledger entry for cash account. <ol style="list-style-type: none"> 1. System tries again to record new ledger entry for cash account. 2. Use case resumes on step 9. 9. System failed to record new ledger entry for customer account. <ol style="list-style-type: none"> 1. System tries again to record new ledger entry for customer account. 2. Use case resumes on step 10. <p>Payment Out</p> <ol style="list-style-type: none"> 2. Vendor account not selected. <ol style="list-style-type: none"> 1. System displays message “Vendor account is required”. 2. Actor selects vendor account. 3. Use case resumes on step 3. 3. Purchase invoice not selected. <ol style="list-style-type: none"> 1. System displays message “Purchase invoice is required”. 2. Actor selects purchase invoice. 3. Use case resumes on step 4. 5a. Amount field is empty. <ol style="list-style-type: none"> 1. System display message “Amount is required”. 2. Actor enters valid amount. 3. Use case resume on step 6. 5b. Amount field contains alphabet. <ol style="list-style-type: none"> 1. System display message “Please enter amount in numbers only”. 2. Actor enters valid amount. 3. Use case resumes on step 6. 5c. Amount is greater than payable amount. <ol style="list-style-type: none"> 1. System display message “Amount can’t be greater than payable amount”. 2. Actor enters valid amount. 3. Use case resumes on step 6.
--------------------	---

	<p>7. System failed to update payment record.</p> <ol style="list-style-type: none"> 1. System tries again to update payment. 2. Use case resumes on step 8. <p>8. System failed to record new ledger entry for cash account.</p> <ol style="list-style-type: none"> 1. System tries again to record new ledger entry for cash account. 2. Use case resumes on step 9. <p>9. System failed to record new ledger entry for customer account.</p> <ol style="list-style-type: none"> 1. System tries again to record new ledger entry for customer account. 2. Use case resumes on step 10.
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	Internet Connection
Assumptions:	N.A
Notes and Issues:	N.A

Table 12(3.5): UC-8 Manage Employee Requests

Use Case ID:	UC-8		
Use Case Name:	Manage Employee Requests		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	Accounts Manager		
Description:	Actor would be able to approve or reject requests.		
Trigger:	Actor navigates to Employee Requests.		
Preconditions:	Account Manager is authorized to the application.		
Post conditions:	<ol style="list-style-type: none"> 1. System record new employee details. 2. System display message "Request Approved". 		
Normal Flow:	<ol style="list-style-type: none"> 1. Actor navigates to Account Requests. 2. System displays all employee requests. 3. Actor views the request. 4. Actor approves the employee creation request. 5. System asks for confirmation to approve request. 6. Actor confirms request approval. 7. System records new employee details. 8. System displays message "Request Approved". 		
Alternative Flows:	N.A		
Exceptions:	<ol style="list-style-type: none"> 4. Actor rejects employee creation request <ol style="list-style-type: none"> 1. System updates request status to "Rejected". 2. Use case ends here. 7. System failed to record new employee details. <ol style="list-style-type: none"> 1. System tries again to record employee details. 2. Use case resumes on step 8. 		
Includes:	N.A		
Frequency of Use:	Could be nearly continuous.		
Special Requirements:	Internet Connection		
Assumptions:	N.A		
Notes and Issues:	N.A		

Table 13(3.5): UC-9 Manage Purchases

Use Case ID:	UC-9		
Use Case Name:	Manage Purchases		
Created By:	M. Haris Siddiqui	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	Procurement manager		
Description:	Actor would be able to add, view purchase and add, view purchase return.		
Trigger:	Actor navigates to sale management.		
Preconditions:	Actor is authorized to the application.		
Post conditions:	<p>Add Purchase</p> <ol style="list-style-type: none"> 1. System record new purchase 2. System record new ledger entry for cash account credit 3. System record new ledger entry for raw material account debit 4. System record new ledger entry for payable account debit 5. System record new ledger entry for vendor account credit 6. System display message “Purchase added successfully” <p>View Purchase</p> <ol style="list-style-type: none"> 1. System display purchase details <p>Return Purchase</p> <ol style="list-style-type: none"> 1. System add new return record 2. System add new ledger entry for raw material account credit 3. System add new ledger entry for cash account debit 4. System add new ledger entry for vendor account credit 5. System add new ledger entry for payable account debit 6. System display message “Purchase return added” <p>View Invoice</p> <ol style="list-style-type: none"> 1. System display invoice details 		
Normal Flow:	<p>Add Purchase</p> <ol style="list-style-type: none"> 1. Actor navigate to purchase management 2. System display list of all purchases 3. Actor select add new purchase 4. Actor select the vendor 5. Actor enter the paid amount 6. Actor select the raw material 7. Actor enter the raw material quantity 8. Actor enter per unit price of raw material 9. Actor send request to add raw material in list <p>Actor repeats step 6-9 until indicates done</p> <ol style="list-style-type: none"> 10. Actor enter expense title 11. Actor enter expense cost 12. Actor send request to add expense in list <p>Actor repeats step 10-12 until indicates done</p> <ol style="list-style-type: none"> 13. Actor send request to add new purchase 14. System record new purchase 15. System record new ledger entry for cash account credit 16. System record new ledger entry for raw material account debit 17. System record new ledger entry for payable account debit 18. System record new ledger entry for vendor account credit 19. System display message “Purchase added successfully” <p>View Purchase</p> <ol style="list-style-type: none"> 1. Actor navigate to purchase management 2. Actor send request to view purchase details 		

	<p>3. System display purchase details</p> <p>Return Purchase</p> <ol style="list-style-type: none"> 1. Actor navigate to purchase management 2. Actor select purchase return 3. Actor select item from item list 4. Actor enter return quantity 5. Actor send a request to return item <p>Actor repeats step 3-5 until indicates done</p> <ol style="list-style-type: none"> 6. Actor send a request to add new return 7. System add new return record 8. System add new ledger entry for raw material account credit 9. System add new ledger entry for cash account debit 10. System add new ledger entry for vendor account credit 11. System add new ledger entry for payable account debit 12. System display message “Purchase return added” <p>View Invoice</p> <ol style="list-style-type: none"> 1. Actor navigate to purchase management 2. Actor send a request to view invoice 3. System display invoice details
Alternative Flows:	N.A
Exceptions:	<p>Add Purchase</p> <p>4. Vendor field is not selected</p> <ol style="list-style-type: none"> 1. System display message “Vendor field is required” 2. Actor select vendor 3. Use case resumes on step 5 <p>5a. Paid amount field is empty</p> <ol style="list-style-type: none"> 1. System display message “Paid amount field is required” 2. Actor enter paid amount 3. Use case resumes on step 6 <p>5b. Paid amount field contains alphabet</p> <ol style="list-style-type: none"> 1. System display message “Paid amount is invalid” 2. Actor enter valid paid amount 3. Use case resumes on step 6 <p>5c. Paid amount is a negative number</p> <ol style="list-style-type: none"> 1. System display message “Paid amount is invalid” 2. Actor enter valid paid amount 3. Use case resumes on step 6 <p>6. Raw material field is not selected</p> <ol style="list-style-type: none"> 1. System display message “Raw material field is required” 2. Actor select product 3. Use case resumes on step 7 <p>7a. Raw material quantity field is empty</p> <ol style="list-style-type: none"> 1. System display message “Raw material quantity field is required” 2. Actor enter raw material quantity 3. Use case resumes on step 8

	<p>7b. Raw material quantity field contains alphabet</p> <ol style="list-style-type: none"> 1. System display message “Raw material quantity is invalid” 2. Actor enter valid raw material quantity 3. Use case resumes on step 8 <p>7c. Raw material quantity is a negative number</p> <ol style="list-style-type: none"> 1. System display message “Raw material quantity is invalid” 2. Actor enter valid raw material quantity 3. Use case resumes on step 8 <p>8a. Per unit price field is empty</p> <ol style="list-style-type: none"> 1. System display message “Per unit price field is required” 2. Actor enter per unit price 3. Use case resumes on step 9 <p>8b. Per unit price field contains alphabet</p> <ol style="list-style-type: none"> 1. System display message “Per unit price is invalid” 2. Actor enter valid per unit price 3. Use case resumes on step 9 <p>8c. Per unit price is a negative number</p> <ol style="list-style-type: none"> 1. System display message “Per unit price is invalid” 2. Actor enter valid per unit price 3. Use case resumes on step 9 <p>10. Expense title field is empty</p> <ol style="list-style-type: none"> 1. System display message “Expense title field is required” 2. Actor enter expense title 3. Use case resume on step 11 <p>11a. Expense cost field is empty</p> <ol style="list-style-type: none"> 1. System display message “Expense cost field is required” 2. Actor enter Expense cost 3. Use case resumes on step 12 <p>11b. Expense cost field contains alphabet</p> <ol style="list-style-type: none"> 1. System display message “Expense cost is invalid” 2. Actor enter valid Expense cost 3. Use case resumes on step 12 <p>11c. Expense cost is a negative number</p> <ol style="list-style-type: none"> 1. System display message “Expense cost is invalid” 2. Actor enter valid Expense cost 3. Use case resumes on step 12 <p>14. System failed to add new record</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 15 <p>15. System failed to add new cash ledger entry</p>
--	--

	<ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 16 <p>16. System failed to add new raw material ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 17 <p>17. System failed to add new payable ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 18 <p>18. System failed to add new vendor ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 19 <p>Return Purchase</p> <p>3. Item field is not selected</p> <ol style="list-style-type: none"> 1. System display message “Item field is required” 2. Actor select item from the list 3. Use case resumes on step 4 <p>4a. Return quantity field is empty</p> <ol style="list-style-type: none"> 1. System display message “Return quantity field is required” 2. Actor enter return quantity 3. Use case resumes on step 5 <p>4b. Return quantity field contains alphabet</p> <ol style="list-style-type: none"> 1. System display message “Return quantity is invalid” 2. Actor enter valid return quantity 3. Use case resumes on step 5 <p>4c. Return quantity is a negative number</p> <ol style="list-style-type: none"> 1. System display message “Return quantity is invalid” 2. Actor enter valid return quantity 3. Use case resumes on step 5 <p>7. System failed to add new record</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 8 <p>8. System failed to add new cash ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 9 <p>9. System failed to add new raw material ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 10 <p>10. System failed to add new payable ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record
--	--

	<ul style="list-style-type: none"> 2. Use case resumes on step 11 <p>11. System failed to add new vendor ledger entry</p> <ul style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 12
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	Internet Connection
Assumptions:	N.A
Notes and Issues:	N.A

Table 14(3.5): UC-10 Manage Production Requests

Use Case ID:	UC-10		
Use Case Name:	Manage Production Requests		
Created By:	M. Zeeshan Fiaz	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	Procurement Manager		
Description:	Actor would be able to approve or reject requests.		
Trigger:	Actor navigates to Production Requests.		
Preconditions:	Procurement Manager is authorized to the application.		
Post conditions:	<ol style="list-style-type: none"> 1. System record new production details. 2. System records a ledger for the raw materials credit against the request. 3. System records a ledger for the production debit against the request. 4. System display message “Request Approved”. 		
Normal Flow:	<ol style="list-style-type: none"> 1. Actor navigates to Production Requests. 2. System displays all add production requests. 3. Actor views the request. 4. Actor approves the add product request. 5. System asks for confirmation to approve request. 6. Actor confirms request approval. 7. System records new production details. 8. System records new ledger entry for the raw materials account credit against the request. 9. System records new ledger entry for the product account debit against the request. 10. System displays message “Request Approved”. 		
Alternative Flows:	N.A		
Exceptions:	<ol style="list-style-type: none"> 4. Actor rejects production request. <ol style="list-style-type: none"> 1. System updates request status to “Rejected”. 2. Use case ends here. 7. System failed to record new production details. <ol style="list-style-type: none"> 1. System tries again to record production details. 2. Use case resumes on step 8. 8. System failed to create a ledger for raw materials account. <ol style="list-style-type: none"> 1. System tries again to record new ledger entry for raw materials account. 2. Use Case resume on step 9. 9. System failed to create a ledger for product account. <ol style="list-style-type: none"> 1. System tries again to record new ledger entry for product account. 2. Use case resumes on step 10. 		
Includes:	N.A		
Frequency of Use:	Could be nearly continuous.		
Special Requirements:	Internet Connection		
Assumptions:	N.A		
Notes and Issues:	N.A		

Table 15(3.5): UC-11 Manage Sales

Use Case ID:	UC-11		
Use Case Name:	Manage Sales		
Created By:	M. Haris Siddiqui	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	Sale manager		
Description:	Actor would be able to add, view sale and add, view sale return.		
Trigger:	Actor navigates to sale management.		
Preconditions:	Actor is authorized to the application.		
Post conditions:	<ol style="list-style-type: none"> Add Sale System record new sale System record new ledger entry for cash account debit System record new ledger entry for product account credit System record new ledger entry for receivable account debit System record new ledger entry for customer account credit System display message “Sale added successfully” <p>View Sale</p> <ol style="list-style-type: none"> System display sale details <p>Return Sale</p> <ol style="list-style-type: none"> System add new return record System add new ledger entry for product account debit System add new ledger entry for cash account credit System add new ledger entry for customer account debit System add new ledger entry for receivable account credit System display message “Sale return added” <p>View Invoice</p> <ol style="list-style-type: none"> System display invoice details 		
Normal Flow:	<p>Add Sale</p> <ol style="list-style-type: none"> Actor navigate to sale management System display list of all sales Actor select add new sale Actor select the customer Actor enter the paid amount Actor select the product Actor enter the product quantity Actor enter per unit price of product Actor send request to add product in list <p>Actor repeats step 6-9 until indicates done</p> <ol style="list-style-type: none"> Actor enter expense title Actor enter expense cost Actor send request to add expense in list <p>Actor repeats step 10-12 until indicates done</p> <ol style="list-style-type: none"> Actor send request to add new sale System record new sale System record new ledger entry for cash account debit System record new ledger entry for product account credit System record new ledger entry for receivable account debit System record new ledger entry for customer account credit System display message “Sale added successfully” <p>View Sale</p> <ol style="list-style-type: none"> Actor navigate to sale management Actor send request to view sale details 		

	<p>3. System display sale details</p> <p>Return Sale</p> <ol style="list-style-type: none"> 1. Actor navigate to sale management 2. Actor select sale return 3. Actor select item from item list 4. Actor enter return quantity 5. Actor send a request to return item <p>Actor repeats step 3-5 until indicates done</p> <ol style="list-style-type: none"> 6. Actor send a request to add new return 7. System add new return record 8. System add new ledger entry for product account debit 9. System add new ledger entry for cash account credit 10. System add new ledger entry for customer account debit 11. System add new ledger entry for receivable account credit 12. System display message “Sale return added” <p>View Invoice</p> <ol style="list-style-type: none"> 1. Actor navigate to sale management 2. Actor send a request to view invoice 3. System display invoice details
Alternative Flows:	N.A
Exceptions:	<p>Add Sale</p> <p>4. Customer field is not selected</p> <ol style="list-style-type: none"> 1. System display message “Customer field is required” 2. Actor select customer 3. Use case resumes on step 5 <p>5a. Paid amount field is empty</p> <ol style="list-style-type: none"> 1. System display message “Paid amount field is required” 2. Actor enter paid amount 3. Use case resumes on step 6 <p>5b. Paid amount contains alphabet</p> <ol style="list-style-type: none"> 1. System display message “Paid amount is invalid” 2. Actor enter valid paid amount 3. Use case resumes on step 6 <p>5c. Paid amount is a negative number</p> <ol style="list-style-type: none"> 1. System display message “Paid amount is invalid” 2. Actor enter valid paid amount 3. Use case resumes on step 6 <p>6. Product field is not selected</p> <ol style="list-style-type: none"> 1. System display message “Product field is required” 2. Actor select product 3. Use case resumes on step 7 <p>7a. Product quantity field is empty</p> <ol style="list-style-type: none"> 1. System display message “Product quantity field is required” 2. Actor enter product quantity 3. Use case resumes on step 8

	<p>7b. Product quantity contains alphabet</p> <ol style="list-style-type: none"> 1. System display message “Product quantity is invalid” 2. Actor enter valid product quantity 3. Use case resumes on step 8 <p>7c. Product quantity is a negative number</p> <ol style="list-style-type: none"> 1. System display message “Product quantity is invalid” 2. Actor enter valid product quantity 3. Use case resumes on step 8 <p>8a. Per unit price field is empty</p> <ol style="list-style-type: none"> 1. System display message “Per unit price field is required” 2. Actor enter per unit price 3. Use case resumes on step 9 <p>8b. Per unit price contains alphabet</p> <ol style="list-style-type: none"> 1. System display message “Per unit price is invalid” 2. Actor enter valid per unit price 3. Use case resumes on step 9 <p>8c. Per unit price is a negative number</p> <ol style="list-style-type: none"> 1. System display message “Per unit price is invalid” 2. Actor enter valid per unit price 3. Use case resumes on step 9 <p>10. Expense title field is empty</p> <ol style="list-style-type: none"> 1. System display message “Expense title field is required” 2. Actor enter expense title 3. Use case resume on step 11 <p>11a. Expense cost field is empty</p> <ol style="list-style-type: none"> 1. System display message “Expense cost field is required” 2. Actor enter Expense cost 3. Use case resumes on step 12 <p>11b. Expense cost contains alphabet</p> <ol style="list-style-type: none"> 1. System display message “Expense cost is invalid” 2. Actor enter valid Expense cost 3. Use case resumes on step 12 <p>11c. Expense cost is a negative number</p> <ol style="list-style-type: none"> 1. System display message “Expense cost is invalid” 2. Actor enter valid Expense cost 3. Use case resumes on step 12 <p>14. System failed to add new record</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 15 <p>15. System failed to add new cash ledger entry</p>
--	--

	<ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 16 <p>16. System failed to add new product ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 17 <p>17. System failed to add new receivable ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 18 <p>18. System failed to add new customer ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 19 <p>Return Sale</p> <p>3. Item field is not selected</p> <ol style="list-style-type: none"> 1. System display message “Item field is required” 2. Actor select item from the list 3. Use case resumes on step 4 <p>4a. Return quantity field is empty</p> <ol style="list-style-type: none"> 1. System display message “Return quantity field is required” 2. Actor enter return quantity 3. Use case resumes on step 5 <p>4b. Return quantity contains alphabet</p> <ol style="list-style-type: none"> 1. System display message “Return quantity is invalid” 2. Actor enter valid return quantity 3. Use case resumes on step 5 <p>4c. Return quantity is a negative number</p> <ol style="list-style-type: none"> 1. System display message “Return quantity is invalid” 2. Actor enter valid return quantity 3. Use case resumes on step 5 <p>7. System failed to add new record</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 8 <p>8. System failed to add new cash ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 9 <p>9. System failed to add new product ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 10 <p>10. System failed to add new receivable ledger entry</p> <ol style="list-style-type: none"> 1. System tries again to add record
--	--

	<ul style="list-style-type: none"> 2. Use case resumes on step 11 <p>11. System failed to add new customer ledger entry</p> <ul style="list-style-type: none"> 1. System tries again to add record 2. Use case resumes on step 12
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	Internet Connection
Assumptions:	N.A
Notes and Issues:	N.A

Table 16(3.5): UC-12 Manage Products

Use Case ID:	UC-12		
Use Case Name:	Manage Products		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	System Manager		
Description:	Actor would be able to add, remove and update products.		
Trigger:	Actor navigates to product management.		
Preconditions:	Actor is authorized to the application.		
Post conditions:	<ol style="list-style-type: none"> 1. System records a new product. 2. System display message “Product added successfully”. 		
Normal Flow:	<ol style="list-style-type: none"> 1. Actor navigates to product management. 2. System displays the list of products. 3. Actor enters product title. 4. Actor enters product code. 5. Actor sends the request to add product. 6. System asks for confirmation to add product. 7. Actor confirms the request. 8. System records a new product. 9. System display message “Product added successfully”. 		
Alternative Flows:	N.A		
Exceptions:	<ol style="list-style-type: none"> 3. Product title is empty. <ol style="list-style-type: none"> 1. System displays message “Product title is required”. 2. Actor enters valid product title. 3. Use case resumes on step 4. 4a. Product code is empty. <ol style="list-style-type: none"> 1. System displays message “Product code is required” 2. Actor enters valid product. 3. Use case resumes on step 5. 4b. Product code is already in record. <ol style="list-style-type: none"> 1. System displays message “Product is already in record” 2. Actor enter valid product code 3. Use case resumes on step 5. 		
Includes:	N.A		
Frequency of Use:	Could be nearly continuous.		
Special Requirements:	Internet Connection		
Assumptions:	N.A		
Notes and Issues:	N.A		

Table 17(3.5): UC-13 Manage Raw Material

Use Case ID:	UC-13		
Use Case Name:	Manage Raw Material		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	System Manager		
Description:	Actor would be able to add, remove and update raw material.		
Trigger:	Actor navigates to raw material management.		
Preconditions:	Actor is authorized to the application.		
Post conditions:	<ol style="list-style-type: none"> 1. System records a new raw material. 2. System display message “Raw material added successfully”. 		
Normal Flow:	<ol style="list-style-type: none"> 1. Actor navigates to raw material management. 2. System displays the list of all raw materials. 3. Actor enters raw material title. 4. Actor sends the request to add raw material. 5. System asks for confirmation to add raw material. 6. Actor confirms the request. 7. System records a new raw material. 8. System displays message “Raw material added successfully”. 		
Alternative Flows:	N.A		
Exceptions:	<ol style="list-style-type: none"> 3a. Raw Martial title is empty. <ol style="list-style-type: none"> 1. System displays message “Raw Martial title is required”. 2. Actor enters valid raw martial title. 4. Use case resumes on step 4. 3b. Raw Martial title is already in record. <ol style="list-style-type: none"> 1. System displays message “Raw Martial title is already in record” 2. Actor enters valid raw martial title 3. Use case resumes on step 4. 		
Includes:	N.A		
Frequency of Use:	Could be nearly continuous.		
Special Requirements:	Internet Connection		
Assumptions:	N.A		
Notes and Issues:	N.A		

Table 18(3.5): UC-14 Manage Departments

Use Case ID:	UC-14		
Use Case Name:	Manage Departments		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	System Manager		
Description:	Actor would be able to add, remove and update departments.		
Trigger:	Actor navigates to department management.		
Preconditions:	Actor is authorized to the application.		
Post conditions:	<ol style="list-style-type: none"> 1. System records a new department. 2. System display message “Department added successfully”. 		
Normal Flow:	<ol style="list-style-type: none"> 1. Actor navigates to department management. 2. System displays the list of all departments. 3. Actor enters department title. 4. Actor sends a request to add department. 5. System asks for confirmation. 6. Actor confirms the request. 7. System records a new department. 8. System displays message “Department added successfully”. 		
Alternative Flows:	N.A		
Exceptions:	<ol style="list-style-type: none"> 3a. Department title is empty. <ol style="list-style-type: none"> 1. System displays message “Department title is required”. 2. Actor enters valid department title. 3. Use case resumes on step 4. 3b. Department title is already record <ol style="list-style-type: none"> 1. System displays message “Department is already in record”. 2. Actor enters valid department title. 3. Use case resumes on step 4. 		
Includes:	N.A		
Frequency of Use:	Could be nearly continuous.		
Special Requirements:	Internet Connection		
Assumptions:	N.A		
Notes and Issues:	N.A		

Table 19(3.5): UC-15 Manage Backup

Use Case ID:	UC-15		
Use Case Name:	Manage Backup		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	System Manager		
Description:	Actor would be able to create or restore backup.		
Trigger:	Actor navigates to product management.		
Preconditions:	Actor is authorized to the application.		
Post conditions:	Create Backup <ol style="list-style-type: none"> 1. System generates backup. 2. System display message “Backup created successfully” Restore Backup <ol style="list-style-type: none"> 1. System restore backup. 2. System display message “Backup restored successfully” 		
Normal Flow:	Create Backup <ol style="list-style-type: none"> 1. Actor navigates to backup management. 2. Actor sends a request to create backup. 3. System asks for confirmation to create backup. 4. Actor confirms backup request. 5. System generates backup. 6. System display message “Backup created successfully” Restore Backup <ol style="list-style-type: none"> 1. Actor navigates to backup management. 2. Actor selects the backup file. 3. Actor sends a request to restore backup. 4. System asks for confirmation to restore backup. 5. Actor confirms backup request. 6. System restores backup. 7. System displays message “Backup restored successfully” 		
Alternative Flows:	N.A		
Exceptions:	Create Backup <ol style="list-style-type: none"> 5. System failed to generate backup. <ol style="list-style-type: none"> 1. System tries again to generate backup. 2. Use case resume on step 6. Restore Backup <ol style="list-style-type: none"> 6. System failed to restore backup. <ol style="list-style-type: none"> 1. System tries again to restore backup. 2. Use case resume on step 7. 		
Includes:	N.A		
Frequency of Use:	Could be nearly continuous.		
Special Requirements:	Internet Connection		
Assumptions:	N.A		
Notes and Issues:	N.A		

Table 20(3.5): UC-16 Manage Attendance

Use Case ID:	UC-16		
Use Case Name:	Manage Attendance		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	HR Manager		
Description:	Actor would be able to add, view and search attendance.		
Trigger:	Actor navigates to Attendance Management.		
Preconditions:	Actor is authorized to the application.		
Post conditions:	Search Attendance <ol style="list-style-type: none"> 1. System display selected date attendance. Add Attendance <ol style="list-style-type: none"> 1. System records attendance details with current time. 2. System displays message “Attendance marked”. 		
Normal Flow:	Search Attendance <ol style="list-style-type: none"> 1. Actor navigates to attendance management. 2. System will display current date’s attendance. 3. Actor selects date to search attendance. 4. Actor sends request to search for attendance. 5. System display selected date attendance. Mark Attendance <ol style="list-style-type: none"> 1. Actor navigates to attendance management. 2. System will display current date’s attendance. 3. Actor enters employee id. 4. Actor sends request to mark attendance. 5. System gets check-in/check-out time. 6. System records attendance details with current time. 7. System displays message “Attendance marked”. 		
Alternative Flows:	N.A		
Exceptions:	Search Attendance <ol style="list-style-type: none"> 3. Date not selected. <ol style="list-style-type: none"> 1. System display message “Date is required”. 2. HR Manager selects date. 3. Use case resumes at step 4. Mark Attendance <ol style="list-style-type: none"> 3a. Employee ID field is empty. <ol style="list-style-type: none"> 1. System display message “Employee ID is required”. 2. HR Manager enters employee id. 3. Use case resume on step 4. 3b. Employee ID length is not 13. <ol style="list-style-type: none"> 1. System display message “Invalid Employee ID”. 2. HR Manager enters valid employee id. 		

	<ul style="list-style-type: none"> 3. Use case resume on step 4. <p>3c. Employee ID field contains alphabet.</p> <ul style="list-style-type: none"> 1. System display message “Invalid Employee ID”. 2. HR Manager enters valid employee id. 3. Use case resume on step 4. <p>5. System failed to record attendance details.</p> <ul style="list-style-type: none"> 1. System tries again to record attendance. 2. Use case resume on step 6.
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	Internet Connection
Assumptions:	N.A
Notes and Issues:	N.A

Table 21(3.5): UC-17 Manage Employees

Use Case ID:	UC-17		
Use Case Name:	Manage Employees		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	HR Manager		
Description:	Actor would be able to add, remove and update employees.		
Trigger:	Actor navigates to employee management		
Preconditions:	N.A		
Post conditions:	Add Employee <ol style="list-style-type: none"> 1. System adds new account record. 2. System displays message “Add account request initiated”. Update Employee <ol style="list-style-type: none"> 1. System adds new account record. 2. System displays message “Account update request initiated”. Remove Employee <ol style="list-style-type: none"> 1. System deletes account from records. 2. System display message “Account delete request initiated”. 		
Normal Flow:	Add Employee <ol style="list-style-type: none"> 1. Actor navigates to employee management. 2. System displays list of all employees. 3. Actor selects add employee option. 4. System displays add employee form. 5. Actor enters first name. 6. Actor enters last name. 7. Actor enters CNIC number. 8. Actor enters mobile number. 9. Actor enters address. 10. Actor enters email address. 11. Actor enters emergency contact number. 12. Actor selects department. 13. Actor enters salary. 14. Actor sends request to add employee. 15. System adds new employee request record. 16. System displays message “Add employee request initiated”. Update Employee <ol style="list-style-type: none"> 1. Actor navigates to employee management. 2. System displays list of all employees. 3. Actor selects edit employee option from employee list. 4. System displays edit employee form. 5. Actor enters first name. 6. Actor enters last name. 7. Actor enters CNIC number. 8. Actor enters mobile number. 9. Actor enters address. 10. Actor enters email address. 11. Actor enters emergency contact number. 12. Actor selects department. 13. Actor enters salary. 14. Actor sends request to update employee. 15. System records update employee request. 16. System displays message “Employee update request initiated”. 		

	Remove Employee <ol style="list-style-type: none"> 1. Actor navigates to employee management. 2. System displays list of all employee. 3. Actor sends request to delete employee from employee list. 4. System deletes employee from records. 5. System display message “Employee delete request initiated”.
Alternative Flows:	N.A
Exceptions:	Add Employee <ol style="list-style-type: none"> 5a. First name field is empty, <ol style="list-style-type: none"> 1. System displays message “First name is required”. 2. Actor enters first name. 3. Use case resumes on step 6. 5b. First name field contains number or special characters, <ol style="list-style-type: none"> 1. System displays message “Invalid first name”. 2. Actor enters first name. 3. Use case resumes on step 6. 6a. Last name field is empty, <ol style="list-style-type: none"> 1. System displays message “Last name is required”. 2. Actor enters last name. 3. Use case resumes on step 7. 6b. Last name field contains number or special characters, <ol style="list-style-type: none"> 1. System displays message “Last name is required”. 2. Actor enters last name. 3. Use case resumes on step 7. 7a. CNIC field is empty, <ol style="list-style-type: none"> 1. System displays message “CNIC is required”. 2. Actor enters CNIC. 3. Use case resumes on step 8. 7b. CNIC field contains alphabet or special characters, <ol style="list-style-type: none"> 1. System displays message “CNIC is invalid”. 2. Actor enters CNIC. 3. Use case resumes on step 8. 7c. CNIC length is not 13, <ol style="list-style-type: none"> 1. System displays message “CNIC is invalid”. 2. Actor enters CNIC. 3. Use case resumes on step 8. 8a. Mobile number field is empty, <ol style="list-style-type: none"> 1. System displays message “Mobile number is required”. 2. Actor enters mobile number. 3. Use case resumes on step 9. 8b. Mobile number field contains alphabet or special characters, <ol style="list-style-type: none"> 1. System displays message “Mobile number is invalid”. 2. Actor enters mobile number. 3. Use case resumes on step 9. 8c. Mobile number length is not 11, <ol style="list-style-type: none"> 1. System displays message “Mobile number is invalid”. 2. Actor enters mobile number. 3. Use case resumes on step 9. 9. Address field is empty, <ol style="list-style-type: none"> 1. System displays message “Address is required”. 2. Actor enters address. 3. Use case resumes on step 10.

	<p>10. Email address format is invalid,</p> <ol style="list-style-type: none"> 1. System displays message “Email address is invalid”. 2. Actor enters email address. 3. Use case resumes on step 11. <p>11a. Emergency number field contains alphabet or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Emergency number is invalid”. 2. Actor enters emergency number. 3. Use case resumes on step 12. <p>11b. Emergency number length is not 11,</p> <ol style="list-style-type: none"> 1. System displays message “Emergency number is invalid”. 2. Actor enters emergency number. 3. Use case resumes on step 12. <p>12. Department is not selected,</p> <ol style="list-style-type: none"> 1. System displays message “Department is required”. 2. Actor enters department. 3. Use case resumes on step 13. <p>13a. Salary field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Salary is required”. 2. Actor enters salary. 3. Use case resumes on step 14. <p>13b. Salary field contains alphabet or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Salary is invalid”. 2. Actor enters salary. 3. Use case resumes on step 14. <p>Update Employee</p> <p>5a. First name field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “First name is required”. 2. Actor enters first name. 3. Use case resumes on step 6. <p>5b. First name field contains number or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Invalid first name”. 2. Actor enters first name. 3. Use case resumes on step 6. <p>6a. Last name field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Last name is required”. 2. Actor enters last name. 3. Use case resumes on step 7. <p>6b. Last name field contains number or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Last name is required”. 2. Actor enters last name. 3. Use case resumes on step 7. <p>7a. CNIC field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “CNIC is required”. 2. Actor enters CNIC. 3. Use case resumes on step 8. <p>7b. CNIC field contains alphabet or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “CNIC is invalid”. 2. Actor enters CNIC. 3. Use case resumes on step 8. <p>7c. CNIC length is not 13,</p> <ol style="list-style-type: none"> 1. System displays message “CNIC is invalid”. 2. Actor enters CNIC. 3. Use case resumes on step 8.
--	---

	<p>8a. Mobile number field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Mobile number is required”. 2. Actor enters mobile number. 3. Use case resumes on step 9. <p>8b. Mobile number field contains alphabet or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Mobile number is invalid”. 2. Actor enters mobile number. 3. Use case resumes on step 9. <p>8c. Mobile number length is not 11,</p> <ol style="list-style-type: none"> 1. System displays message “Mobile number is invalid”. 2. Actor enters mobile number. 3. Use case resumes on step 9. <p>9. Address field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Address is required”. 2. Actor enters address. 3. Use case resumes on step 10. <p>10. Email address format is invalid,</p> <ol style="list-style-type: none"> 1. System displays message “Email address is invalid”. 2. Actor enters email address. 3. Use case resumes on step 11. <p>11a. Emergency number field contains alphabet or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Emergency number is invalid”. 2. Actor enters emergency number. 3. Use case resumes on step 12. <p>11b. Emergency number length is not 11,</p> <ol style="list-style-type: none"> 1. System displays message “Emergency number is invalid”. 2. Actor enters emergency number. 3. Use case resumes on step 12. <p>12. Department is not selected,</p> <ol style="list-style-type: none"> 1. System displays message “Department is required”. 2. Actor enters department. 3. Use case resumes on step 13. <p>13a. Salary field is empty,</p> <ol style="list-style-type: none"> 1. System displays message “Salary is required”. 2. Actor enters salary. 3. Use case resumes on step 14. <p>13b. Salary field contains alphabet or special characters,</p> <ol style="list-style-type: none"> 1. System displays message “Salary is invalid”. 2. Actor enters salary. 3. Use case resumes on step 14. <p>Remove Employee</p> <p>3. Selected account has entries in ledger,</p> <ol style="list-style-type: none"> 1. System displays message “This employee can’t be deleted”. 2. Use case ends here.
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	N.A
Assumptions:	N.A
Notes and Issues:	N.A

Table 22(3.5): UC-18 Manage Salaries

Use Case ID:	UC-18		
Use Case Name:	Manage Salary		
Created By:	Abdul Raheem	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	HR Manager		
Description:	Actor would be able to generate and pay salaries.		
Trigger:	Actor navigates to Salary Management.		
Preconditions:	Actor is authorized to the application.		
Post conditions:	<p>Generate salaries</p> <ol style="list-style-type: none"> 1. System display pending salaries of all employees. 2. System display message “Salaries Generated”. <p>Pay salary to all</p> <ol style="list-style-type: none"> 1. System records new ledgers entries for employee account debit against salary. 2. System records new ledgers entries for cash account credit against salary. 3. System updates all employees’ salary status. 4. System display message “Salary paid to all employees”. <p>Pay salary to one</p> <ol style="list-style-type: none"> 1. System records new ledger entry for employee account debit against salary. 2. System records new ledger entry for cash account credit against salary. 3. System updates employee salary status. 4. System display message “Salary paid”. 		
Normal Flow:	<p>Generate salaries</p> <ol style="list-style-type: none"> 1. Actor navigates to salary management. 2. Actor sends a request to generate salaries. 3. System fetches salary details. 4. System displays pending salaries of all employees. 5. System displays message “Salaries Generated”. <p>Pay salary to all</p> <ol style="list-style-type: none"> 1. Actor navigates to salary management. 2. Actor sends a request to pay all. 3. System records new ledger entries for employee account debit against salary. 4. System records new ledger entries for cash account credit against salary. 5. System updates all employees’ salary status. 6. System displays message “Salary paid to all employees”. <p>Pay salary to one</p> <ol style="list-style-type: none"> 1. Actor navigates to salary management. 2. Actor sends a request to pay salary of single employee. 3. System records new ledger entry for employee account debit against salary. 4. System records new ledger entry for cash account credit against salary. 5. System updates employee salary status. 6. System displays message “Salary paid”. 		

/Alternative Flows:	N.A
Exceptions:	<p>Generate salaries</p> <p>3. System failed to fetch salary details.</p> <ol style="list-style-type: none"> 1. System tries again to fetch salary details. 2. Use case resumes at step 4. <p>Pay salary to all</p> <p>3. System failed to add new ledger entry for employee accounts</p> <ol style="list-style-type: none"> 1. System tries again to add new ledger entry for employee accounts. 2. Use case resumes at step 4. <p>4. System failed to add new ledger entry for cash account</p> <ol style="list-style-type: none"> 1. System tries again to add new ledger entry for cash account. 2. Use case resumes at step 5. <p>5. System failed to update salary status</p> <ol style="list-style-type: none"> 1. System tries again to update salary status. 2. Use case resumes at step 6. <p>Pay salary to one</p> <p>3. System failed to add new ledger entry for employee account</p> <ol style="list-style-type: none"> 1. System tries again to add new ledger entry for employee account. 2. Use case resumes at step 4. <p>4. System failed to add new ledger entry for cash account</p> <ol style="list-style-type: none"> 1. System tries again to add new ledger entry for cash account. 2. Use case resumes at step 5. <p>5. System failed to update salary status</p> <ol style="list-style-type: none"> 1. System tries again to update salary status. 2. Use case resumes at step 6.
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	Internet Connection
Assumptions:	N.A
Notes and Issues:	N.A

Table 23(3.5): UC-19 Manage Production

Use Case ID:	UC-19		
Use Case Name:	Manage Production		
Created By:	Abu-Bakar Rehan	Last Updated By:	Abdul Raheem
Date Created:	July 03, 2022	Last Revision Date:	Oct 10, 2022
Actors:	Production manager		
Description:	Actor would be able to add, view, update and finish production.		
Trigger:	Actor navigates to production management.		
Preconditions:	Actor is authorized to the application.		
Post conditions:	<p>Add Production</p> <ol style="list-style-type: none"> 1. System records new production request. 2. System displays message “Production request initiated” <p>Update Production</p> <ol style="list-style-type: none"> 1. System records new production update request. 2. System displays message “Production update request initiated” <p>Finish Production</p> <ol style="list-style-type: none"> 1. System adds new product(s) to records. 2. System displays message “Product(s) added to inventory” 		
Normal Flow:	<p>Add Production</p> <ol style="list-style-type: none"> 1. Actor navigates to production management. 2. Actor selects add production option. 3. Actor selects product code. 4. Actor enters quantity of product. 5. Actor selects raw material. 6. Actor enters quantity of raw material. 7. Actor sends request to add item. <p>Actor repeats step 5-7 until indicates done.</p> <ol style="list-style-type: none"> 8. Actor enters expense title. 9. Actor enters expense cost. 10. Actor sends request to add expense. <p>Actor repeats step 8-10 until indicates done.</p> <ol style="list-style-type: none"> 11. Actor sends request to start production. 12. System records new production request. 13. System displays message “Production request initiated” <p>Update Production</p> <ol style="list-style-type: none"> 1. Actor navigates to production management. 2. Actor selects update production option. 3. Actor selects product code. 4. Actor enters quantity of product. 5. Actor selects raw material. 6. Actor selects quantity of raw material. 7. Actor sends request to add item. <p>Actor repeats step 5-7 until indicates done.</p> <ol style="list-style-type: none"> 8. Actor enters expense title. 9. Actor enters expense cost. 10. Actor sends request to add expense. <p>Actor repeats step 8-10 until indicates done.</p> <ol style="list-style-type: none"> 11. Actor sends request to update production. 12. System records new production update request. 13. System displays message “Production update request initiated” <p>Finish Production</p> <ol style="list-style-type: none"> 1. Actor navigates to production management. 		

	<ol style="list-style-type: none"> 2. Actor sends request to finish production. 3. System adds new product(s) to records. 4. System displays message “Product(s) added to inventory”
Alternative Flows:	N.A
Exceptions:	<p>Add Production</p> <ol style="list-style-type: none"> 3. Product code is empty. <ol style="list-style-type: none"> 1. System displays message “Product code is required” 2. Actor selects product code. 3. Use case resumes on step 4. 4a. Product quantity is empty. <ol style="list-style-type: none"> 1. System displays message “Product quantity is required” 2. Actor enters product quantity. 3. Use case resumes on step 5. 4b. Product quantity field contains alphabet. <ol style="list-style-type: none"> 1. System displays message “Product quantity is invalid” 2. Actor enters valid product quantity. 3. Use case resumes on step 5. 4c. Product quantity is a negative number. <ol style="list-style-type: none"> 1. System displays message “Product quantity is invalid”. 2. Actor enters valid product quantity. 3. Use case resumes on step 5. 5. Raw material is empty. <ol style="list-style-type: none"> 1. System displays message “Raw material is required”. 2. Actor selects raw material. 3. Use case resumes on step 6. 6a. Raw material quantity is empty. <ol style="list-style-type: none"> 1. System displays message “Raw material quantity is required”. 2. Actor enters Raw material quantity. 3. Use case resumes on step 7. 6b. Raw material quantity field contains alphabet. <ol style="list-style-type: none"> 1. System displays message “Raw material quantity is invalid”. 2. Actor enters valid raw material quantity. 3. Use case resumes on step 7. 6c. Raw material quantity is non-positive. <ol style="list-style-type: none"> 1. System displays message “Raw material quantity is invalid” 2. Actor enters valid raw material quantity. 3. Use case resumes on step 7. 8. Expense title is empty. <ol style="list-style-type: none"> 1. System displays message “Expense title is required” 2. Actor enters expense title. 3. Use case resumes on step 9. 9a. Expense cost is empty. <ol style="list-style-type: none"> 1. System displays message “Expense cost is required” 2. Actor enters valid expense cost. 3. Use case resumes on step 10. 9b. Expense cost field contains alphabet. <ol style="list-style-type: none"> 1. System displays message “Expense cost is invalid” 2. Actor enters valid expense cost. 3. Use cases resume on step 10. 9c. Expense cost is a negative number. <ol style="list-style-type: none"> 1. System displays message “Expense cost is invalid” 2. Actor enters valid expense cost.

	<ul style="list-style-type: none"> 3. Use case resumes on step 10.
	<ul style="list-style-type: none"> 12. System failed to add new record. <ul style="list-style-type: none"> 1. System tries again to add new record 2. Use case resumes on step 13.
	<p>Update Production</p>
	<ul style="list-style-type: none"> 3. Product code is empty. <ul style="list-style-type: none"> 1. System displays message “Product code is required” 2. Actor selects product code. 3. Use case resumes on step 4.
	<ul style="list-style-type: none"> 4a. Product quantity is empty. <ul style="list-style-type: none"> 1. System displays message “Product quantity is required” 2. Actor enters product quantity. 3. Use case resumes on step 5.
	<ul style="list-style-type: none"> 4b. Product quantity field contains alphabet. <ul style="list-style-type: none"> 1. System displays message “Product quantity is invalid” 2. Actor enters valid product quantity. 3. Use case resumes on step 5.
	<ul style="list-style-type: none"> 4c. Product quantity is a negative number. <ul style="list-style-type: none"> 1. System displays message “Product quantity is invalid” 2. Actor enters valid product quantity. 3. Use case resumes on step 5.
	<ul style="list-style-type: none"> 5. Raw material is empty. <ul style="list-style-type: none"> 1. System displays message “Raw material is required” 2. Actor selects raw material. 3. Use case resumes on step 6.
	<ul style="list-style-type: none"> 6a. Raw material quantity is empty. <ul style="list-style-type: none"> 1. System displays message “Raw material quantity is required” 2. Actor enters Raw material quantity. 3. Use case resumes on step 7.
	<ul style="list-style-type: none"> 6b. Raw material quantity field contains alphabet. <ul style="list-style-type: none"> 1. System displays message “Raw material quantity is invalid” 2. Actor enters valid raw material quantity. 3. Use case resumes on step 7.
	<ul style="list-style-type: none"> 6c. Raw material quantity is a negative number. <ul style="list-style-type: none"> 1. System displays message “Raw material quantity is invalid” 2. Actor enters valid raw material quantity. 3. Use case resumes on step 7.
	<ul style="list-style-type: none"> 8. Expense title is empty. <ul style="list-style-type: none"> 1. System displays message “Expense title is required” 2. Actor enters expense title. 3. Use case resumes on step 9.
	<ul style="list-style-type: none"> 9a. Expense cost is empty. <ul style="list-style-type: none"> 1. System displays message “Expense cost is required” 2. Actor enters valid expense cost. 3. Use case resumes on step 10.
	<ul style="list-style-type: none"> 9b. Expense cost field contains alphabet. <ul style="list-style-type: none"> 1. System displays message “Expense cost is invalid” 2. Actor enters valid expense cost. 3. Use case resume on step 10.
	<ul style="list-style-type: none"> 9c. Expense cost is a negative number. <ul style="list-style-type: none"> 1. System displays message “Expense cost is invalid” 2. Actor enters valid expense cost. 3. Use case resumes on step 10.

	12. System failed to add new record. <ol style="list-style-type: none"> 1. System tries again to add new record 2. Use case resumes on step 13. Finish Production 3. System failed to add new record. <ol style="list-style-type: none"> 1. System tries again to add new record 2. Use case resumes on step 4.
Includes:	N.A
Frequency of Use:	Could be nearly continuous.
Special Requirements:	Internet Connection
Assumptions:	N.A
Notes and Issues:	N.A

3.6 User interfaces (mock screens)

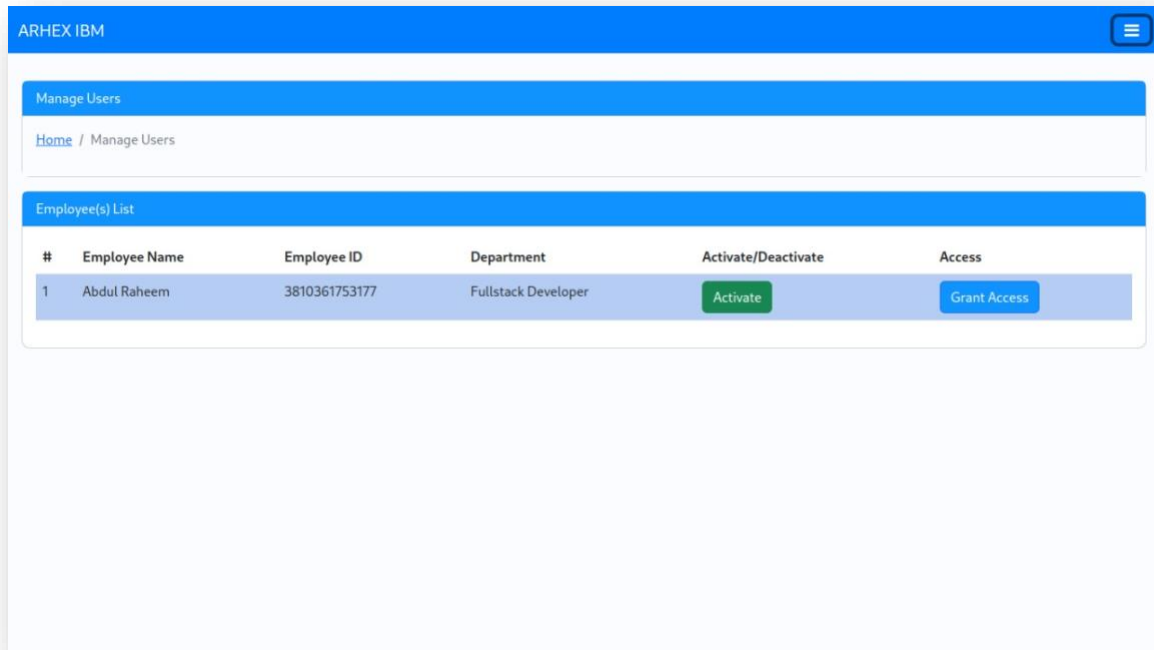


Figure 2(3.6): Manage User

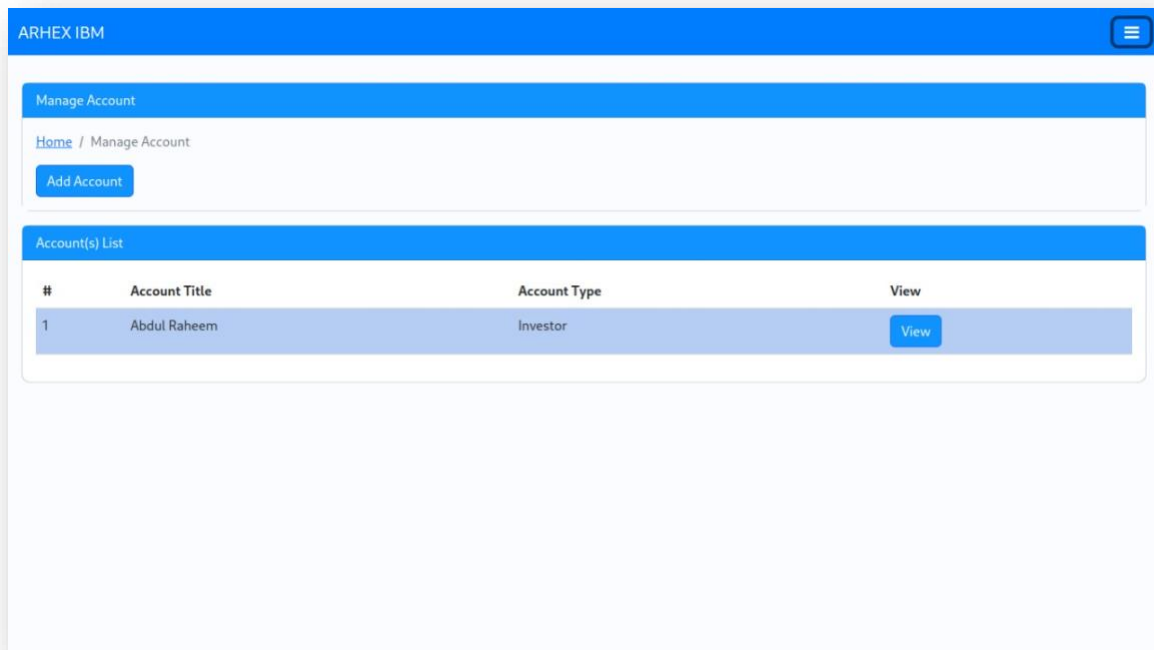



Figure 3(3.6): Manage Account

Account Details



Abdul Raheem

Investor

Update

Deactivate

Account Information

Type	Investor
CNIC	3810361753177
Email Address	Null
Mobile	03134386826
Emergency Contact	Null
Address	House No. 69, Farooq e Azam street, High Court Society, Johar Town, Lahore

Ledgers

Description	Date	Account	Dr	Cr
Capital Ammount added	Dec 23, 2022	Cash	1500	0
Capital Ammount added	Dec 23, 2022	Capital	0	1500
Total: 0				

Figure 4(3.6): Account Details

ARHEX IBM

Manage Capitals

[Home](#) / Manage Capitals

Add Capital

Capitals List

#	Account Title	Amount	Delete
---	---------------	--------	--------

Figure 5(3.6): Manage Capital

ARHEX IBM

Generate Reports

Home / Generate Reports

Generate new report

Report Type

Select Report Type

Start Date

mm/dd/yyyy

End Date

mm/dd/yyyy

Generate

Figure 6(3.6): Generate Report

ARHEX IBM

Manage Payments

Home / Manage Payments

Payment In

Invoice

Select Sale Invoice

Receivable

Receivable Amount

Amount

Enter Amount

Update

Payment Out

Invoice

Select Purchase Invoice

Payable

Payable Amount

Amount

Enter Amount

Update

Figure 7(3.6): Manage Payments

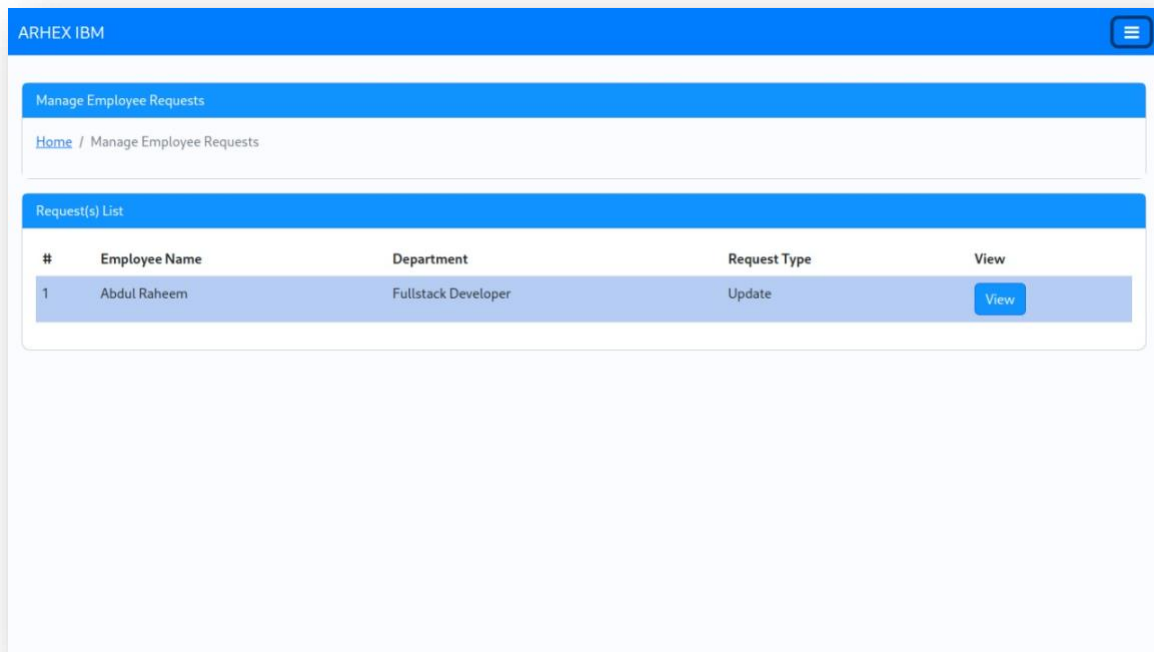


Figure 8(3.6): Manage Employee Request

Employee Request

Name	Abdul Raheem
CNIC	2937482910387
Mobile	03004871213
Salary	140000
Address	Null

[Accept](#) [Reject](#)

Figure 9(3.6): Employee Request

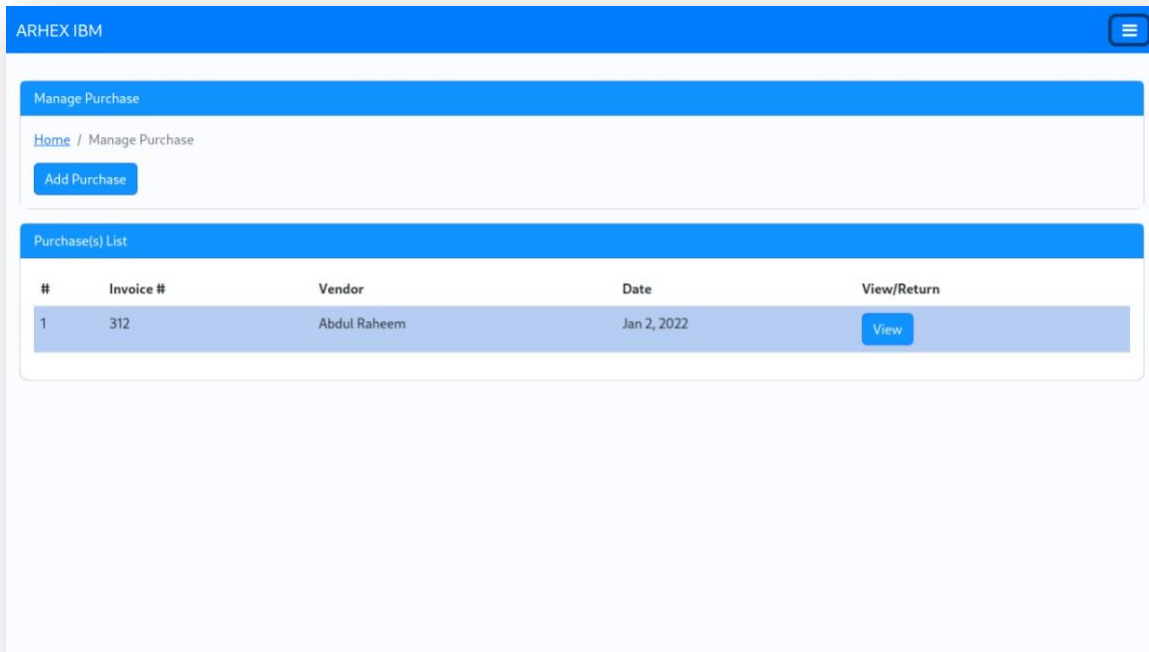


Figure 10(3.6): Manage Purchase

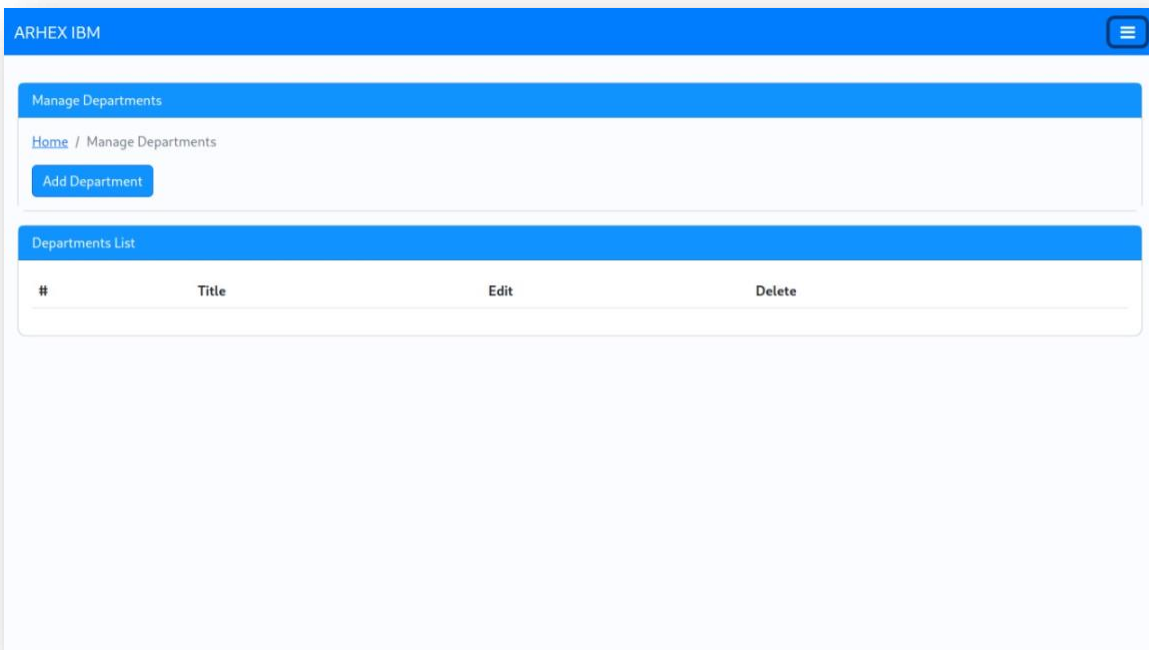


Figure 11(3.6): Manage Department

ARHEX IBM

Manage Backup

[Home](#) / Manage Backup

Create Backup

Date

Today Data

Backup

Restore Backup

Restore File

Choose File No file chosen

Restore

Figure 13(3.6): Manage Backup

ARHEX IBM

Manage Attendance

[Home](#) / Manage Attendance

Search By Date

Date

mm/dd/yyyy

Search

Mark Attendance

Employee ID

Enter Employee ID

Check In Check Out

Attendance

#	ID	Check In	Check Out
---	----	----------	-----------

Figure 12(3.6): Manage Attendance

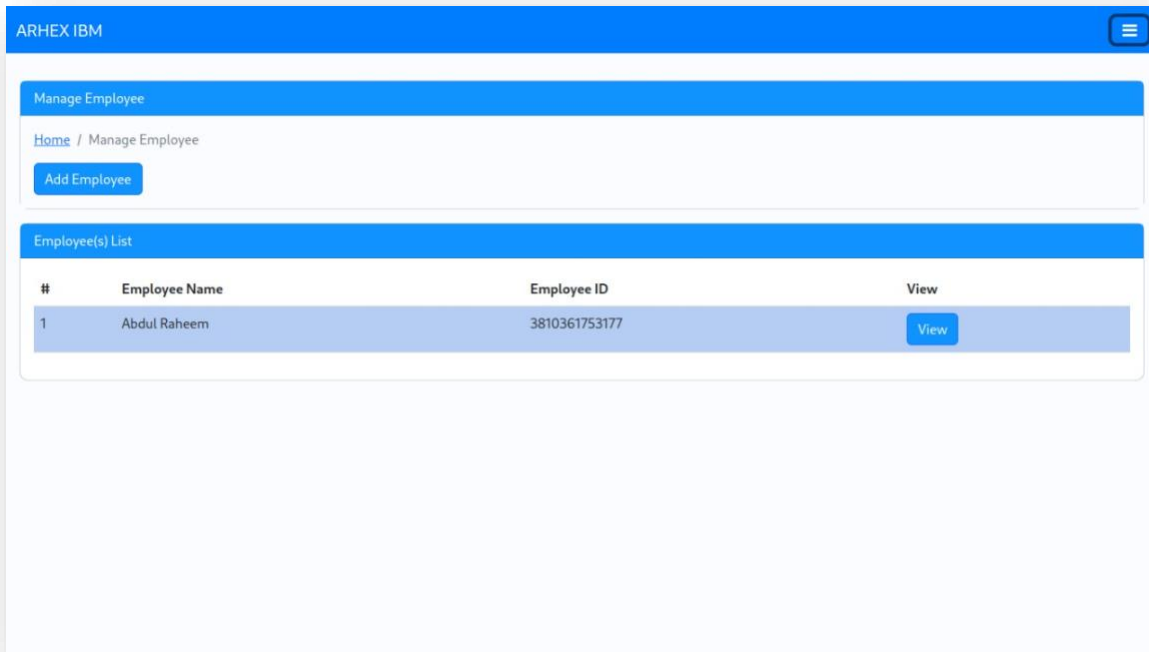


Figure 14(3.6): Manage Employee

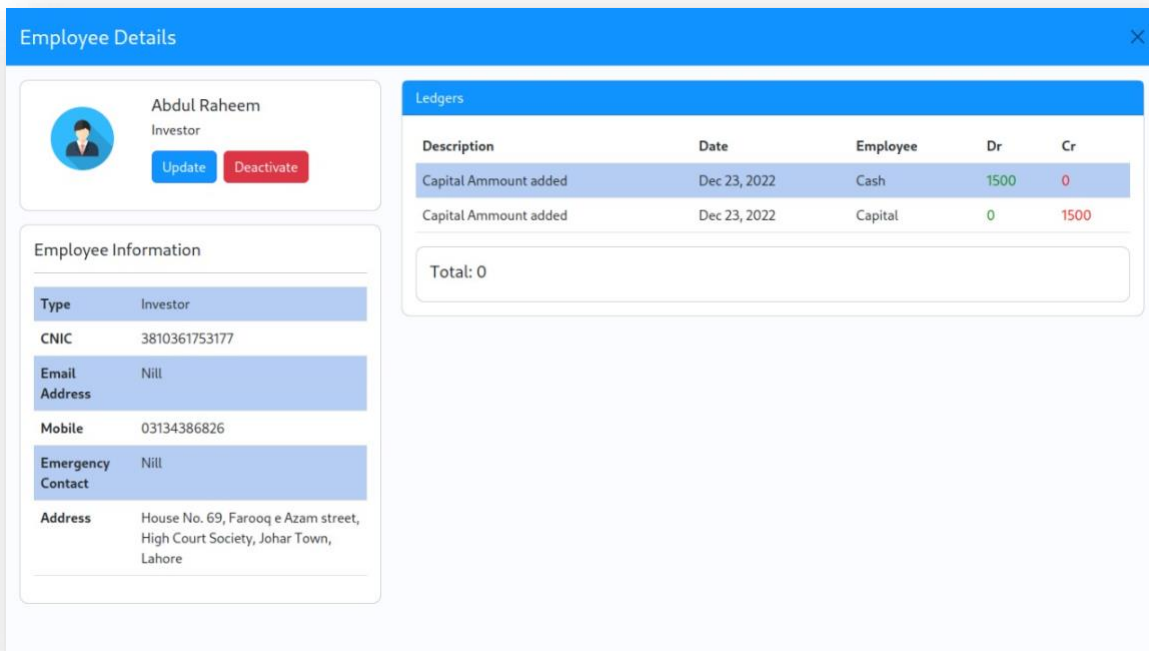


Figure 15(3.6): Employee Details

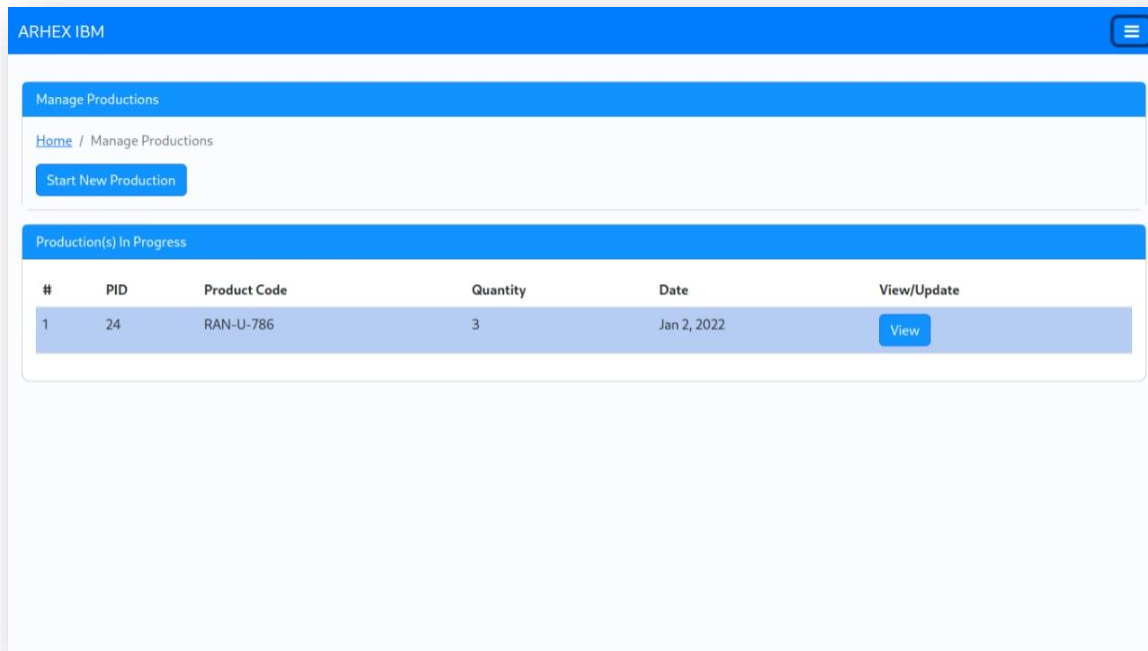


Figure 16(3.6): Manage Production

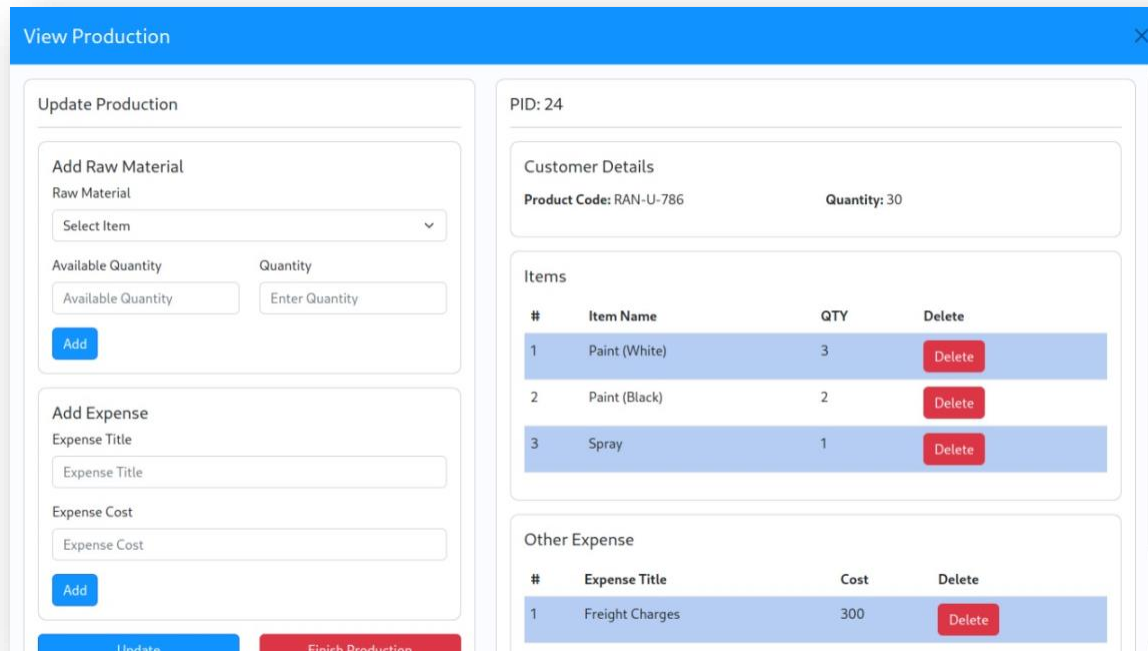


Figure 17(3.6): View Production

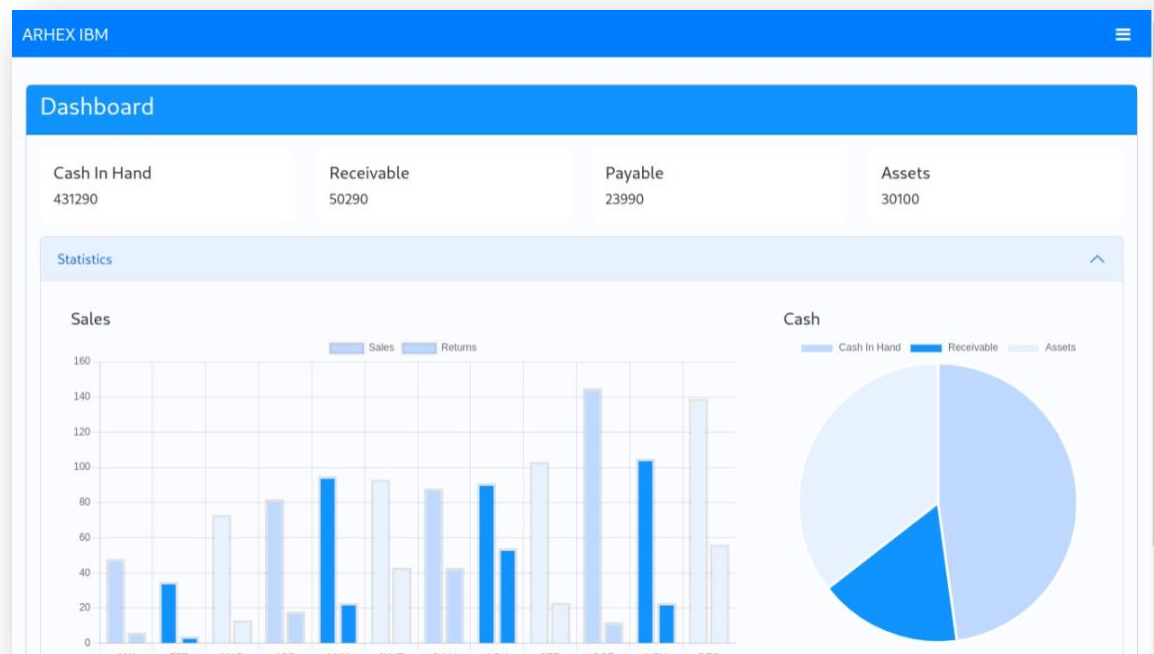


Figure 18(3.6): Dashboard

4. DATA FLOW DIAGRAM (OPTIONAL)

4.1 Data Flow Diagram Level 0

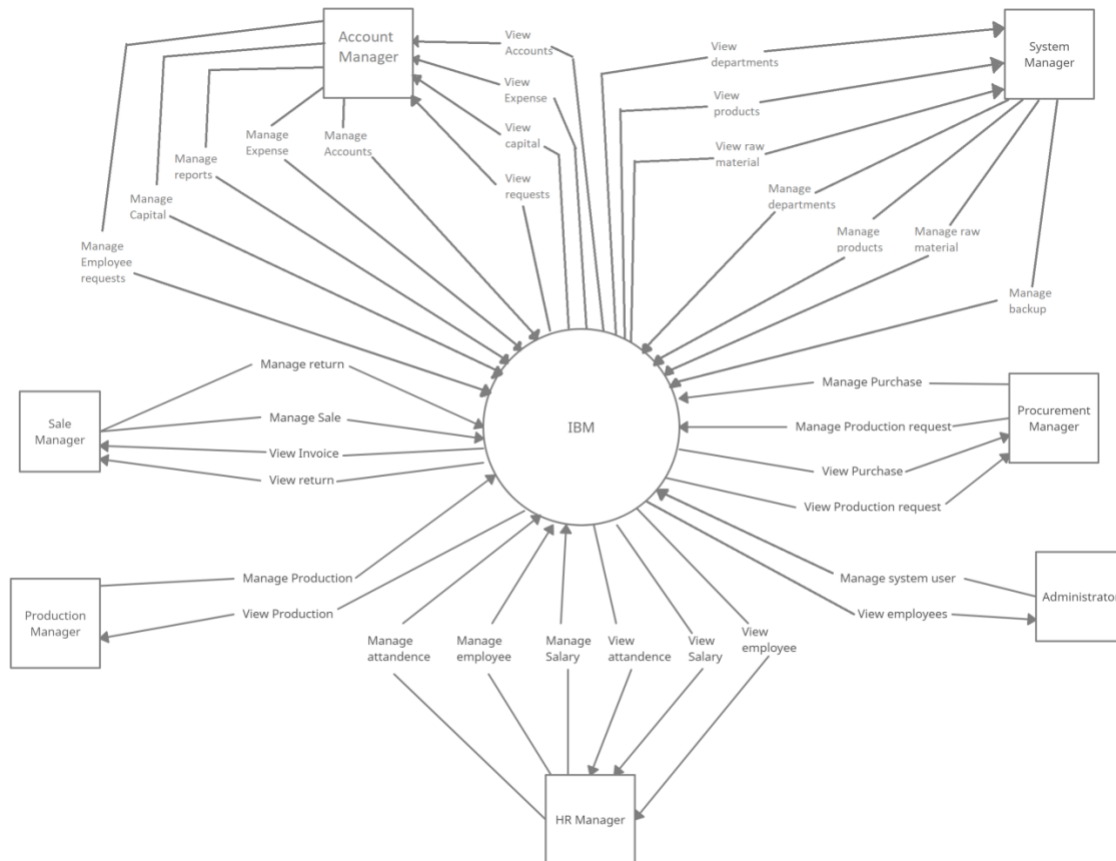


Figure 19(4.1): DFD Level 0 Diagram

4.2 Data Flow Diagram Level 1



Figure 20(4.2): DFD Level 1 Diagram

5. SYSTEM DESIGN

5.1 System Architecture Diagram

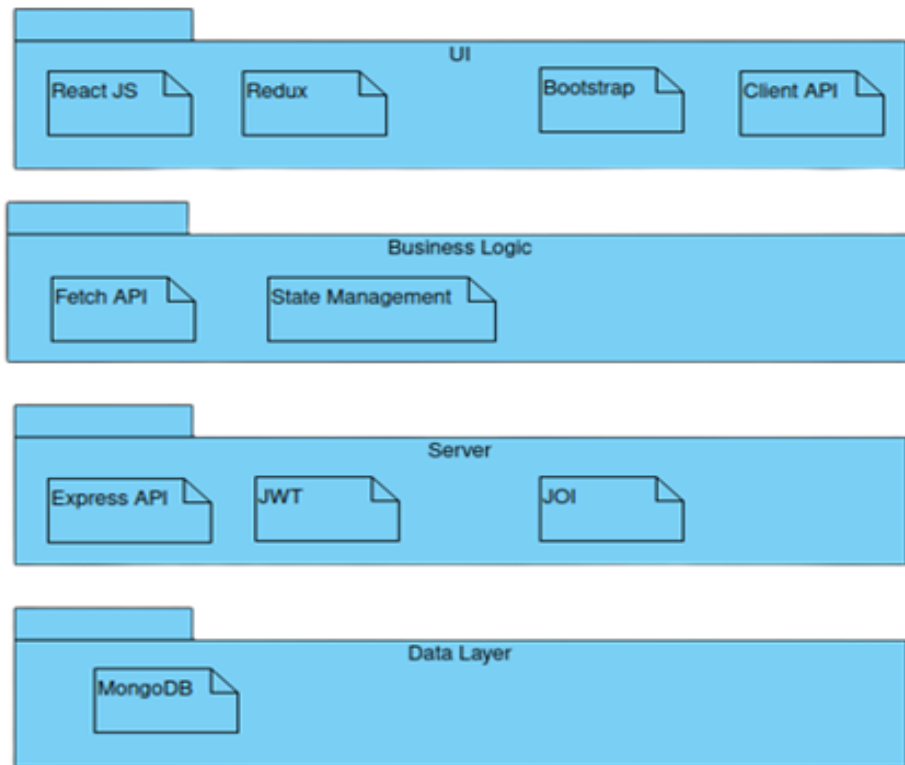


Figure 21(5.1): System Architecture

5.2 Class Diagram

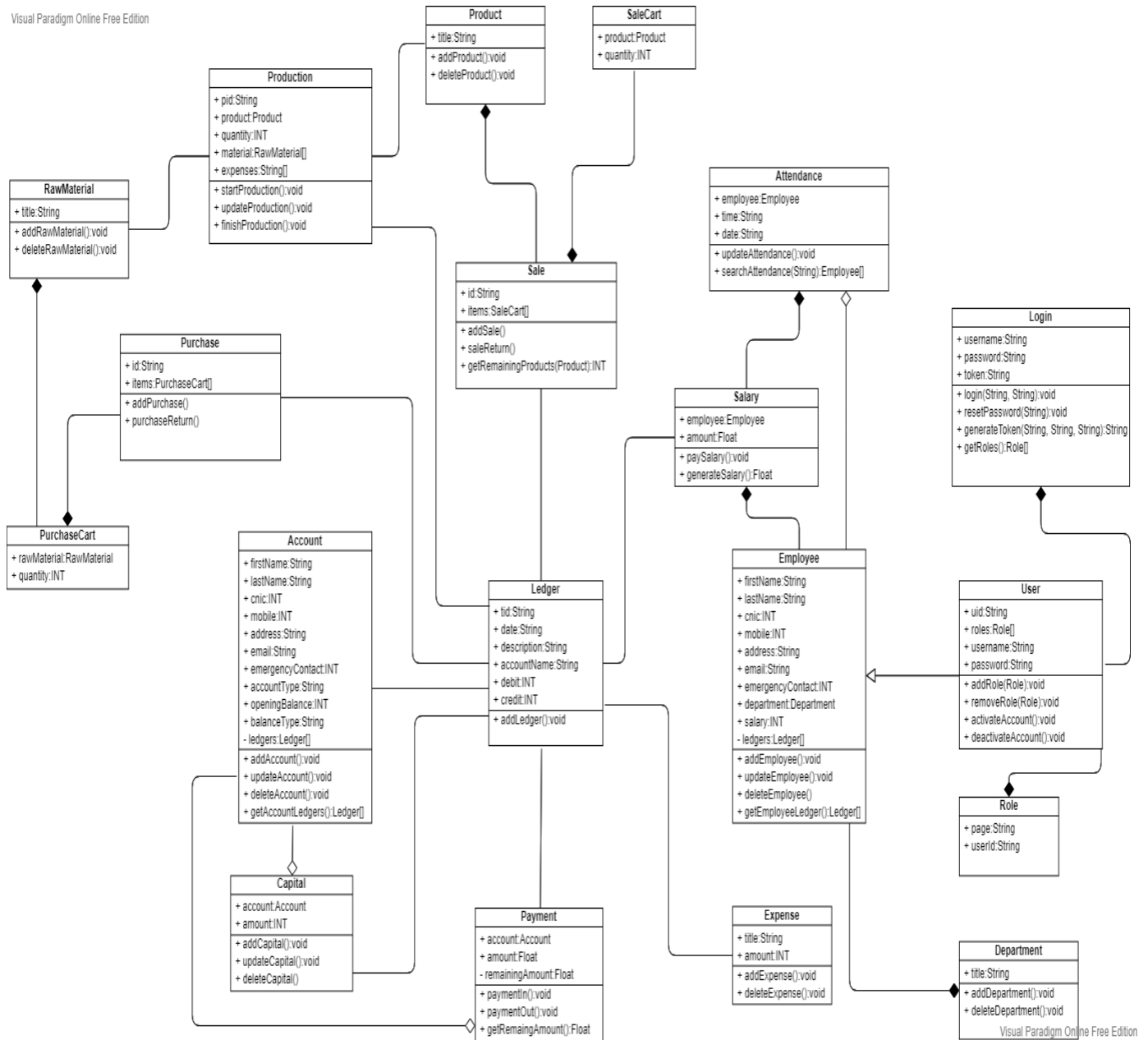


Figure 22(5.2): Class Diagram

5.3 Sequence Diagrams

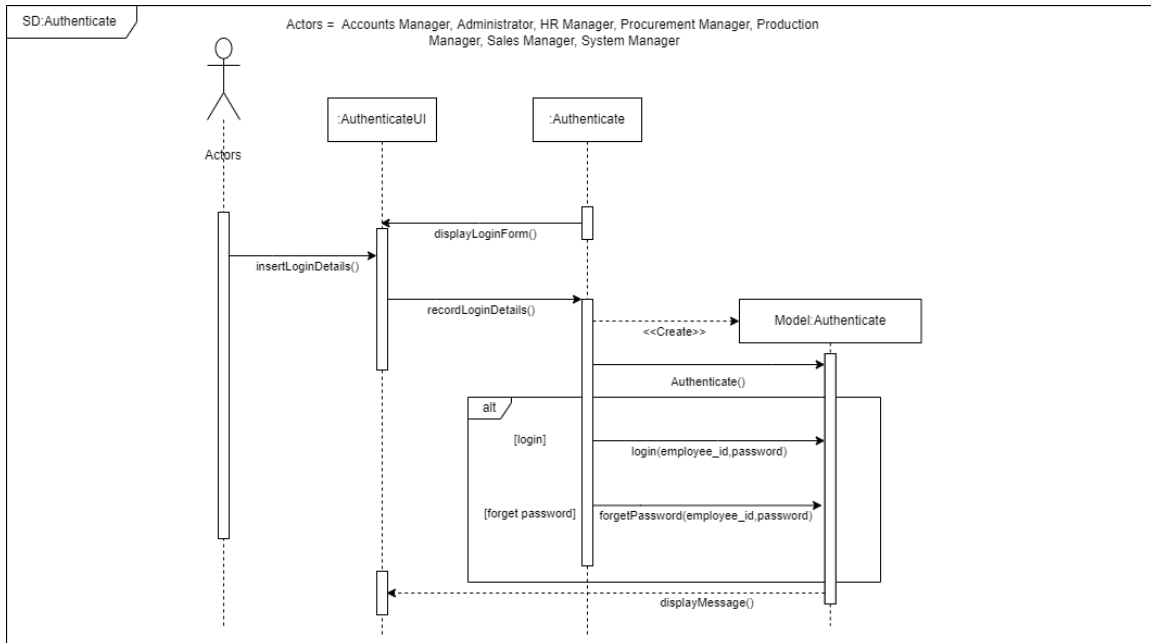


Figure 23(5.3): Authenticate User

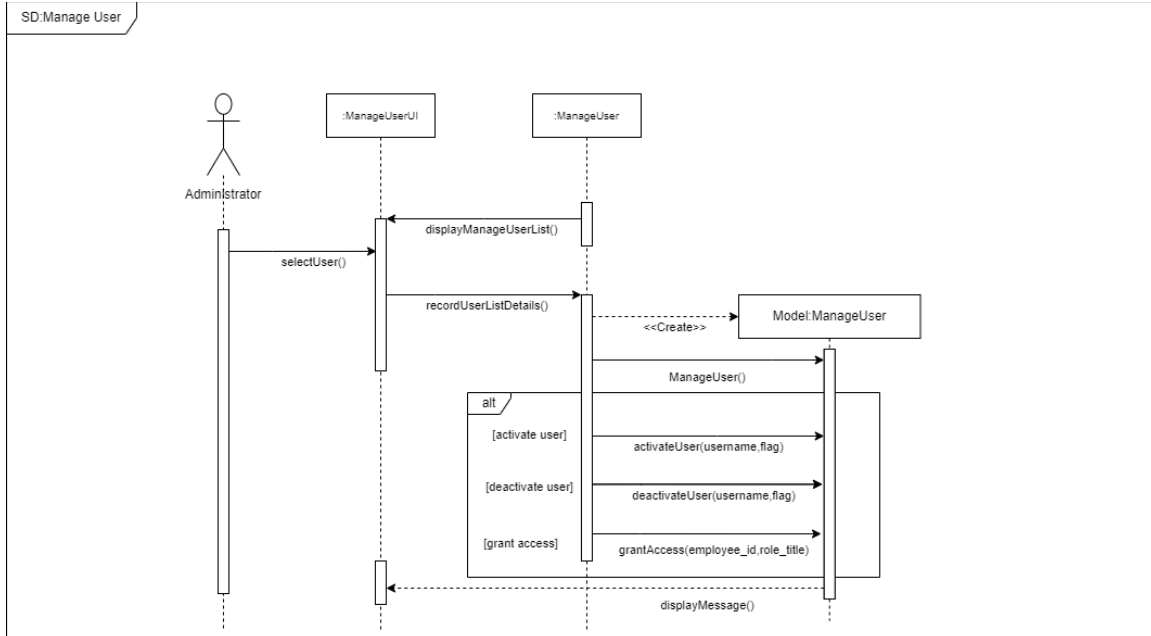


Figure 24(5.3): Manage User

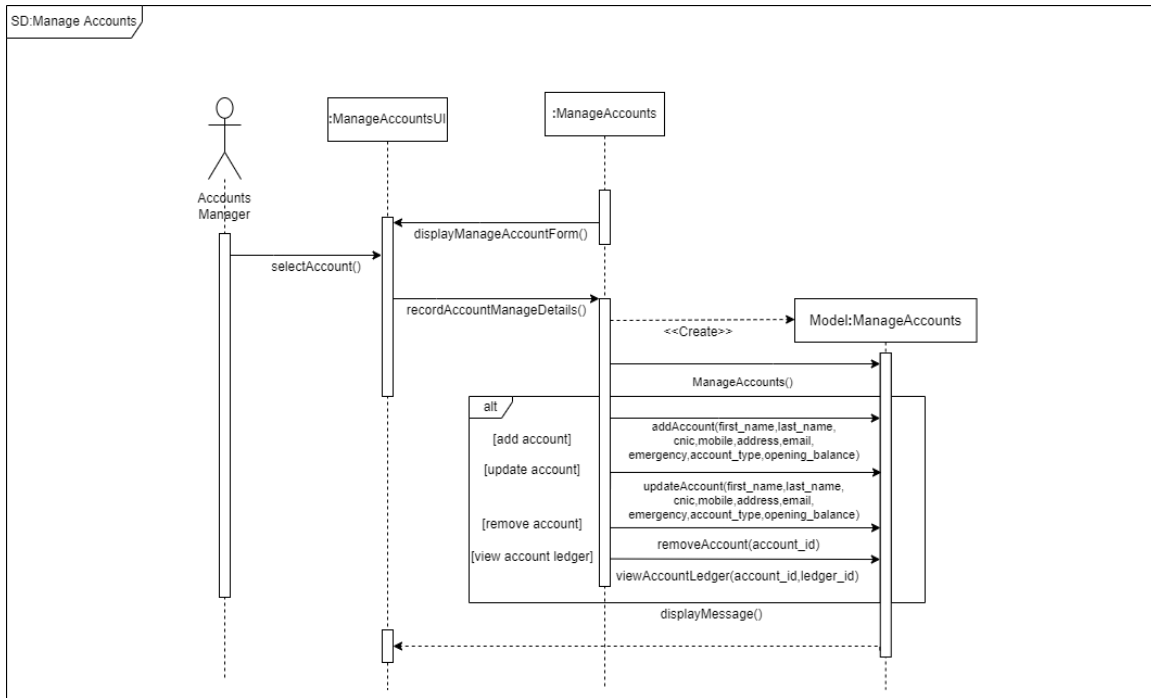


Figure 25(5.3): Manage Accounts

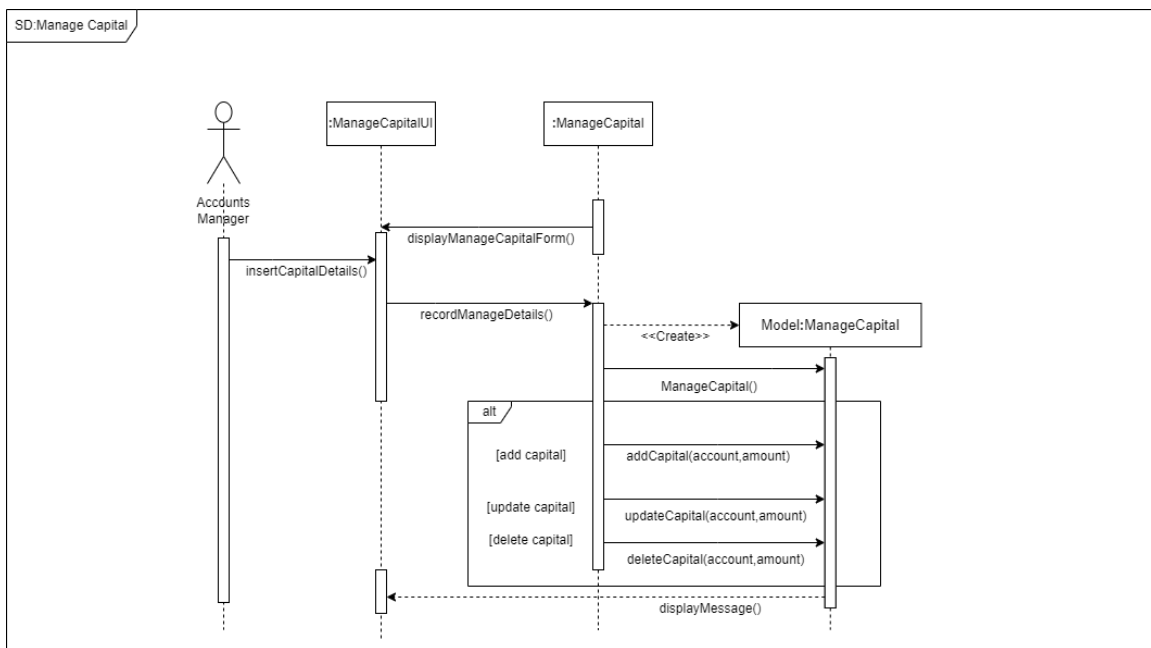


Figure 26(5.3): Manage Capital

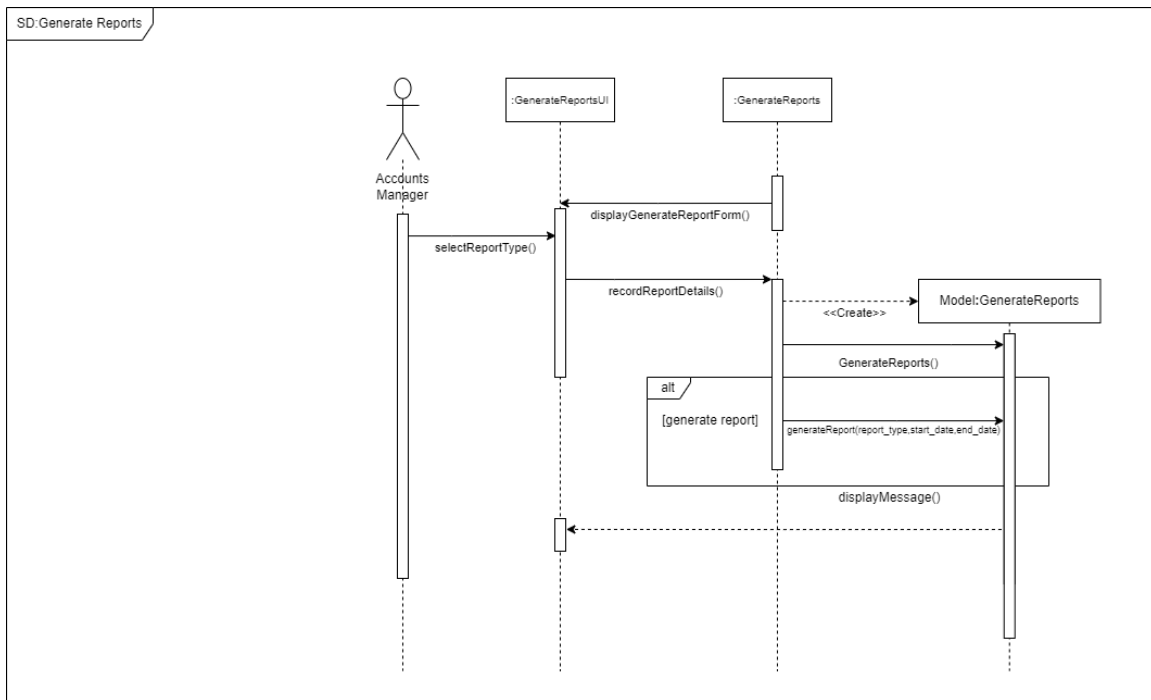


Figure 27(5.3): Generate Report

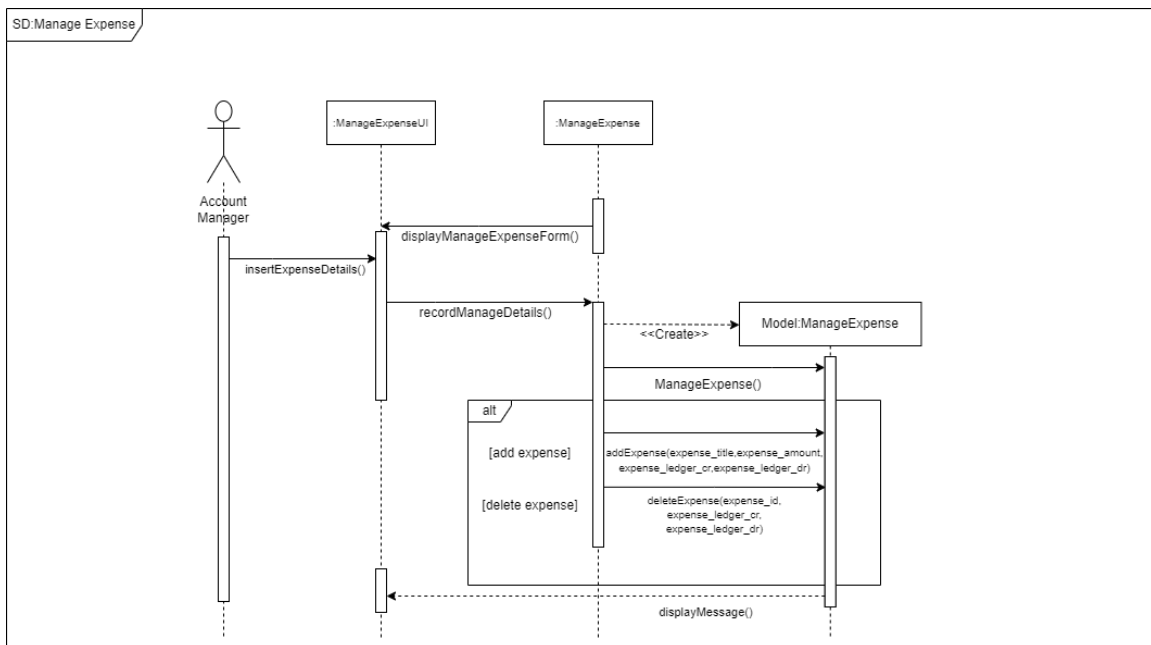


Figure 28(5.3): Manage Expense

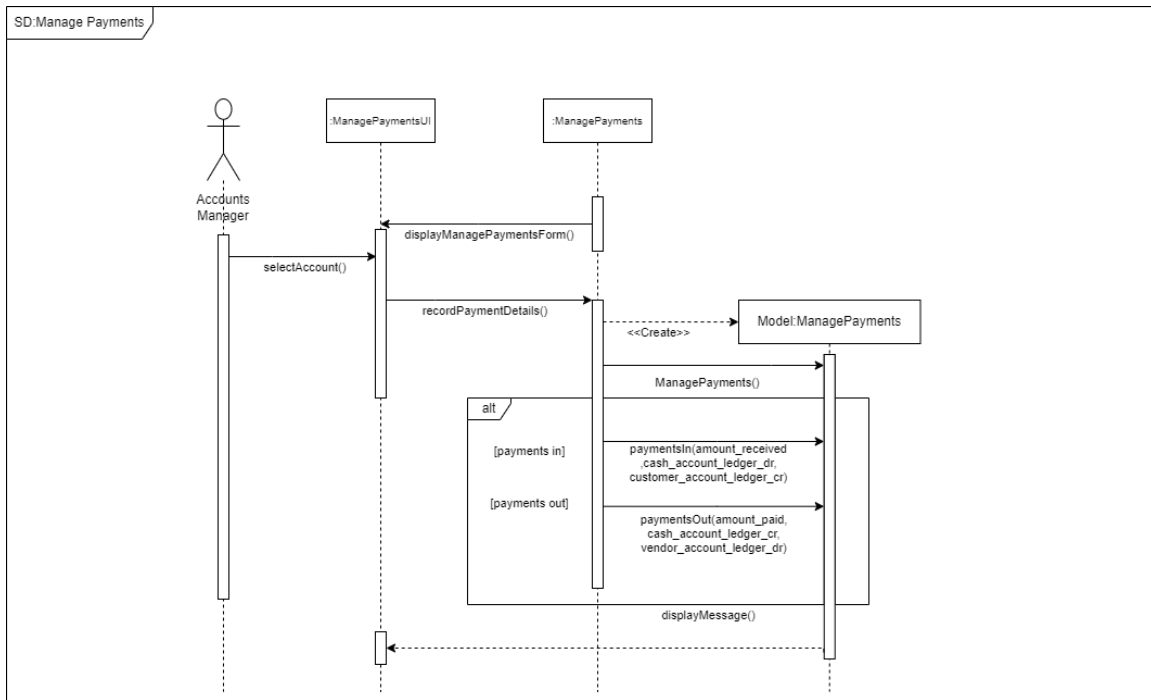


Figure 29(5.3): Manage Payments

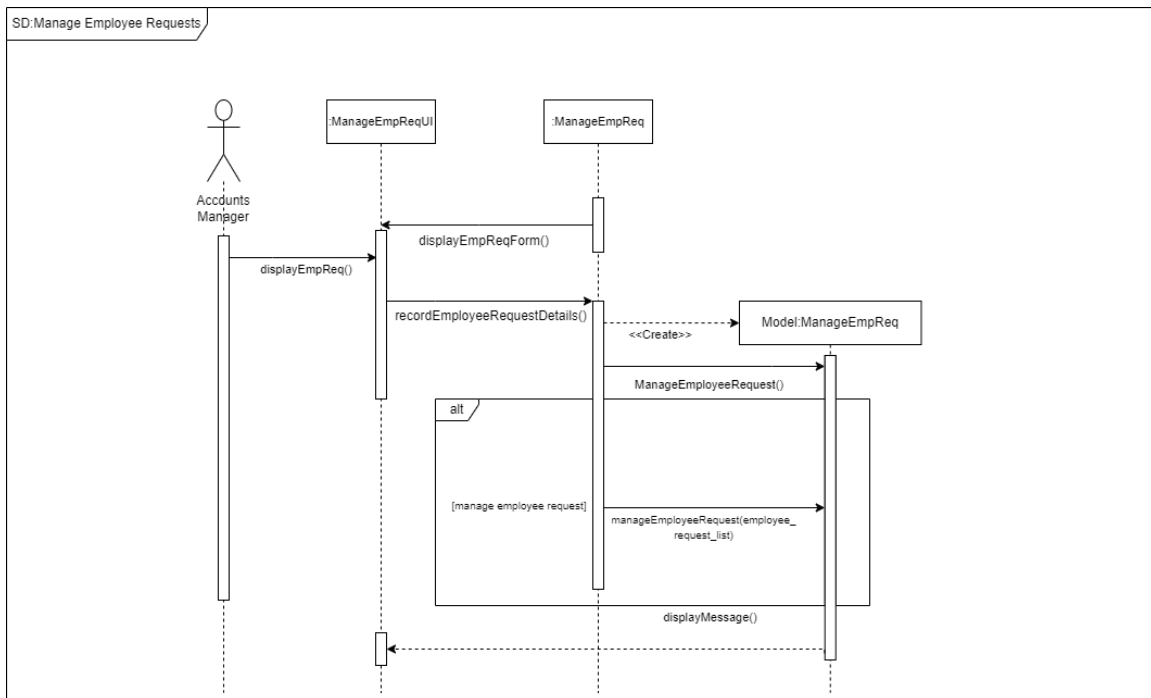


Figure 30(5.3): Manage Employee Requests

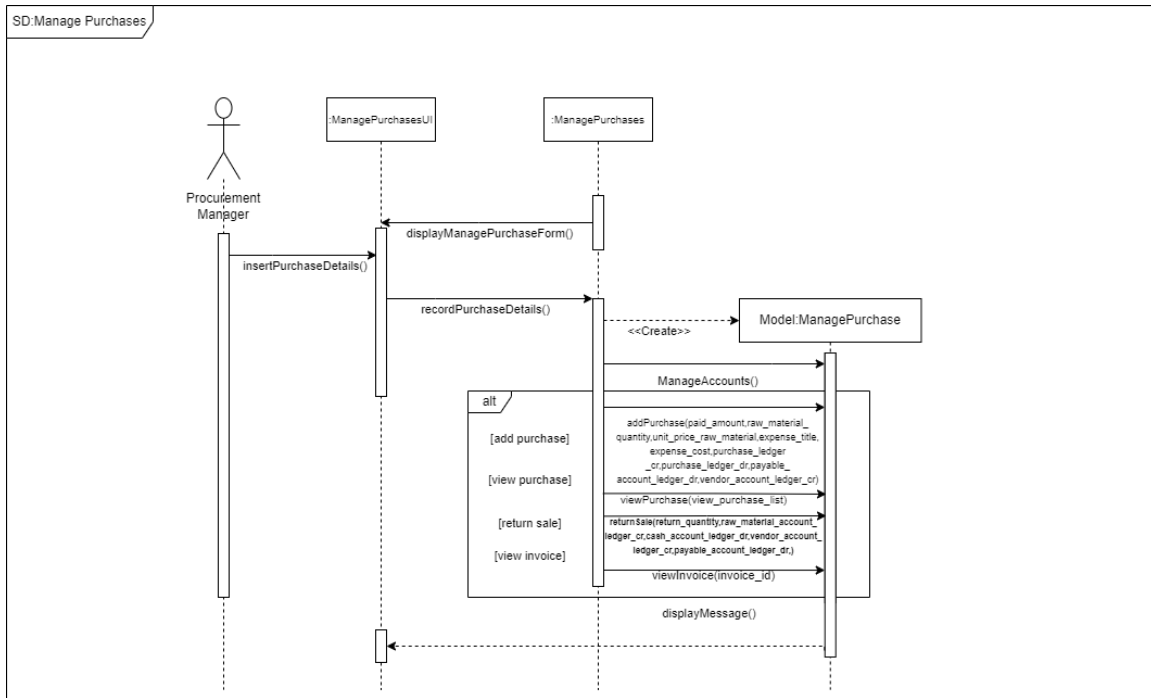


Figure 31(5.3): Manage Purchases

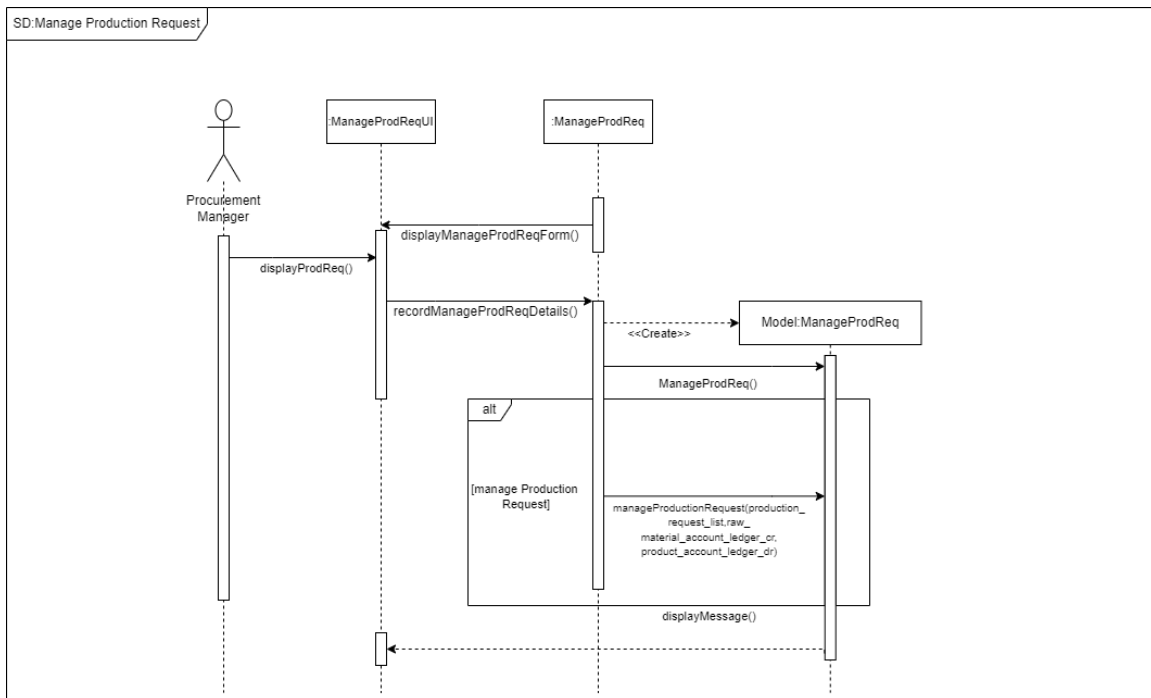


Figure 32(5.3): Manage Production Requests

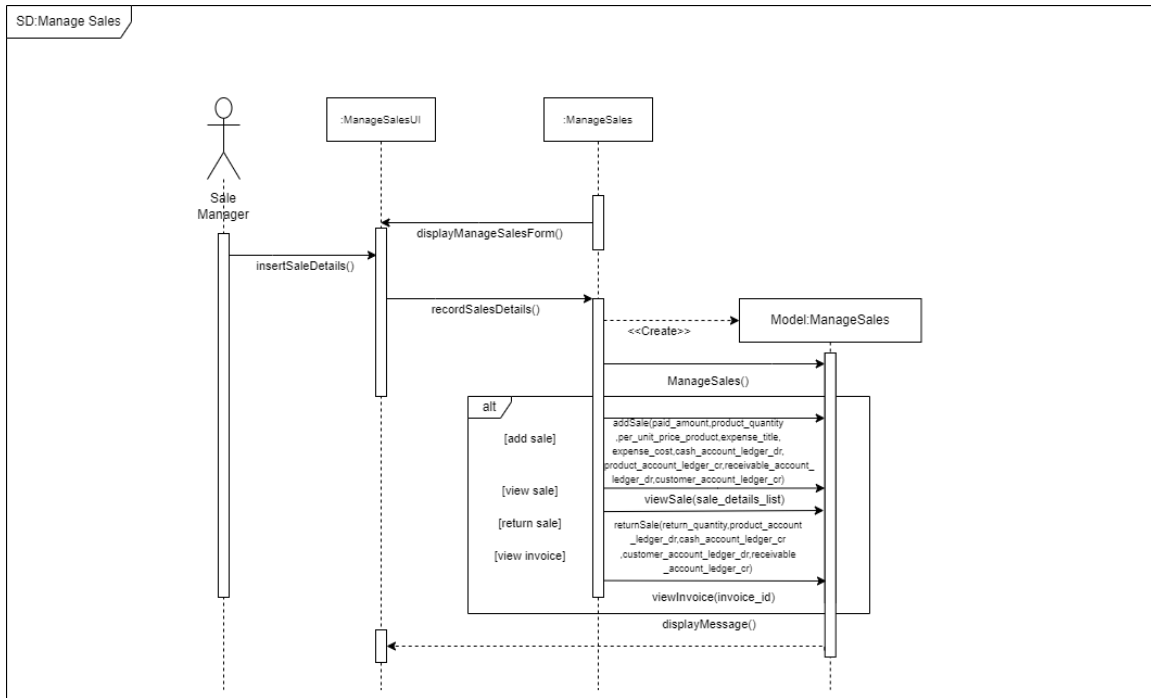


Figure 33(5.3): Manage Sales

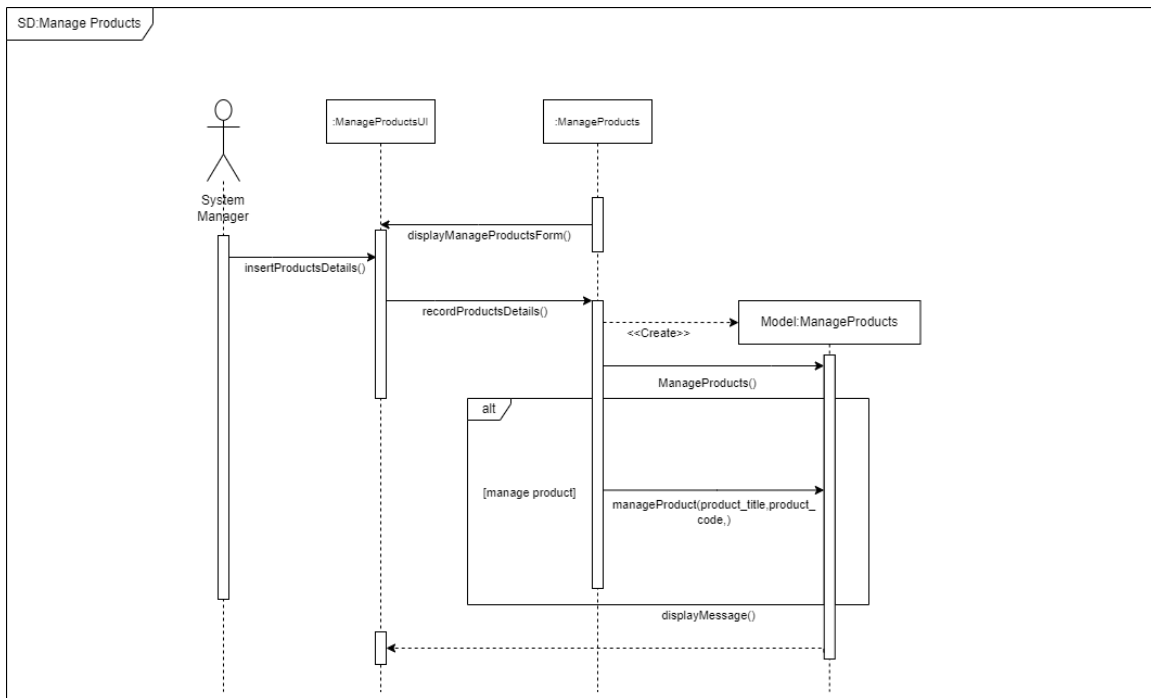


Figure 34(5.3): Manage Products

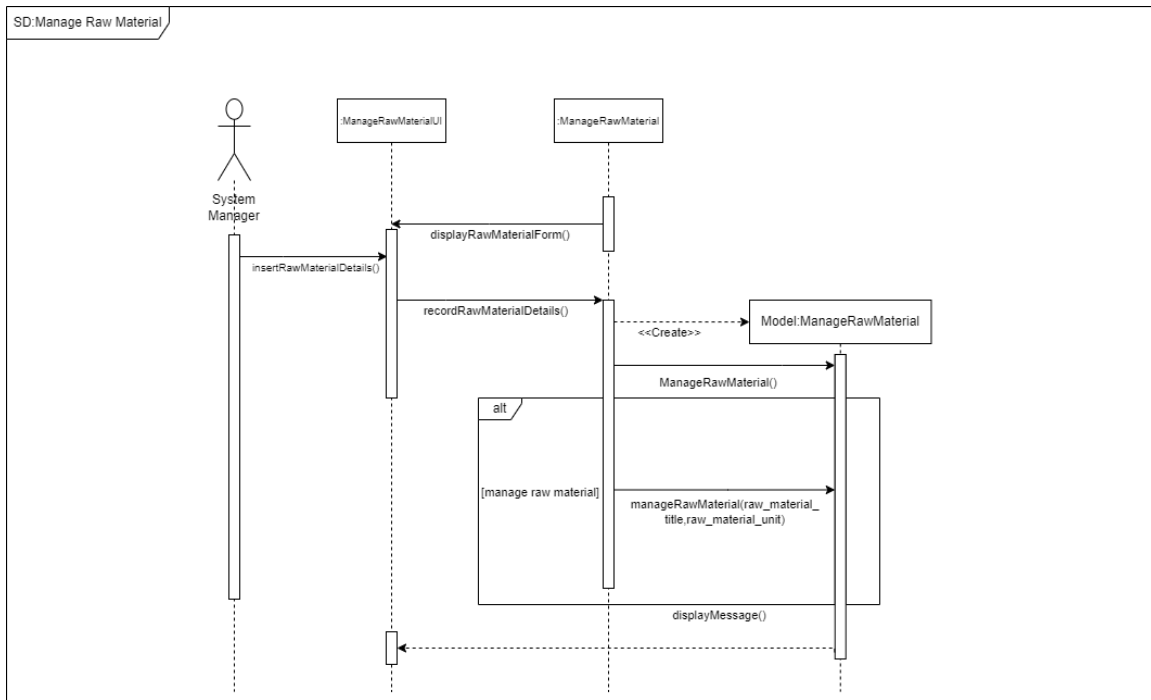


Figure 35(5.3): Manage Raw Material

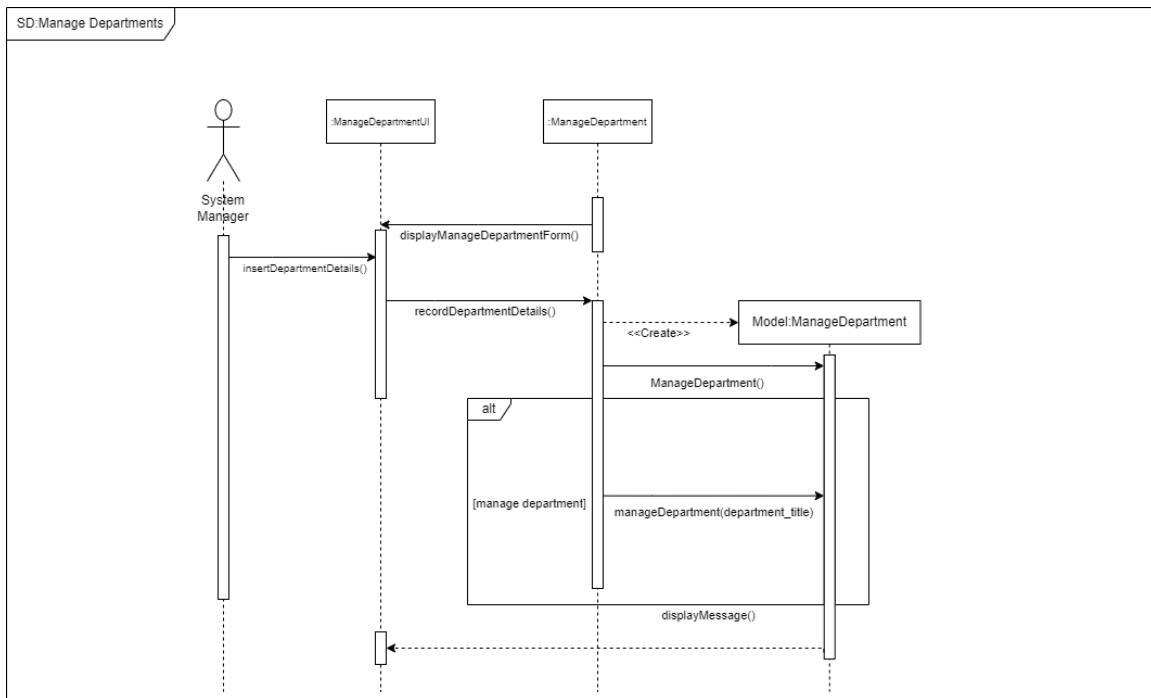


Figure 36(5.3): Manage Departments

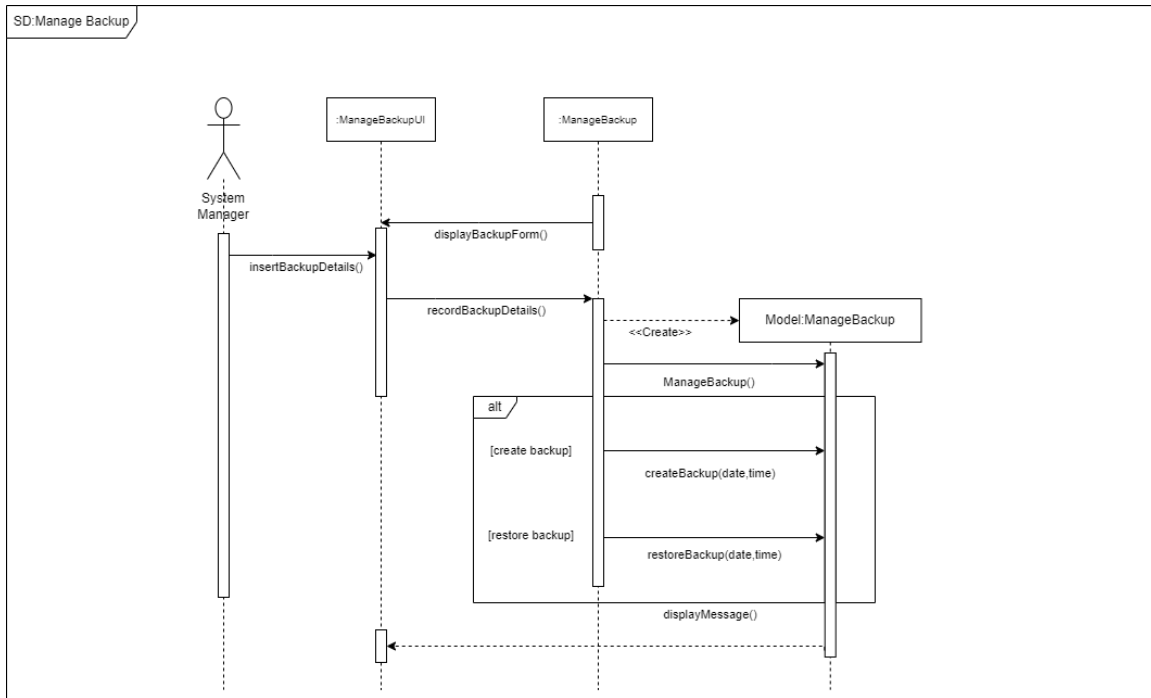


Figure 37(5.3): Manage Backup

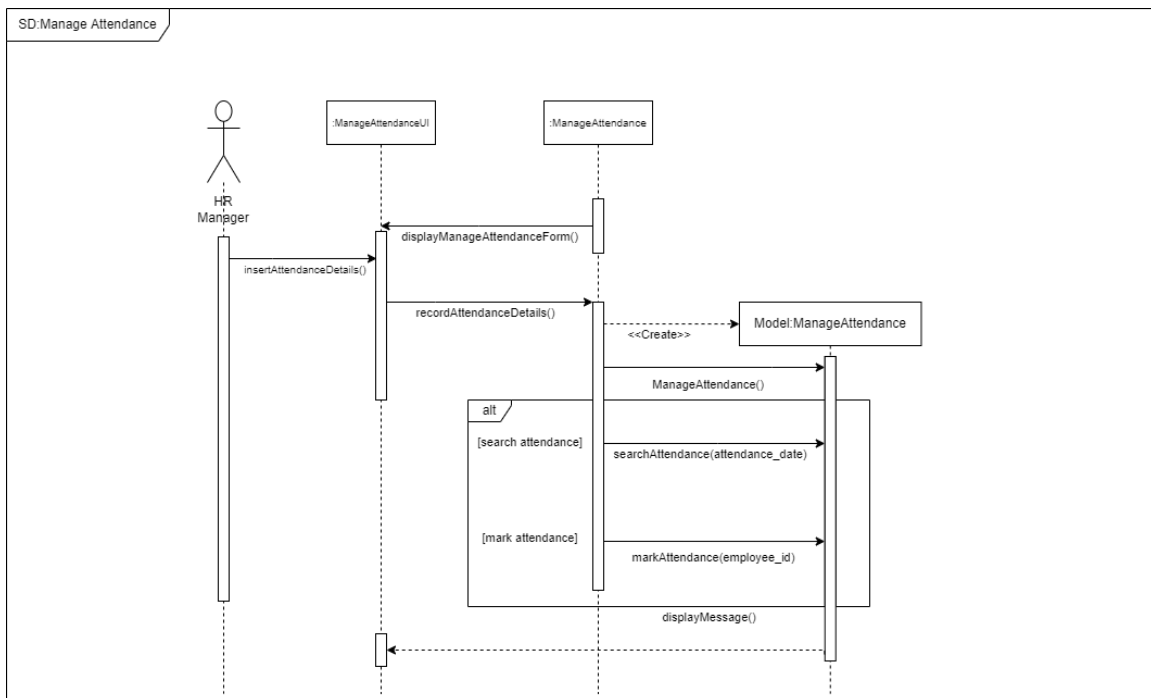


Figure 38(5.3): Manage Attendance

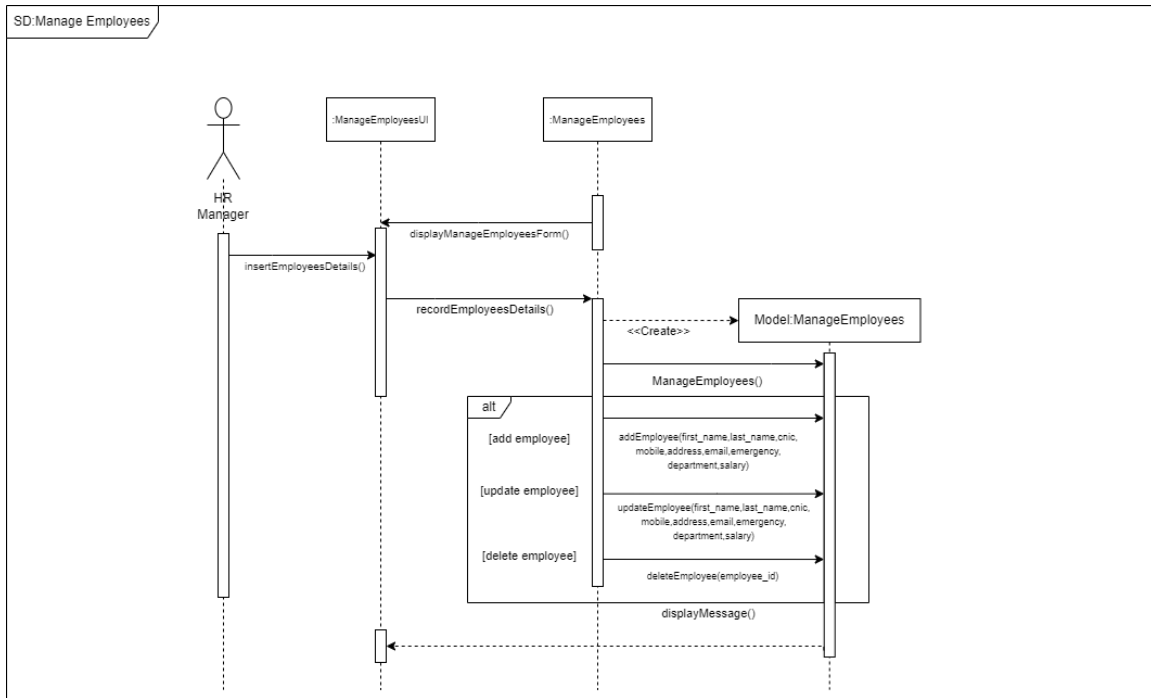


Figure 39(5.3): Manage Employees

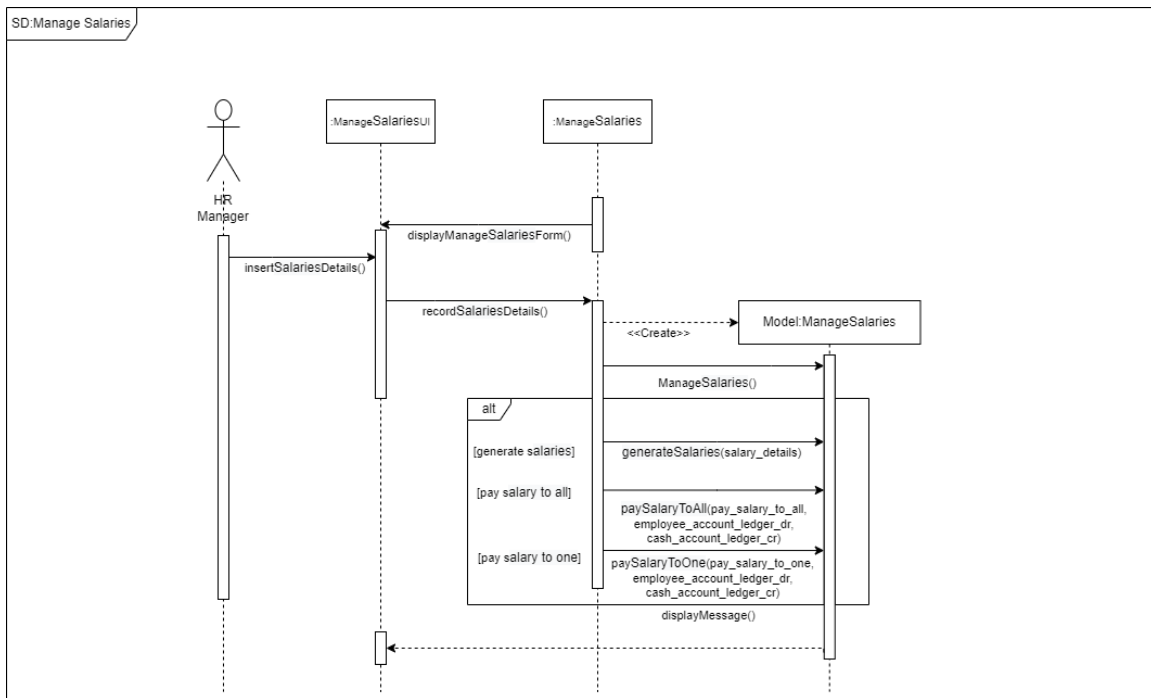


Figure 40(5.3): Manage Salaries

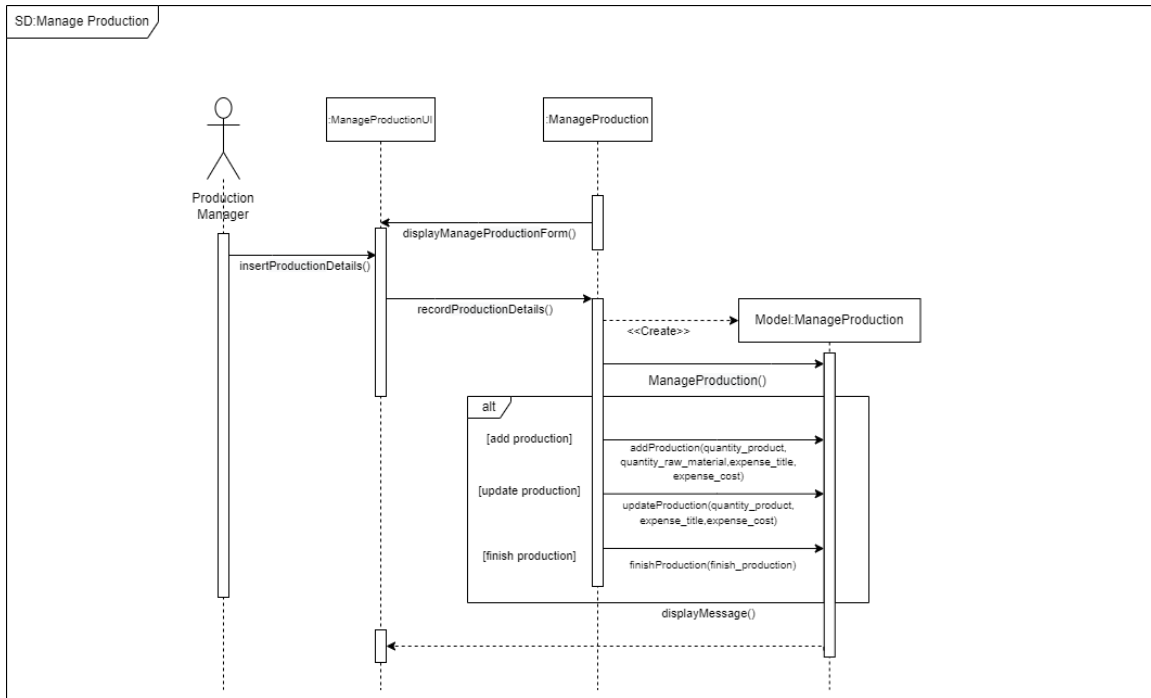


Figure 41(5.3): Manage Production

5.4 Collaboration Diagrams

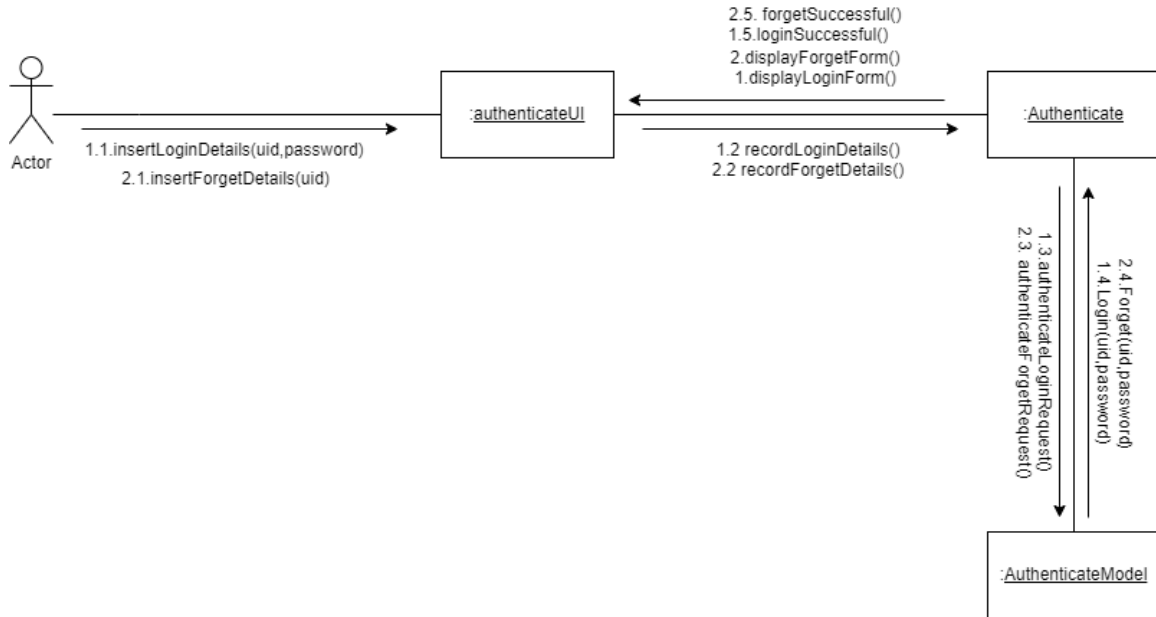


Figure 42(5.4): Authenticate User

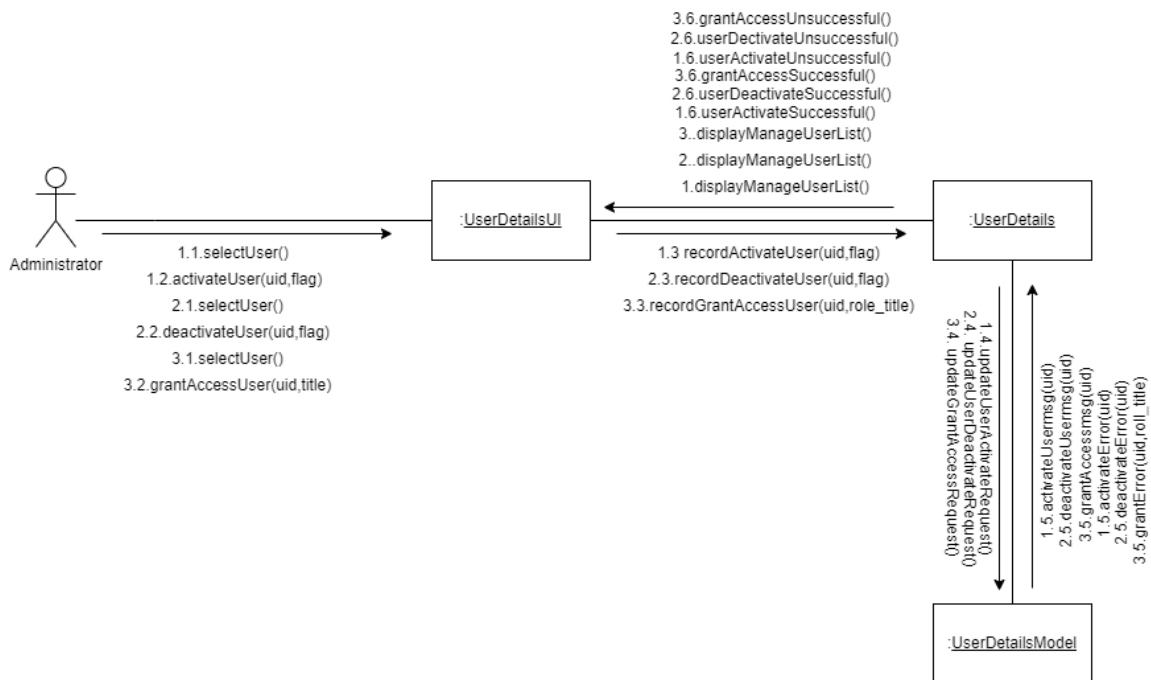


Figure 43(5.4): Manage User

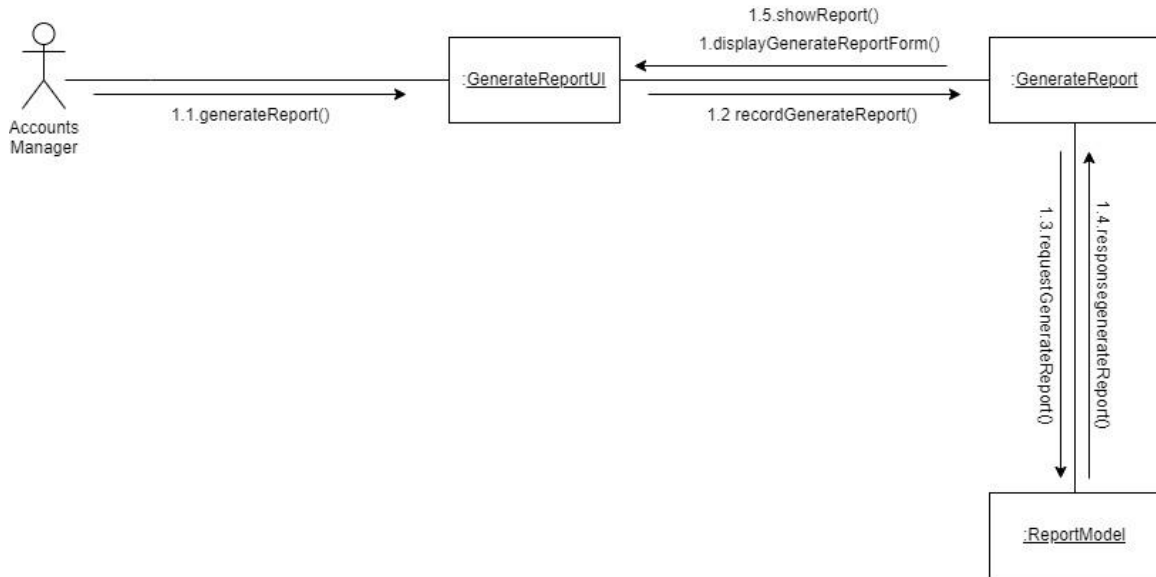


Figure 46(5.4): Generate Report

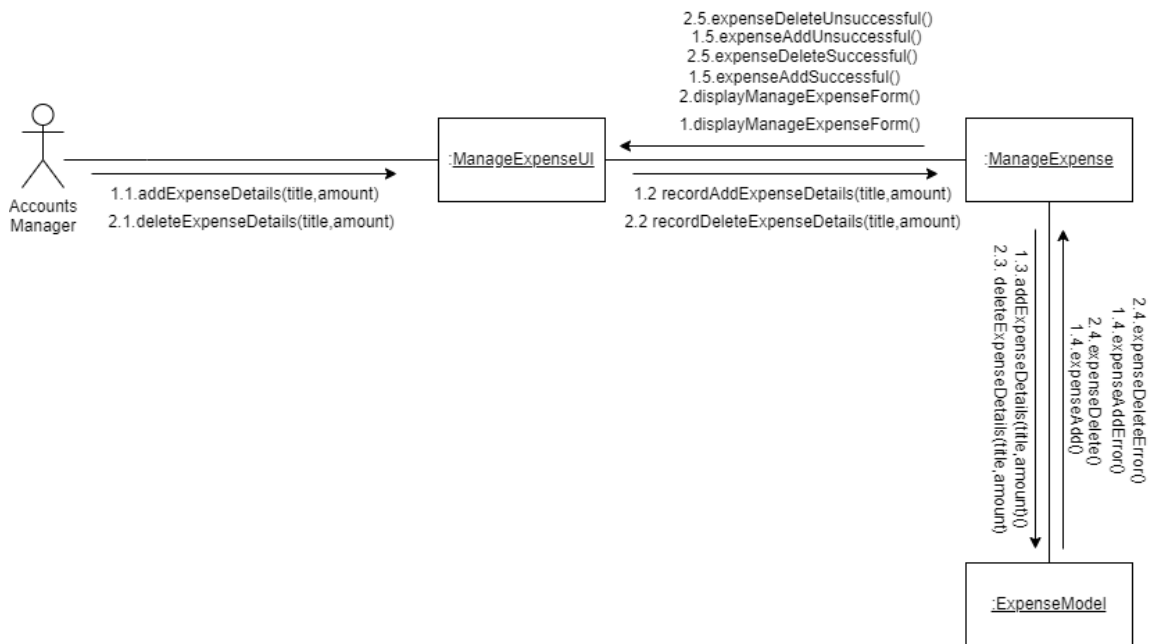


Figure 47(5.4): Manage Expense

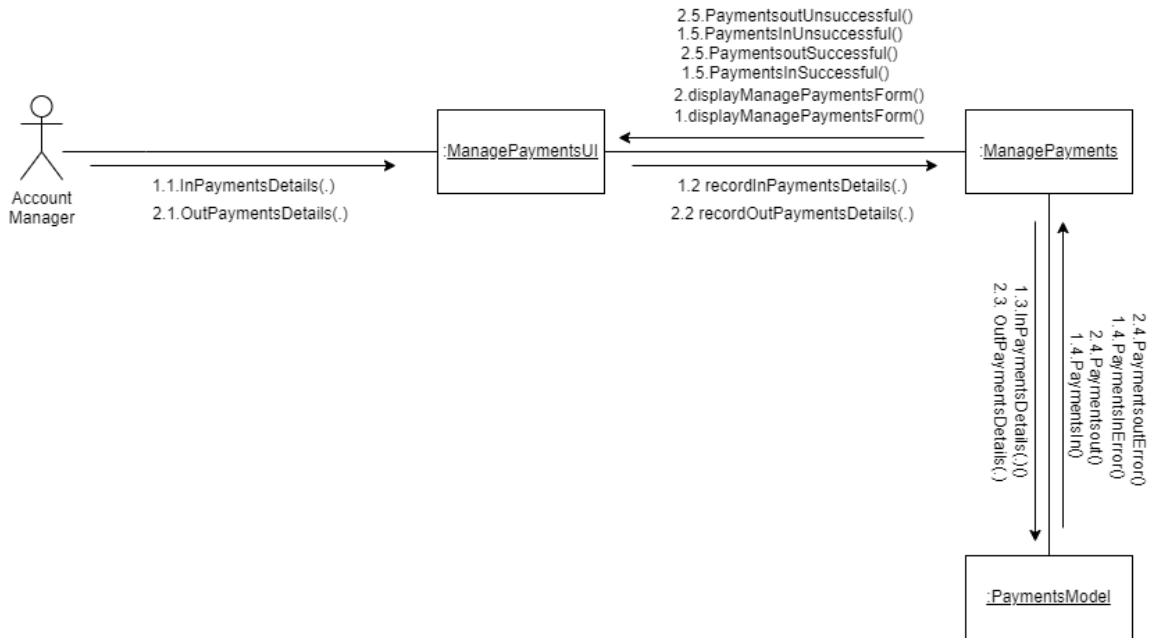


Figure 48(5.4): Manage Payments

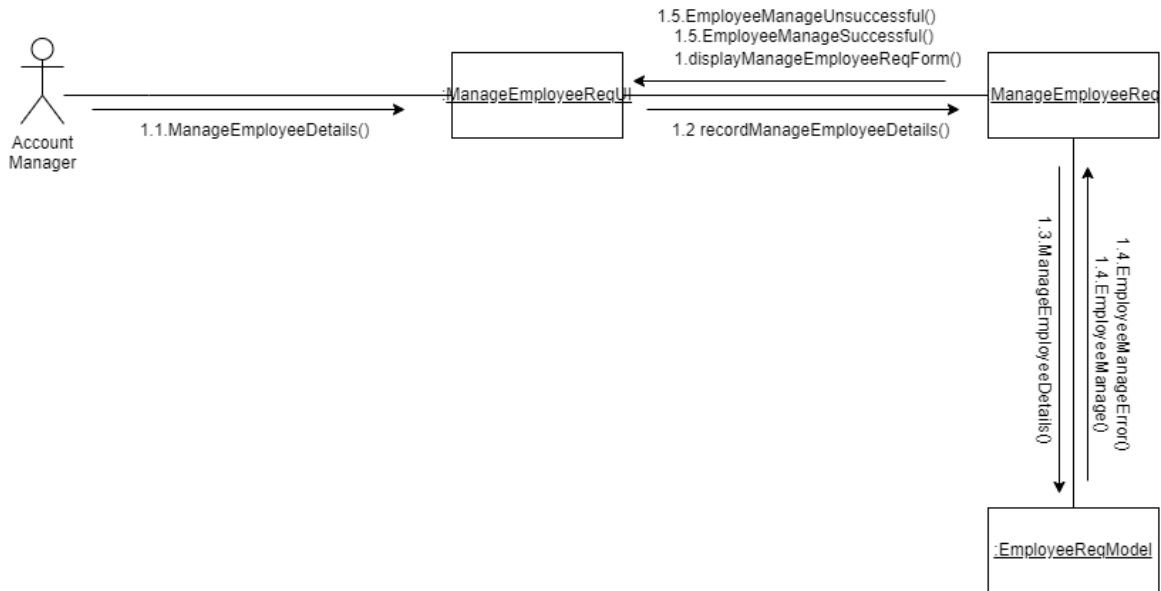


Figure 49(5.4): Manage Employee Request

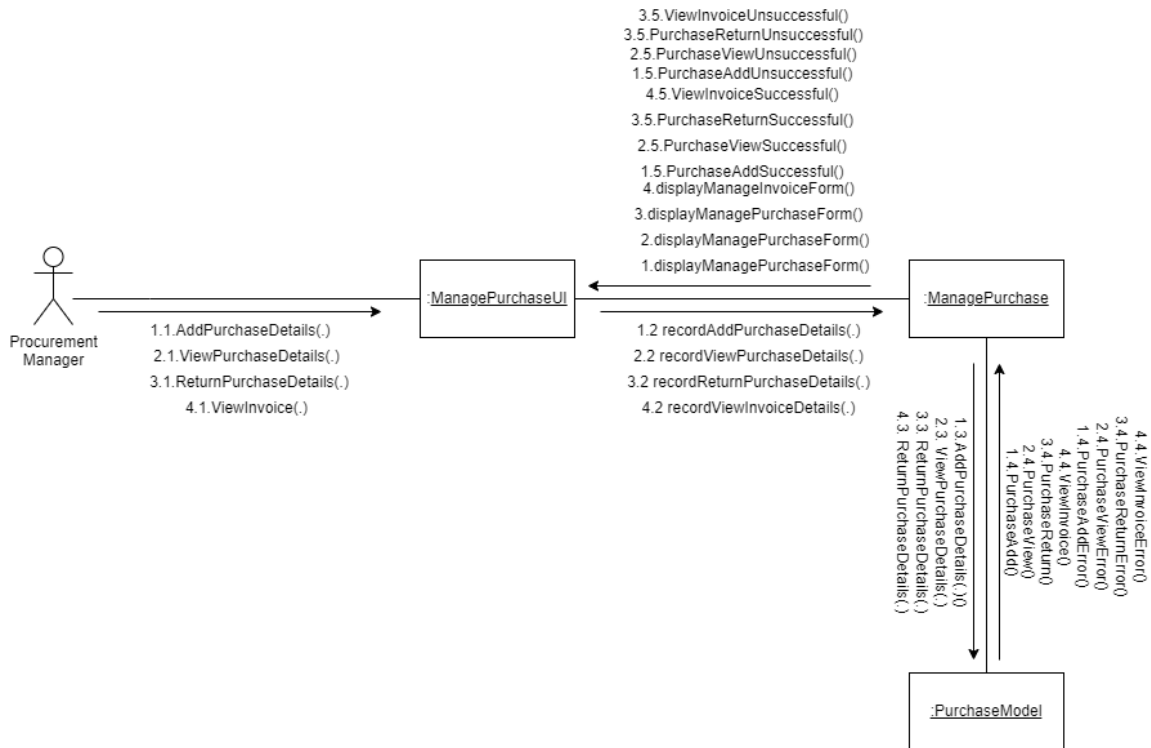


Figure 50(5.4): Manage Purchases

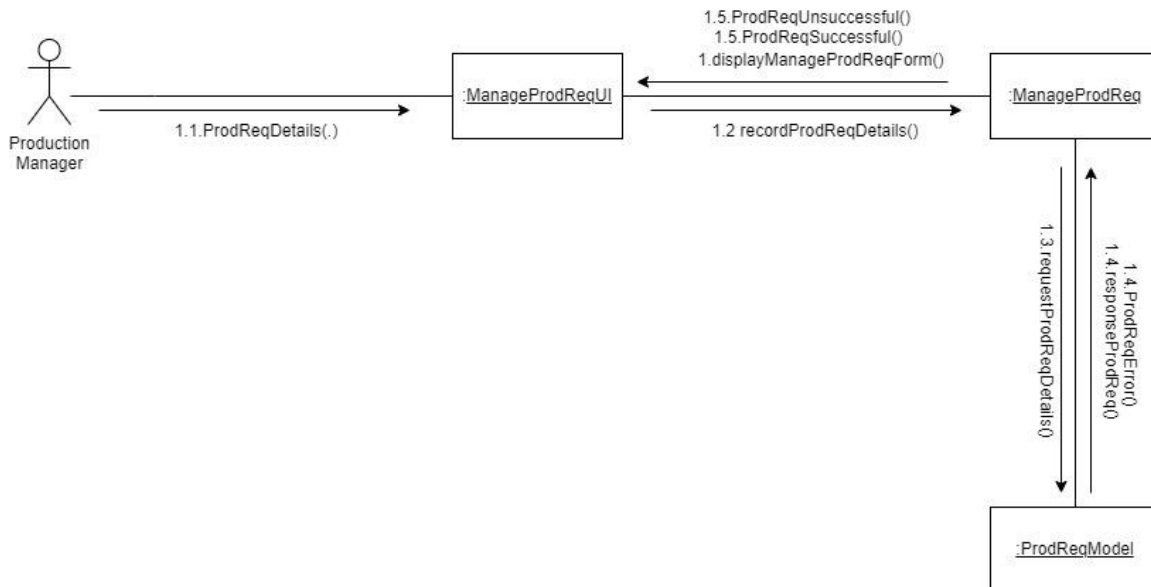


Figure 51(5.4): Manage Production Request

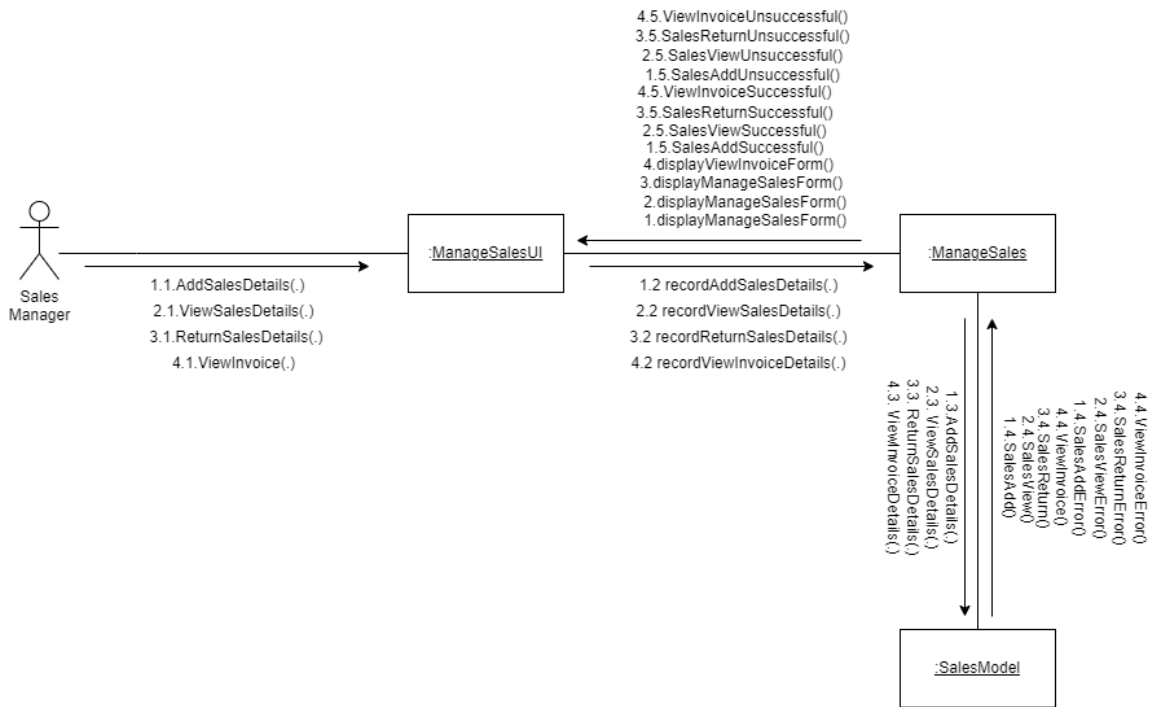


Figure 52(5.4): Manage Sales

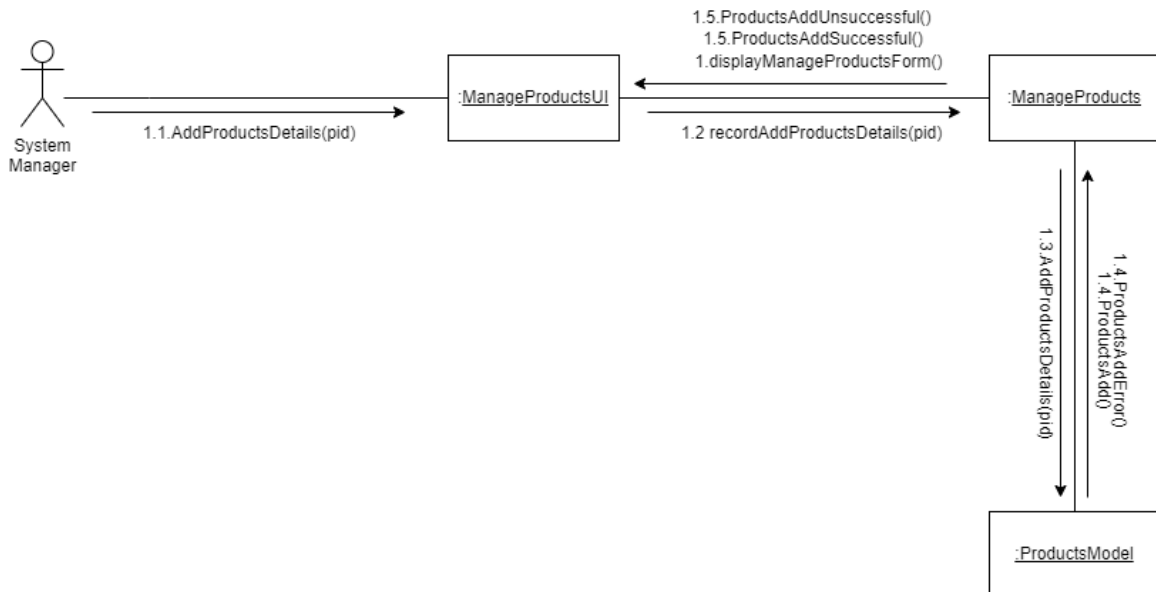


Figure 53(5.4): Manage Products

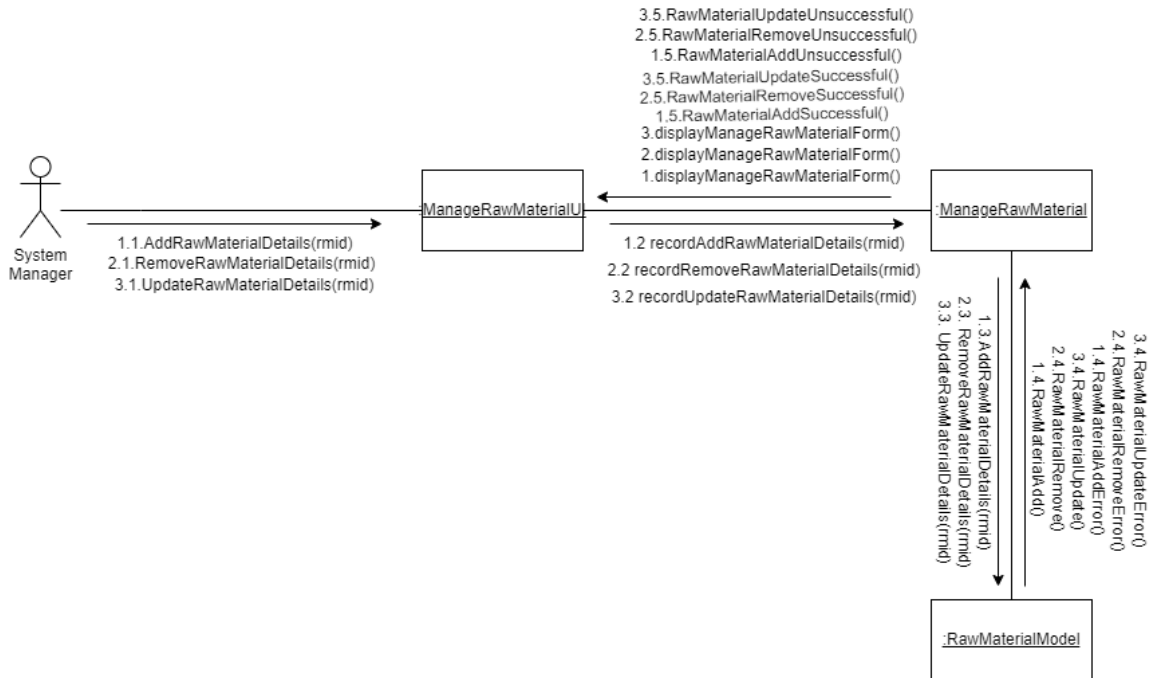


Figure 54(5.4): Manage Raw Material

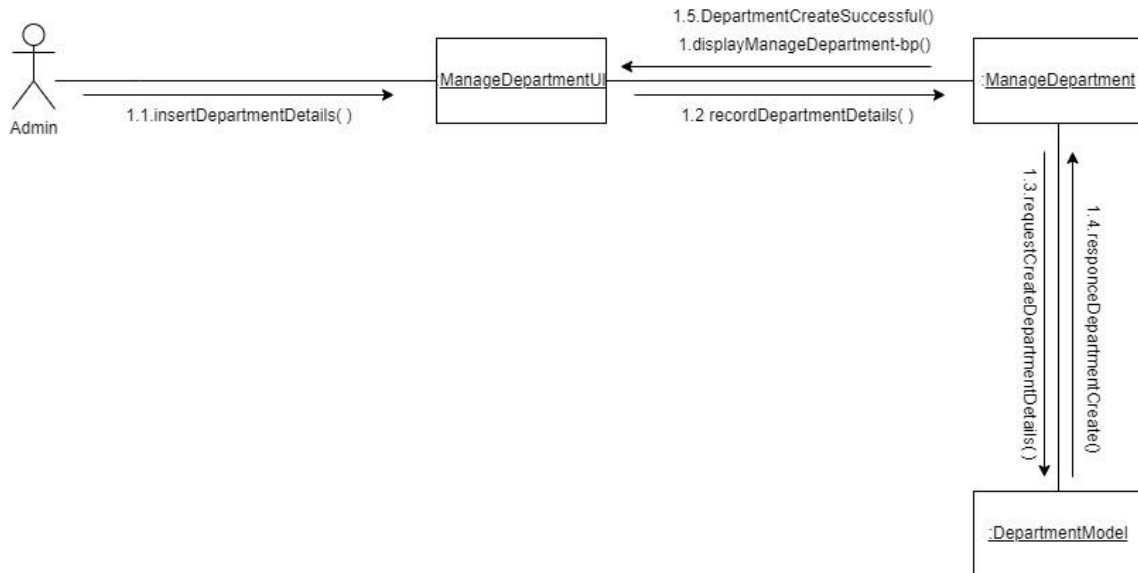


Figure 55(5.4): Manage Department

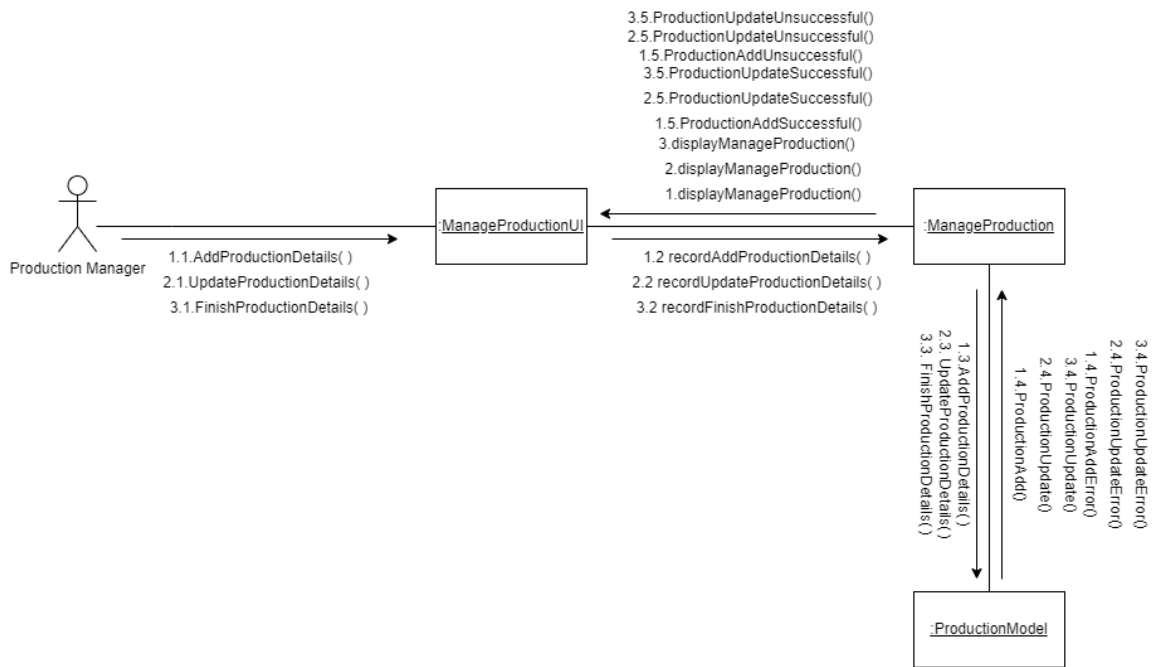


Figure 60(5.4): Manage Production

5.5 ERD

Visual Paradigm Online Free Edition

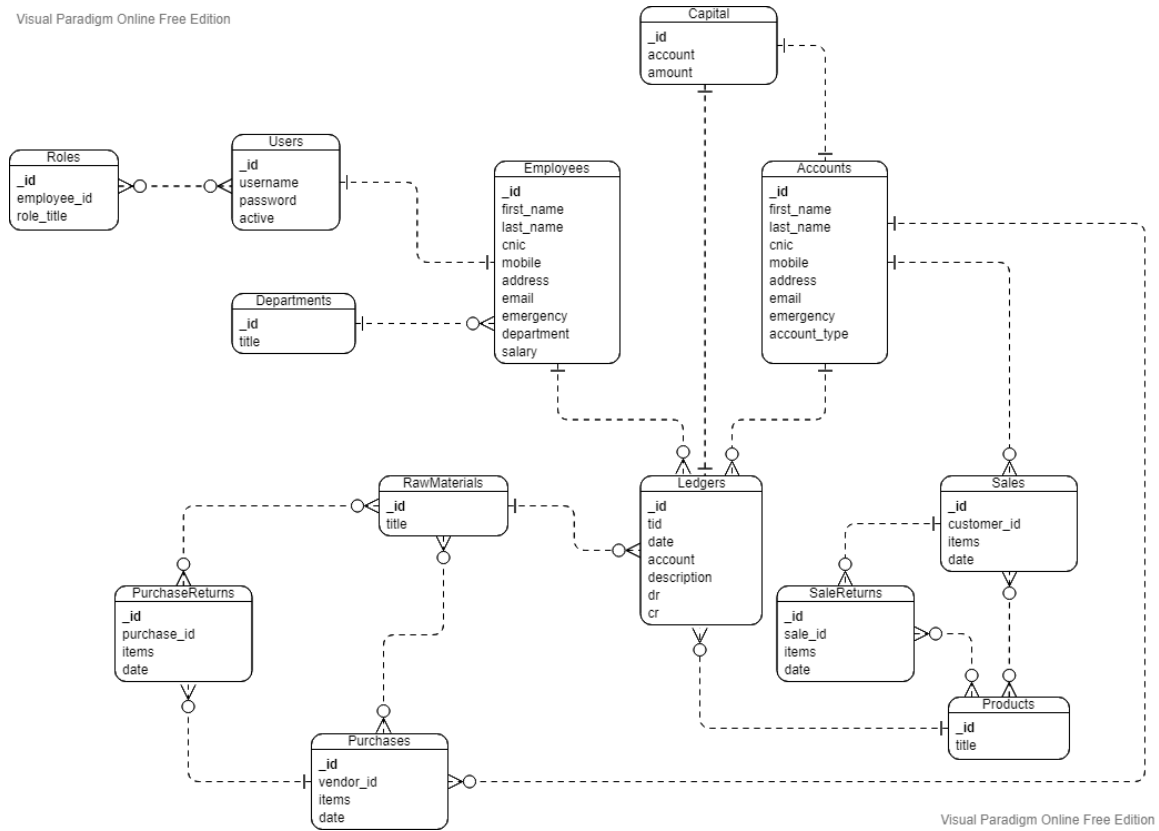


Figure 61(5.5): ERD

5.6 Data Dictionary

Table 24(5.6): Data Dictionary

Entity Name	Column Name	Data Type	Required	Accepts Values?	Null
Roles	_id	Mongo.ObjectID	Yes	No	
Roles	employee_id	Mongo.ObjectID	Yes	No	
Roles	role_title	String	Yes	No	
Users	_id	Mongo.ObjectID	Yes	No	
Users	username	String	Yes	No	
Users	password	String	Yes	No	
Users	active	Boolean	Yes	No	
Departments	_id	Mongo.ObjectID	Yes	No	
Departments	title	String	Yes	No	
Products	_id	Mongo.ObjectID	Yes	No	
Products	title	String	Yes	No	
RawMaterials	_id	Mongo.ObjectID	Yes	No	
RawMaterials	title	String	Yes	No	
Capitals	_id	Mongo.ObjectID	Yes	No	
Capitals	account	Mongo.ObjectID	Yes	No	
Capitals	amount	Number	Yes	No	
Employees	_id	Mongo.ObjectID	Yes	No	
Employees	first_name	String	Yes	No	
Employees	last_name	String	Yes	No	
Employees	cnic	Number	Yes	No	
Employees	mobile	Number	Yes	No	
Employees	address	String	Yes	No	
Employees	email	String	No	No	
Employees	emergency	Number	No	No	
Employees	department	String	Yes	No	
Employees	salary	Float	Yes	No	
Accounts	_id	Mongo.ObjectID	Yes	No	
Accounts	first_name	String	Yes	No	
Accounts	last_name	String	Yes	No	
Accounts	cnic	Number	Yes	No	

Accounts	mobile	Number	Yes	No
Accounts	address	String	Yes	No
Accounts	email	String	No	No
Accounts	emergency	Number	No	No
Accounts	account_type	String	Yes	No
Ledgers	_id	Mongo.ObjectID	Yes	No
Ledgers	tid	Mongo.ObjectID	Yes	No
Ledgers	account	Mongo.ObjectID	Yes	No
Ledgers	date	String	Yes	No
Ledgers	description	String	Yes	No
Ledgers	dr	Number	Yes	No
Ledgers	cr	Number	Yes	No
Sales	_id	Mongo.ObjectID	Yes	No
Sales	customer_id	Mongo.ObjectID	Yes	No
Sales	items	String	Yes	No
Sales	date	String	Yes	No
SaleReturns	_id	Mongo.ObjectID	Yes	No
SaleReturns	sale_id	Mongo.ObjectID	Yes	No
SaleReturns	items	String	Yes	No
SaleReturns	date	String	Yes	No
Purchases	_id	Mongo.ObjectID	Yes	No
Purchases	vendor_id	Mongo.ObjectID	Yes	No
Purchases	items	String	Yes	No
Purchases	date	String	Yes	No
PurchaseReturns	_id	Mongo.ObjectID	Yes	No
PurchaseReturns	purchase_id	Mongo.ObjectID	Yes	No
PurchaseReturns	items	String	Yes	No
PurchaseReturns	date	String	Yes	No

6. IMPLEMENTATION DETAILS

6.1 Development Setup

Table 25(6.1): Development Setup

Tool/Technology	Description
VS Code	Used VS Code IDE for coding.
Postman	Postman is use to test API.
Google Chrome	To run web application
Mozilla Firefox	To run web application
MongoDB	NoSQL database
React JS	JS library for front end development
Express JS	JS frameforwork for RESTful API development
Node JS	JS enviroment
Redux Toolkit	JS library for state management
Json Web Token	For creating x-access token
Mongoose	JS library for MongoDB
Font Awesome	CDN for fonts and icons
React-Bootstrap	JS bootstrap library for frontend

6.2 Deployment setup

We created a github repository and named it FYP. Each member of group was assigned a task and after completing the task that member will commit and push his code to the github repository. Group leader assigns new task after the iteration.

6.3 Algorithms

6.3.1 Add Account

```
route.post("/", async (req, res) => {  
  const data = req.body;  
  if (!data.fname || data.fname == "") {  
    return setResponse(405, "First name is required", null, res);  
  }  
  if (iterateWithFunction(data.fname, isNumeric)) {  
    return setResponse(405, "First name can not contain number", null, res);  
  }  
  if (iterateWithFunction(data.fname, isSymbol)) {  
    return setResponse(405, "First name can not contain symbol", null, res);  
  }  
}
```

```

    }
    if (!data.lname || data.lname == "") {
        return setResponse(405, "Last name is required", null, res);
    }
    if (iterateWithFunction(data.lname, isNumeric)) {
        return setResponse(405, "Last name can not contain number", null, res);
    }
    if (iterateWithFunction(data.lname, isSymbol)) {
        return setResponse(405, "Last name can not contain symbol", null, res);
    }
    if (!data.cnic || data.cnic == "") {
        return setResponse(405, "CNIC is required", null, res);
    }
    if (iterateWithFunction(data.cnic, isAlphabat)) {
        return setResponse(
            405,
            "CNIC can not contain alphabat or symbol",
            null,
            res
        );
    }
    if (iterateWithFunction(data.cnic, isSymbol)) {
        return setResponse(405, "CNIC can not contain symbol", null, res);
    }
    if (data.cnic.split("").length < 13 || data.cnic.split("").length > 13) {
        return setResponse(405, "CNIC length should be 13", null, res);
    }
    const cnicExist = await Account.findOne({ cnic: data.cnic });
    if (cnicExist) {
        return setResponse(405, "Account already exist with this cnic", null, res);
    }
    if (!data.mobile || data.mobile == "") {
        return setResponse(405, "Mobile number is required", null, res);
    }
    if (iterateWithFunction(data.mobile, isAlphabat)) {
        return setResponse(
            405,
            "Mobile number can not contain alphabat or symbol",
            null,
            res
        );
    }
    if (iterateWithFunction(data.mobile, isSymbol)) {

```

```

        return setResponse(405, "Mobile number can not contain symbol", null, res);
    }
    if (data.mobile.split("").length < 11 || data.mobile.split("").length > 11) {
        return setResponse(405, "Mobile number length should be 11", null, res);
    }
    if (!data.address || data.address == "") {
        return setResponse(405, "Address is required", null, res);
    }
    if (data.email) {
        const validEmail =
            /^[a-zA-Z0-9.!#$%&'*/=?^_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)$/;
        if (!data.email.match(validEmail)) {
            return setResponse(405, "Email address is not valid", null, res);
        }
        const emailExist = await Account.findOne({ email: data.email });
        if (emailExist) {
            return setResponse(
                405,
                "Account already exist with this email",
                null,
                res
            );
        }
    }
    if (data.emergency) {
        if (iterateWithFunction(data.emergency, isAlphabat)) {
            return setResponse(
                405,
                "Emergency number can not contain alphabat or symbol",
                null,
                res
            );
        }
        if (iterateWithFunction(data.emergency, isSymbol)) {
            return setResponse(
                405,
                "Emergency number can not contain symbol",
                null,
                res
            );
        }
        if (
            data.emergency.split("").length < 11 ||

```



```

        data.emergency.split("").length > 11
    ) {
        return setResponse(
            405,
            "Emergency number length should be 11",
            null,
            res
        );
    }
}

if (!data.accountType || data.accountType == "") {
    return setResponse(405, "Account Type is required", null, res);
}

data.status = true;
const newAccount = new Account(data);
newAccount.save();
return setResponse(201, "Account Created", newAccount, res);
});

```

6.3.2 Generate Salaries

```

route.post("/", async (req, res) => {
    const date = new Date();
    const year = date.getFullYear();
    const month = date.getMonth() + 1;
    const day = date.getDate();
    const myDate = month + "/" + year;
    const salaryExist = await Salary.findOne({ date: myDate });
    if (salaryExist) {
        return setResponse(
            405,
            "Salaries already gennerated for this month",
            null,
            res
        );
    }

    const employees = await Employee.find({});
    const attendance = await Attendance.find({});
    const newAttendance = [];
    attendance.map((i) => {
        if (i.date.search(myDate) > -1) {
            i.checkIn =
                parseInt(i.checkIn[0] + "" + i.checkIn[1]) * 60 +
                parseInt(i.checkIn[3] + "" + i.checkIn[4]);
            i.checkOut =

```

```

        parseInt(i.checkOut[0] + "" + i.checkOut[1]) * 60 +
        parseInt(i.checkOut[3] + "" + i.checkOut[4]);
        i.checkOut = i.checkOut - i.checkIn;
        newAttendance.push(i);
    }
});
const final = [];
employees.map((i) => {
    const cnic = i.cnic;
    const temp = {
        account: i._id,
        amount: 0,
        date: myDate,
    };
    newAttendance.map((j) => {
        if (j.cnic == cnic) {
            temp.amount = temp.amount + parseInt(j.checkOut);
        }
    });
    temp.amount = parseInt(temp.amount) * i.salaryPerMinute;
    final.push(temp);
});
await Salary.insertMany(final);
return setResponse(200, "Salaries generated", null, res);
});

```

6.3.3 Add Raw Material

```

route.post("/", async (req, res) => {
    const data = req.body;
    const response = {
        status: null,
        message: null,
        data: null,
    };
    if (!data.title || data.title == "") {
        response.status = 405;
        response.message = "Title field is required";
        return res.status(response.status).json(response);
    }
    const isExist = await RawMaterial.findOne({ title: data.title });
    if (isExist) {
        response.status = 405;
        response.message = "Raw Material already exist";
        return res.status(response.status).json(response);
    }
}

```

```

    }

    const rawMaterial = new RawMaterial(data);
    await rawMaterial.save();
    response.status = 201;
    response.message = "Raw Material added";
    response.data = rawMaterial;
    return res.status(response.status).json(response);
  });

```

6.3.4 Add Purchase

```

route.post("/", async (req, res) => {
  const data = req.body;
  if (!data.vendor) {
    return setResponse(405, "Vendor is required", null, res);
  }
  const vendorExist = await Account.findOne({
    _id: ObjectId(data.vendor),
    accountType: "vendor",
  });
  if (!vendorExist) {
    return setResponse(405, "Account type not allowed", null, res);
  }
  if (!data.paidAmount) {
    return setResponse(405, "Paid Amount is required", null, res);
  }
  if (iterateWithFunction(data.paidAmount, isAlphabat)) {
    return setResponse(
      405,
      "Amount can not contain alphabat or symbol",
      null,
      res
    );
  }
  if (!data.rawMaterial) {
    return setResponse(405, "Raw Material is required", null, res);
  }
  if (data.rawMaterial.length === 0) {
    return setResponse(405, "Raw Material is required", null, res);
  }

  if (
    data.rawMaterial.some(async (i) => {
      if (!i.material) {
        return true;
      }
    })
  ) {
    return setResponse(405, "Raw Material is required", null, res);
  }

```

```

    }

    const myRawMaterial = await RawMaterial.findOne({
      _id: ObjectId(i.material),
    });

    if (!myRawMaterial) {
      return true;
    }

    if (!i.quantity) {
      return true;
    }

    if (parseInt(i.quantity) < 1) {
      return true;
    }

    if (!i.unitPrice) {
      return true;
    }

    if (parseFloat(i.unitPrice) < 0.1) {
      return true;
    }
  })
) {
  return setResponse(405, "Invalid Material", null, res);
}

const date = new Date();
const day = date.getDate();
const month = date.getMonth() + 1;
const year = date.getFullYear();
const now = `${month}/${day}/${year}`;
data.date = now;
const count = await Purchase.find({});
data.invoice = count.length + 1;
const newPurchase = new Purchase(data);
newPurchase.save();
return setResponse(201, "Purchase Added", newPurchase, res);
});

```

6.3.5 Add Employee

```

route.post("/", async (req, res) => {
  const data = req.body;
  if (!data.fname || data.fname == "") {
    return setResponse(405, "First name is required", null, res);
  }
  if (iterateWithFunction(data.fname, isNumeric)) {

```

```

        return setResponse(405, "First name can not contain number", null, res);
    }
    if (iterateWithFunction(data.fname, isSymbol)) {
        return setResponse(405, "First name can not contain symbol", null, res);
    }
    if (!data.lname || data.lname == "") {
        return setResponse(405, "Last name is required", null, res);
    }
    if (iterateWithFunction(data.lname, isNumeric)) {
        return setResponse(405, "Last name can not contain number", null, res);
    }
    if (iterateWithFunction(data.lname, isSymbol)) {
        return setResponse(405, "Last name can not contain symbol", null, res);
    }
    if (!data.cnic || data.cnic == "") {
        return setResponse(405, "CNIC is required", null, res);
    }
    if (iterateWithFunction(data.cnic, isAlphabat)) {
        return setResponse(
            405,
            "CNIC can not contain alphabat or symbol",
            null,
            res
        );
    }
    if (iterateWithFunction(data.cnic, isSymbol)) {
        return setResponse(405, "CNIC can not contain symbol", null, res);
    }
    if (data.cnic.split("").length < 13 || data.cnic.split("").length > 13) {
        return setResponse(405, "CNIC length should be 13", null, res);
    }
    const cnicExist = await Employee.findOne({ cnic: data.cnic });
    if (cnicExist) {
        return setResponse(405, "Employee already exist with this cnic", null,
res);
    }
    if (!data.mobile || data.mobile == "") {
        return setResponse(405, "Mobile number is required", null, res);
    }
    if (iterateWithFunction(data.mobile, isAlphabat)) {
        return setResponse(

```

```

        405,
        "Mobile number can not contain alphabat or symbol",
        null,
        res
    );
}
if (iterateWithFunction(data.mobile, isSymbol)) {
    return setResponse(405, "Mobile number can not contain symbol", null, res);
}
if (data.mobile.split("").length < 11 || data.mobile.split("").length > 11) {
    return setResponse(405, "Mobile number length should be 11", null, res);
}
if (!data.address || data.address == "") {
    return setResponse(405, "Address is required", null, res);
}
if (data.email) {
    const validEmail =
        /^[a-zA-Z0-9.!#$%&'*/=?^_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)$/;
    if (!data.email.match(validEmail)) {
        return setResponse(405, "Email address is not valid", null, res);
    }
    const emailExist = await Employee.findOne({ email: data.email });
    if (emailExist) {
        return setResponse(
            405,
            "Employee already exist with this email",
            null,
            res
        );
    }
}
if (data.emergency) {
    if (iterateWithFunction(data.emergency, isAlphabat)) {
        return setResponse(
            405,
            "Emergency number can not contain alphabat or symbol",
            null,
            res
        );
    }
}

```

```

    if (iterateWithFunction(data.emergency, isSymbol)) {
        return setResponse(
            405,
            "Emergency number can not contain symbol",
            null,
            res
        );
    }
    if (
        data.emergency.split("").length < 11 ||
        data.emergency.split("").length > 11
    ) {
        return setResponse(
            405,
            "Emergency number length should be 11",
            null,
            res
        );
    }
}
if (!data.department || data.department == "") {
    return setResponse(405, "Department is required", null, res);
}
if (!ObjectId.isValid(data.department)) {
    return setResponse(405, "Department type is invalid", null, res);
}
if (!data.salary || data.salary == "") {
    return setResponse(405, "Salary is required", null, res);
}
if (iterateWithFunction(data.salary, isAlphabat)) {
    return setResponse(
        405,
        "Salary can not contain alphabat or symbol",
        null,
        res
    );
}
if (iterateWithFunction(data.salary, isSymbol)) {
    return setResponse(405, "Salary can not contain symbol", null, res);
}
}

```

```

    data.status = true;
    const eSalary = parseFloat(data.salary);
    const minuteSalary = (eSalary / 40) / 60;
    data.salaryPerMinute = Math.round(minuteSalary);
    const newEmployee = new Employee(data);
    newEmployee.save();
    return setResponse(201, "Employee Created", newEmployee, res);
});

```

6.3.6 Capital Management

```

const route = require("express").Router();
const { Capital, Account } = require("../models/model");
const ObjectId = require("mongoose").Types.ObjectId;

function setResponse(status = null, message = null, data = null, resp) {
    const response = {};
    response.status = status;
    response.message = message;
    response.data = data;
    return resp.status(response.status).json(response);
}

const isAlphabat = (val) => {
    return isNaN(val);
};

const isSymbol = (val) => {
    const ascii = val.charCodeAt(0);
    if (ascii < 48 || ascii > 122) {
        return true;
    }
    if (ascii > 57 && ascii < 65) {
        return true;
    }
    if (ascii > 90 && ascii < 97) {
        return true;
    }
    return false;
};

const iterateWithFunction = (data, func) => {
    return data.split("").some((i) => {
        return func(i);
    });
};

```



```

    });
};

// Get all capitals
route.get("/", async (req, res) => {
    const capitals = await Capital.find();
    return setResponse(200, null, capitals, res);
});

// Get single capital
route.get("/:id", async (req, res) => {
    if (!ObjectId.isValid(req.params.id)) {
        return setResponse(405, "Invalid ID", null, res);
    }
    const capital = await Capital.findOne({ _id: ObjectId(req.params.id) });
    if (!capital) {
        return setResponse(404, "Capital not found", capital, res);
    }
    return setResponse(200, null, capital, res);
});

// Add new capital
route.post("/", async (req, res) => {
    const data = req.body;
    if (!data.account || data.account == "") {
        return setResponse(405, "Account is required", null, res);
    }
    if (!ObjectId.isValid(data.account)) {
        return setResponse(405, "Invalid ID", null, res);
    }
    const accountExist = await Account.findOne({
        _id: ObjectId(data.account),
        accountType: "investor",
    });
    if (!accountExist) {
        return setResponse(405, "Invalid account id", null, res);
    }
    if (!data.amount || data.amount == "") {
        return setResponse(405, "Amount is required", null, res);
    }
    if (iterateWithFunction(data.amount, isAlphabat)) {
        return setResponse(
            405,

```

```

        "AMOUNT can not contain alphanat or symbol",
        null,
        res
    );
}
if (iterateWithFunction(data.amount, isSymbol)) {
    return setResponse(405, "Amount can not contain symbol", null, res);
}
const newCapital = new Capital(data);
newCapital.save();
return setResponse(201, "Capital Added", newCapital, res);
});

// Delete capital
route.delete("/:id", async (req, res) => {
    if (!ObjectId.isValid(req.params.id)) {
        return setResponse(405, "Invalid ID", null, res);
    }
    const capitalExist = await Capital.findOne({
        _id: ObjectId(req.params.id),
    });
    if (!capitalExist) {
        return setResponse(404, "Capital not found", null, res);
    }
    const deleted = await Capital.deleteOne({ _id: ObjectId(req.params.id) });
    return setResponse(200, "Capital deleted", deleted, res);
});

module.exports = route;

```

6.3.7 Sales Management

```

const route = require("express").Router();
const { Sale, Product, Account } = require("../models/model");
const ObjectId = require("mongoose").Types.ObjectId;

function setResponse(status = null, message = null, data = null, resp) {
    const response = {};
    response.status = status;
    response.message = message;
    response.data = data;
    return resp.status(response.status).json(response);
}

const isAlphanat = (val) => {

```

```

    return isNaN(val);
};

const iterateWithFunction = (data, func) => {
    return data.split("").some((i) => {
        return func(i);
    });
};

function getQuantity(data, id) {
    if (data.length === 0) return 0;
    let qty = 0;
    for (let i = 0; i < data.length; i++) {
        if (data[i].item === id) {
            qty = qty + parseInt(data[i].quantity);
        }
    }
    return qty;
}

// Get all sales
route.get("/", async (req, res) => {
    const sales = await Sale.find({});
    return setResponse(200, null, sales, res);
});

// Get single sale
route.get("/:id", async (req, res) => {
    if (!ObjectId.isValid(req.params.id)) {
        return setResponse(405, "Invalid ID", null, res);
    }
    const sale = await Sale.findOne({ _id: ObjectId(req.params.id) });
    if (!sale) {
        return setResponse(404, "Sale record not found", null, res);
    }
    return setResponse(200, null, sale, res);
});

// Add new sale
route.post("/", async (req, res) => {
    const data = req.body;
    if (!data.customer) {
        return setResponse(405, "Customer is required", null, res);
    }

```

```

}

const customerExist = await Account.findOne({
  _id: ObjectId(data.customer),
  accountType: "customer",
});

if (!customerExist) {
  return setResponse(405, "Account type not allowed", null, res);
}

if (!data.receivedAmount) {
  return setResponse(405, "Received Amount is required", null, res);
}

if (iterateWithFunction(data.receivedAmount, isAlphabat)) {
  return setResponse(
    405,
    "Amount can not contain alphabat or symbol",
    null,
    res
  );
}

if (!data.product) {
  return setResponse(405, "Product is required", null, res);
}

if (data.product.length === 0) {
  return setResponse(405, "Product is required", null, res);
}

if (
  data.product.some(async (i) => {
    if (!i.item) {
      return true;
    }

    const myProduct = await Product.findOne({
      _id: ObjectId(i.item),
    });

    if (!myProduct) {
      return true;
    }

    if (!i.quantity) {
      return true;
    }

    if (parseInt(i.quantity) < 1) {
      return true;
    }
  })
)

```

```

        if (!i.unitPrice) {
            return true;
        }
        if (parseFloat(i.unitPrice) < 0.1) {
            return true;
        }
    })
} {
    return setResponse(405, "Invalid iTem", null, res);
}
const date = new Date();
const day = date.getDate();
const month = date.getMonth() + 1;
const year = date.getFullYear();
const now = `${month}/${day}/${year}`;
data.date = now;
const count = await Sale.find({});
data.invoice = count.length + 1;
const newSale = new Sale(data);
newSale.save();
return setResponse(201, "Sale Added", newSale, res);
});

// Return item
route.put("/:id", async (req, res) => {
    const data = req.body;
    if (!ObjectId.isValid(req.params.id)) {
        return setResponse(405, "Invalid ID", null, res);
    }
    if (!data.product || data.product.length === 0) {
        return setResponse(405, "Nothing to return", null, res);
    }
    const mySale = await Sale.findOne({ _id: ObjectId(req.params.id) });
    if (
        data.product.some((i) => {
            if (!i.quantity) {
                return true;
            }
            if (!i.item) {
                return true;
            }
            if (parseInt(i.quantity) < 1) {
                return true;
            }
        })
    ) {
        return setResponse(405, "Invalid quantity", null, res);
    }
    const newSale = new Sale({
        ...mySale.toObject(),
        ...data,
    });
    newSale.save();
    return setResponse(201, "Sale Updated", newSale, res);
});

```

```

    }
    if (
      parseInt(getQuantity(mySale.returns, i.item)) +
      parseInt(i.quantity) >
      parseInt(getQuantity(mySale.product, i.item))
    ) {
      return true;
    }
  })
) {
  return setResponse(405, "Invalid Return", null, res);
}
const updatedData = [...mySale.returns, ...data.product];
const update = await Sale.updateOne(
  { _id: ObjectId(req.params.id) },
  { $set: { returns: updatedData } }
);
return setResponse(200, "Return Successful", update, res);
});

module.exports = route;

```

6.4 Constraints

6.4.1 Assumptions

We assume that person using this system:

- Authorized to system
- Have access to use module
- Have data to be inserted
- Will not use the system for un-ethical causes

6.4.2 System constraints

- The system is designed to be used online.
- User must have an internet connection.
- Actor should be an authorized user of the system.

6.4.3 Restrictions

- Un-authorized users are not allowed to use system.
- Special permissions for all system users will be granted by admin.

6.4.4 Limitations

- Cannot calculate taxes.

7. TESTING

7.1 Extended Test Cases

Table 26(7.1): Test Case 1 Login

Test Case ID: 1			Test Design By: M Haris Siddiqui			
Test Module Name: login			Test Design date:12/7/22			
Test Priority: high			Test Executed By: M Haris Siddiqui			
Test title: Valid Employee Information			Test Executed Date: 12/7/22			
Description: Test the login module with valid and invalid login details						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to login page		Login form is displayed	Login form is displayed	Pass	
2	User submit form without employee id and password	Id: Password:	System displays “Employee id and password is required”	System displays “Employee id and password is required”		
3	User submit form with valid employee id and empty password field	Id:12 Password:	System displays” Password is required”	System displays” Password is required”		
4	User submit form with empty employee id and valid password field	Id: Password:1234	System displays” Employee id is required”	System displays” Employee id is required”		

5	User submit form with valid employee id and invalid password field	Id:123 Password:1255	System displays” invalid password”	System displays” invalid password”		
6	User submit form with invalid employee id and valid password field	Id:12>@ Password:12345678	System displays” invalid employee id”	System displays” invalid employee id”		
7	User submit form with valid employee id and valid password field	Id:12 Password:12345678	System displays” form submitted successfully”	System displays” form submitted successfully”		
Post Condition: User login						

Table 27(7.1): Test Case 2 Forget Password

Test Case ID: 1.1				Test Design By: M Haris Siddiqui		
Test Module Name: Forget Password				Test Design date: 12/7/22		
Test Priority: high				Test Executed By: M Haris Siddiqui		
Test title: Forget Password				Test Executed Date: 12/7/22		
Description: Employee forget the password						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to login page		Login form is displayed	Login form is displayed	Pass	
2	Navigate to forget password page		Forget form is displayed	Forget form is displayed		
3	User submit form without employee id	Id:	System display” employee id is required”	System display” employee id is required”		
4	User submit form containing alphabet in employee id	Id:12345gfifu	System display” invalid employee id”	System display” invalid employee id”		
5	User submit form containing employee id length not equal to 13	Id: 1234567	System display” invalid employee id”	System display” invalid employee id”		
6	User submit form containing employee id	Id:1234567892345	System display” Password reset link	System display” Password reset link sent to		

	length equal to 13		sent to your email address”	your email address”		
Post Condition: New password send successfully.						

Table 28(7.1): Test Case Manage Account

Test Case ID: 3				Test Design By: M Haris Siddiqui		
Test Module Name: Manage Account				Test Design date: 12/7/22		
Test Priority: high				Test Executed By: M Haris Siddiqui		
Test title: Add account / Update Account				Test Executed Date: 12/7/22		
Description: Actor would be able to add and update accounts.						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to add / Update account page		Add account form is displayed	Add account form is displayed	Pass	
2	User Submit form with empty first name field	First name:	System displays" First name is required"	System displays" First name is required"		
3	User submit form in which first name containing number or special character	First name:Ha@1	System displays" Invalid first name"	System displays" Invalid first name"		
4	User Submit form with empty last name field	Last name:	System displays" last name is required"	System displays" last name is required"		
5	User submit form in which last name containing number or special character	Last name:sid@12	System displays" Invalid last name"	System displays" Invalid last name"		
6	User	CNIC:	System displays	System displays		

	submit form with empty CNIC		“CNIC required”. is	“CNIC required”. is		
7	User submit form in which CNIC containing alphabet or special character	CNIC:234sa@	System displays “CNIC is invalid”.	System displays “CNIC is invalid”.		
8	User submit form in which CNIC length is not equal to 13	CNIC:123456	System displays “CNIC is invalid”.	System displays “CNIC is invalid”.		
9	User submit form with empty mobile number field	Mobile number:	System displays” Mobile number is required ”	System displays” Mobile number is required ”		
10	User submit form containing alphabet and special character in mobile number field	Mobile No:12kdd@	System displays” Mobile number is invalid ”	System displays” Mobile number is invalid ”		
11	User submit form in which mobile number length is not equal to 11	Mobile No:123456				
12	User	Address:	System	System display”		

	submit form with empty address field		display” Address is required ”	Address is required ”		
13	User submit form with invalid email address format	Email address:harissid@gmail.com	System displays” Email address is invalid”	System displays” Email address is invalid”		
14	User submit form containing alphabet and special char in emergency number field	Emergency number:123h@	System display” Emergency number is invalid”	System display” Emergency number is invalid”		
15	User submit form in which emergency number length is not equal to 11	Emergency number:123456	System display” Emergency number is invalid”	System display” Emergency number is invalid”		
14	User submit form in which account type is not selected		System display “Account type is required”	System display “Account type is required”		
15	User submit form in which opening		System Display” Opening balance is	System Display” Opening balance is required”		

	balance field is empty		required”			
16	User submit form containing alphabet and special char in opening balance field	Opening balance:12g@	System display” Opening balance is invalid”	System display” Opening balance is invalid”		
17	User submit form in which opening balance is not selected		System display” Balance type is required”	System display” Balance type is required”		
18	User submit form with correct first name, Last name, CNIC, Mobile number, Address, Email address, emergency number, Account type, opening balance, opening balance type,		System display” Account added successfully”	System display” Account added successfully”		
Post condition: User account added successfully.						

Table 29(7.1): Test Case 4 Manage Capital

Test Case ID: 4				Test Design By: M Haris Siddiqui		
Test Module Name: Manage Capital				Test Design date: 12/8/22		
Test Priority: high				Test Executed By: M Haris Siddiqui		
Test title: Add / Update Capital				Test Executed Date: 12/8/22		
Description: Test update and add capital with valid and invalid details						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to add capital page		Add capital form is displayed	Add capital form is displayed	Pass	
2	User submit form in which account field is not selected		System displays “Amount is required”.	System displays “Amount is required”.		
3	User submit form in which amount field is empty	Amount :	System displays “Amount is required”.	System displays “Amount is required”.		
4	User submit form containing special char and alphabet in amount field	Amount:123d@	System displays “Amount is invalid”.	System displays “Amount is invalid”.		
5	User submit form with valid information	Amount:30000	System display” Form submitted successfully”	System display” Form submitted successfully”		
Post Condition: User added or updated capital successfully.						

Table 30(7.1): Test Case 6 Manage Expense

Test Case ID: 6				Test Design By: M Haris Siddiqui		
Test Module Name: Manage Expense				Test Design date: 12/8/22		
Test Priority: high				Test Executed By: M Haris Siddiqui		
Test title: Add expense				Test Executed Date: 12/8/22		
Description: Actor will able to add expense.						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	User navigate to add expense page		Expense page is displayed	Expense page is displayed	Pass	
2	User submit form in which expense title is empty	Expense title:	System displays “Expense title is required”.	System displays “Expense title is required”.		
3	User submit form in which expense amount is negative number	Expense amount:-12	System display “Expense amount should be greater than 0”.	System display “Expense amount should be greater than 0”.		
4	User submit form in which expense amount field is empty	Expense amount:	System displays “Expense amount is required”.	System displays “Expense amount is required”		
5	User submit form in which expense amount is not a number	Expense amount: dhdh	System displays “Expense amount should be a number”.	System displays “Expense amount should be a number”.		

Post Condition: User added expense successfully.						

Table 31(7.1): Test Case 7 Manage Payments (Payments In)

Test Case ID: 7				Test Design By: M Haris Siddiqui		
Test Module Name: Manage Payment				Test Design date: 12/8/22		
Test Priority: Middle				Test Executed By: M Haris Siddiqui		
Test title: Payment In				Test Executed Date: 12/8/22		
Description: Test whether payment In details are valid						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	User navigate to Payment In page		Payment In form is displayed	Payment In form is displayed	Pass	
2	User submitted form in which customer account is not selected		System display “Customer account is required”.	System display “Customer account is required”.		
3	User submitted form in which sale invoice is not selected		System display “Sale invoice is required”.	System display “Sale invoice is required”.		
4	User submitted form in which amount field is empty		System display “Amount is required”.	System display “Amount is required”.		

5	User submitted form containing alphabet in amount field		System display "Please enter amount in numbers only".	System display "Please enter amount in numbers only".		
6	User submitted form in which amount is greater than receivable amount		System display "Amount can't be greater than receivable amount".	System display "Amount can't be greater than receivable amount".		
Post Condition: User manage payment In successfully						

Table 32(): Test Case 8 Manage Payments (Payments Out)

Test Case ID: 7.1				Test Design By: M Haris Siddiqui		
Test Module Name: Manage Payment				Test Design date: 12/8/22		
Test Priority: Middle				Test Executed By: M Haris Siddiqui		
Test title: Payment out				Test Executed Date: 12/8/22		
Description: Test the payment out details are valid or not						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	User navigate to Payment out page		Payment out form is displayed	Payment out form is displayed	Pass	
2	User submitted form in which vendor account is not selected		System display “Vendor account is required”.	System display “Vendor account is required”.		
3	User submitted form in which purchase invoice is not selected		System display “Purchase invoice is required”.	System display “Purchase invoice is required”.		
4	User submitted form in which amount field is empty		System display “Amount is required”.	System display “Amount is required”.		
5	User submitted form containing alphabet in amount field		System display “Please enter amount in numbers only”.	System display “Please enter amount in numbers only”.		

6	User submitted form in which amount is greater than payable amount		System display“ Amount can’t be greater than payable amount	System display“ Amount can’t be greater than payable amount		
Post Condition: User manage payment out successfully						

Table 33(7.1): Test Case 9 Manage Purchases

Test Case ID: 9				Test Design By: M Haris Siddiqui		
Test Module Name: Manage Purchases				Test Design date: 12/8/22		
Test Priority: Middle				Test Executed By: M Haris Siddiqui		
Test title: Add Purchases				Test Executed Date: 12/8/22		
Description: Test the add purchase field.						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to add purchase page		Add purchase form is displayed	Add purchase form is displayed	Pass	
2	User submit form in which vendor field is not selected		System display “Vendor field is required”	System display “Vendor field is required”		
3	User submit form in which paid amount field is empty	Paid amount:	System display “Paid amount field is required”	System display “Paid amount field is required”		
4	User submit form in which paid amount field contain alphabet	Paid amount:12s	System display “Paid amount is invalid”	System display “Paid amount is invalid”		

5	User submit form in which paid amount field contain negative number	Paid amount:-12	System display “Paid amount is invalid”	System display “Paid amount is invalid”		
6	User submit form in which raw material field is not selected		System display “Raw material field is required”	System display “Raw material field is required”		
7	User submit form in which raw material quantity field is empty	Raw material quantity:	System display “Raw material quantity field is required”	System display “Raw material quantity field is required”		
8	User submit form in which raw material quantity field contain alphabet	Raw material quantity:12s	System display “Raw material quantity is invalid”	System display “Raw material quantity is invalid”		
9	User submit form in which raw material quantity field contain negative number	Raw material quantity:-12	System display “Raw material quantity is invalid”	System display “Raw material quantity is invalid”		
10	User submit form in which per unit price field is empty	Per unit price:	System display “Per unit price field is required”	System display “Per unit price field is required”		
11	User submit form in which per unit price	Per unit price:123g	System display “Per unit price is	System display “Per unit price is invalid”		

	field contain alphabet		invalid”			
12	User submit form in which per unit price field contain negative number	Per unit price:-123	System display “Per unit price is invalid”	System display “Per unit price is invalid”		
13	User submit form in which expense title field is empty	Expense title:	System display “Expense title field is required”	System display “Expense title field is required”		
14	User submit form in which expense cost field is empty	Expense cost:	System display “Expense cost field is required”	System display “Expense cost field is required”		
Post Condition: User added purchases successfully						

Table 34(7.1): Test Case 10 Add sale

Test Case ID: 11			Test Design By: M Zeeshan Fiaz			
Test Module Name: Add sale			Test Design date: 12/7/22			
Test Priority: high			Test Executed By: M Zeeshan Fiaz			
Test title: Valid Add sale Information			Test Executed Date: 12/7/22			
Description: Test the add sale module with valid and invalid data						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to Sale management page		Sale management is displayed	Sale management is displayed	Pass	
2	User submit form without selecting customer id	Customer Id:	System displays “Customer id is required”	System displays “Customer id is required”		
3	User submit form with empty paid amount	Paid amount:	System displays” Paid amount is required”	System displays” Paid amount is required”		
4	User submit form that contains alphabets in Paid Amount	Paid Amount: 1200@78	System displays” Paid amount is invalid”	System displays” Paid amount is invalid”		
5	User submit form that contains Negative number in Paid Amount	Paid Amount: -12500	System displays” Paid amount is invalid”	System displays” Paid amount is invalid”		
6	User submit form without selecting	Product:	System displays” Product field	System displays” Product field		

	Product		is required”	is required”		
7	User submit form without selecting Product quantity	Product quantity:	System displays” Product quantity field is required”	System displays” Product quantity field is required”		
8	User submit form that contains Alphabets in Product quantity	Product quantity: 675@67	System displays” Product quantity is invalid”	System displays” Product quantity is invalid”		
9	User submit form that contains Negative number in Product quantity	Product quantity: -656756	System displays” Product quantity is invalid”	System displays” Product quantity is invalid”		
10	User submit form that contains alphabets in Per Unit Price	Per Unit price:	System displays “Per unit price is invalid”	System displays “Per unit price is invalid”		
11	User submit form that contains Negative number in Per unit price	Per unit price: -236154	System displays “Per unit price is invalid”	System displays “Per unit price is invalid”		
12	User submit form with empty Expense title	Expense title:	System displays ”Expense title field is required”	System displays ”Expense title field is required”		
13	User submit form with empty Expense cost	Expense Cost:	System displays” Expense cost field is required”	System displays” Expense cost field is required”		

14	User submit form that contains alphabets in Expense cost	Expense cost: 145@65	System displays" Expense cost is invalid"	System displays" Expense cost is invalid"		
15	User submit form that contains negative number in Expense cost	Expense cost: -18768	System displays" Expense cost is invalid"	System displays" Expense cost is invalid"		
16	User submit form with valid Customer, Paid Amount, Product field, Product quantity, Per unit price, Expense title and Expense cost fields.	Customer: Paid Amount: Product: Product quantity: Per Unit price: Expense title: Expense Cost:	System displays" Sale added successfully"	System displays" Sale added successfully"		
Post Condition: Sale added successfully in system.						

Table 35(7.1): Test Case 11 Return Sale

Test Case ID: 11.1			Test Design By: M Zeeshan Fiaz			
Test Module Name: Return sale			Test Design date: 12/7/22			
Test Priority: high			Test Executed By: M Zeeshan Fiaz			
Test title: Valid Return sale Information			Test Executed Date: 12/7/22			
Description: Test the return sale module with valid and invalid data						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to Sale management page		Sale management is displayed	Sale management is displayed	Pass	
2	User submit form without selecting Item field	Item Id:	System displays “Item field is required”	System displays “Item field is required”		
3	User submit form with empty return quantity field	Return quantity:	System displays” Return quantity field is required”	System displays” Return quantity field is required”		
4	User submit form that contains alphabets in Return quantity field	Return quantity: 128@962	System displays” Return quantity is invalid”	System displays” Return quantity is invalid”		

5	User submit form that contains Negative number in Return quantity field	Return quantity: -128648	System displays” Return quantity is invalid”	System displays” Return quantity is invalid”		
6	User submit form with valid Item id and Return quantity fields.	Item id: Return quantity:	System displays” Sale return added successfully”	System displays” Sale return added successfully”		
Post Condition: Sale return added successfully in system.						

Table 36(7.1): Test Case 12 Manage Products

Test Case ID: 12			Test Design By: M Zeeshan Fiaz			
Test Module Name: Manage products			Test Design date: 12/7/22			
Test Priority: high			Test Executed By: M Zeeshan Fiaz			
Test title: Valid product information			Test Executed Date: 12/7/22			
Description: Test the manage product module with valid and invalid data						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to Product management page		Product management is displayed	Product management is displayed	Pass	
2	User submit form without selecting product title field	Product title:	System displays “Product title is required”	System displays “Product title is required”		
3	User submit form without selecting product code field	Product code:	System displays “Product code is required”	System displays “Product code is required”		
4	User submit form that contains valid Product title and product code fields	Product title: Product code:	System displays” Product added successfully”	System displays” Product added successfully”		
Post Condition: Product added successfully in system.						

Table 37(7.1): Test Case 13 Manage Raw Material

Test Case ID: 13			Test Design By: M Zeeshan Fiaz			
Test Module Name: Manage Raw Material			Test Design date: 12/7/22			
Test Priority: high			Test Executed By: M Zeeshan Fiaz			
Test title: Valid raw material information			Test Executed Date: 12/7/22			
Description: Test the manage raw material module with valid and invalid data						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to Raw Material Management page		Raw Material Management is displayed	Raw Material Management is displayed	Pass	
2	User submit form without selecting Department title field	Department title:	System displays “Department title is required”	System displays “Department title is required”		
3	User submit form with valid Raw Material title field	Raw Material title:	System displays “Raw Material added successfully”	System displays “Raw Material added successfully”		
Post Condition: Raw Material added successfully						

Table 38(7.1): Test Case 14 Manage Departments

Test Case ID: 14			Test Design By: M Zeeshan Fiaz			
Test Module Name: Manage Departments			Test Design date: 12/7/22			
Test Priority: high			Test Executed By: M Zeeshan Fiaz			
Test title: Valid Department information			Test Executed Date: 12/7/22			
Description: Test the manage department module with valid and invalid data						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to Department Management page		Department Management is displayed	Department Management is displayed	Pass	
2	User submit form without selecting Raw Material title field	Raw Material title:	System displays “Raw Material title is required”	System displays “Raw Material title is required”		
3	User submit form with valid Raw Material title field	Raw Material title:	System displays “Department added successfully”	System displays “Department added successfully”		
Post Condition: Department added successfully						

Table 39(7.1): Test Case 15 Search Attendance

Test Case ID: 16			Test Design By: M Zeeshan Fiaz			
Test Module Name: Search Attendance			Test Design date: 12/7/22			
Test Priority: high			Test Executed By: M Zeeshan Fiaz			
Test title: Valid Search attendance information			Test Executed Date: 12/7/22			
Description: Test the search attendance module with valid and invalid data						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to Attendance Management page		Attendance Management is displayed	Attendance Management is displayed	Pass	
2	User submit form without selecting Date field	Date:	System displays “Date is required”	System displays “Date is required”		
3	User submit form with valid Date field	Date:	System displays “Attendance searched successfully”	System displays “Attendance searched successfully”		
Post Condition: Attendance searched successfully.						

Table 40(7.1): Test Case 16 Marked Attendance

Test Case ID: 16.1			Test Design By: M Zeeshan Fiaz			
Test Module Name: Marked Attendance			Test Design date: 12/7/22			
Test Priority: high			Test Executed By: M Zeeshan Fiaz			
Test title: Valid Marked attendance information			Test Executed Date: 12/7/22			
Description: Test the marked attendance module with valid and invalid data						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to Attendance Management page		Attendance Management is displayed	Attendance Management is displayed	Pass	
2	User submit form without selecting Employee Id field	Employee ID:	System displays “Employee ID is required”	System displays “Employee ID is required”		
3	User submit form that contains length of 13 digits in Employee ID field	Employee ID: 156 987 123 3214	System displays “Invalid Employee ID”	System displays “Invalid Employee ID”		
4	User submit form that contains alphabets in Employee ID field	Employee ID: 123@786 78681	System displays” Invalid Employee ID”	System displays” Invalid Employee ID”		
5	User submit with valid Employee Id field	Employee ID:	System displays” Attendance marked successfully”	System displays” Attendance marked successfully”		
Post Condition: Attendance marked successfully.						

Table 41(7.1): Test Case 17 Add/Update Employees

Test Case ID: 17			Test Design By: M Zeeshan Fiaz			
Test Module Name: Add/Update Employees			Test Design date: 12/7/22			
Test Priority: high			Test Executed By: M Zeeshan Fiaz			
Test title: Valid ADD/UPDATE Employees information			Test Executed Date: 12/7/22			
Description: Test the Add/Update module with valid and invalid data						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to Employee Management page		Employee Management is displayed	Employee Management is displayed	Pass	
2	User submit form with empty First name field	First name:	System displays “First name is required”	System displays “First name is required”		
3	User submit form that contains number or special characters in First name field	First Name: Abdul@Habib786	System displays “Invalid First name”	System displays “Invalid First name”		
4	User submit form with empty Last name field	Last name:	System displays” Last name is required”	System displays” Last name is required”		
5	User submit form that contains number or special characters in Last name field	Last name:	System displays ”Invalid Last name”	System displays ”Invalid Last name”		

6	User submit form with empty CNIC field	CNIC:	System displays” CNIC field is required”	System displays” CNIC field is required”		
7	User submit form that contains the special characters or alphabets in CNIC field	CNIC: 35201-78@7864-0	System displays” Invalid CNIC”	System displays” Invalid CNIC”		
8	User submit form whose CNIC length is not 13 digits	CNIC: 35301-55568798-9	System displays“ Invalid CNIC”	System displays“ Invalid CNIC”		
9	User submit form with empty Mobile Number field	Mobile Number:	System displays” Mobile Number is required”	System displays” Mobile Number is required”		
10	User submit form that contains alphabets or special characters in Mobile Number field	Mobile Number: 0316-400@863	System displays” Mobile Number us required”	System displays” Mobile Number us required”		
11	User submit the form where mobile number length is not 11	Mobile Number: 0321-58461295	System displays” Mobile Number is invalid”	System displays” Mobile Number is invalid”		

12	User submit form with empty address field	Address:	System displays" Address is required"	System displays" Address is required"		
13	User submit form with invalid Email Address field	Address: 123 765#gmail.com	System displays" Email Address is required"	System displays" Email Address is required"		
14	User submit form that contains special characters or alphabets in Emergency number field	Emergency Number:	System displays" Invalid Emergency Number"	System displays" Invalid Emergency Number"		
15	User submit form where Emergency number length is not 11	Emergency Number: 0321-23541125	System displays" Invalid Emergency Number"	System displays" Invalid Emergency Number"		
16	User submit form with empty department field	Department:	System displays" Department is required"	System displays" Department is required"		
17	User submit form with empty Salary field	Salary:	System displays" Salary is required"	System displays" Salary is required"		

18	User submit form that contains alphabets or special characters in Salary field	Salary: 174\$4521 Pkr	System displays" Salary Invalid" is	System displays" Salary Invalid" is		
19	User submit form with valid First name, Last name, CNIC, Mobile Number, Address, Emergency Number, Department and Salary	First Name: Last Name: CNIC: Mobile Number: Address: Emergency Number: Department: Salary:	System displays" Employee Added/Updated successfully"	System displays" Employee Added/Updated successfully"		
Post Condition: Employees Added/Updated successfully.						

Table 42(7.1): Test Case 18 Add/Update Production

Test Case ID: 19			Test Design By: M Zeeshan Fiaz			
Test Module Name: Add/Update Production			Test Design date: 12/7/22			
Test Priority: high			Test Executed By: M Zeeshan Fiaz			
Test title: Valid Add/Update Production information			Test Executed Date: 12/7/22			
Description: Test the add/update production module with valid and invalid data						
Pre-Condition:						
Dependencies:						
Step	Test Step	Test Date	Expected Result	Actual Result	Status	Notes
1	Navigate to Production Management page		Production Management is displayed	Production Management is displayed	Pass	
2	User submit form with empty Product code field	Product Code:	System displays “Product code is required”	System displays “Product code is required”		
3	User submit form with empty Product quantity field	Product quantity:	System displays “Product quantity is required”	System displays “Product quantity is required”		
4	User submit form that contains alphabets in Product quantity field	Product quantity: 1234sa45	System displays” Invalid Product quantity”	System displays” Invalid Product quantity”		
5	User submit with negative number product quantity field	Product quantity: -12563	System displays” Invalid Product quantity”	System displays” Invalid Product quantity”		

6	User submit form with empty Raw Material field	Raw Material:	System displays" Raw Material is required"	System displays" Raw Material is required"		
7	User submit form with empty Raw Material quantity field	Raw Material quantity:	System displays" Raw Material quantity is required"	System displays" Raw Material quantity is required"		
8	User submit form that contains alphabets in Raw Material quantity field	Raw Material quantity: 231k212	System displays" Invalid Raw Material quantity"	System displays" Invalid Raw Material quantity"		
9	User submit form with negative number in Raw Material quantity field	Raw Material quantity: -1346232	System displays" Invalid Raw Material quantity"	System displays" Invalid Raw Material quantity"		
10	User submit form with empty Expense title field	Expense title:	System displays" Expense title is required"	System displays" Expense title is required"		
11	User submit form with empty Expense cost field	Expense cost:	System displays" Expense cost is required"	System displays" Expense cost is required"		
12	User submit form that contains alphabets in Expense cost field	Expense cost:	System displays" Expense cost is invalid"	System displays" Expense cost is invalid"		

13	User submit form that contains the negative number in expense cost field	Expense cost: -12162323	System displays" Invalid Expense cost"	System displays" Invalid Expense cost"		
14	User submit form with valid Product code, Product quantity, Raw Material, Raw Material quantity, Expense title and Expense cost fields	Product code: Product quantity: Raw Material: Raw Material quantity: Expense title: Expense cost:	System displays" Production added/updated successfully"	System displays" Production added/updated successfully"		
Post Condition: Production added/updated successfully.						

7.2 Traceability Matrix

7.2.1 RID vs UCID (requirements vs use cases)

Table 43(7.3): RID vs UCID

UCID/ RID	R01	R02	R03	R04	R05	R06	R07	R08	R09	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26
UC-1	✓																			✓	✓	✓	✓	✓	✓	✓
UC-2		✓																		✓	✓		✓	✓	✓	✓
UC-3			✓																	✓	✓		✓	✓	✓	✓
UC-4				✓																✓	✓		✓	✓	✓	✓
UC-5					✓															✓	✓		✓	✓	✓	✓
UC-6						✓														✓	✓		✓	✓	✓	✓
UC-7							✓													✓	✓		✓	✓	✓	✓
UC-8								✓												✓	✓		✓	✓	✓	✓
UC-9									✓											✓	✓		✓	✓	✓	✓
UC-10										✓										✓	✓		✓	✓	✓	✓
UC-11											✓									✓	✓		✓	✓	✓	✓
UC-												✓								✓	✓		✓	✓	✓	✓

7.2.2 Test Cases (RID vs TID)

Table 44(7.3): RID vs TID

T I D/ R I D	R 0 1	R 0 2	R 0 3	R 0 4	R 0 5	R 0 6	R 0 7	R 0 8	R 0 9	R 1 0	R 1 1	R 1 2	R 1 3	R 1 4	R 1 5	R 1 6	R 1 7	R 1 8	R 1 9	R 2 0	R 2 1	R 2 2	R 2 3	R 2 4	R 2 5	R 2 6	
T C -1	✓																			✓	✓	✓	✓	✓	✓	✓	
T C -2		✓																		✓	✓			✓	✓	✓	✓
T C -3			✓																	✓	✓			✓	✓	✓	✓
T C -4				✓																✓	✓			✓	✓	✓	✓
T C -5					✓															✓	✓			✓	✓	✓	✓
T C -6						✓														✓	✓			✓	✓	✓	✓
T C -7							✓													✓	✓			✓	✓	✓	✓
T C -8								✓												✓	✓			✓	✓	✓	✓
T C -9									✓											✓	✓			✓	✓	✓	✓
T C - 10										✓										✓	✓			✓	✓	✓	✓
T C - 11											✓									✓	✓			✓	✓	✓	✓
T C -												✓								✓	✓			✓	✓	✓	✓

7.2.3 Coverage (UCID vs TID)

Table 45(7.3): UCID vs TID

UCID /RID	T C -1	T C -2	T C -3	T C -4	T C -5	T C -6	T C -7	T C -8	T C -9	T C - 10	T C - 11	T C - 12	T C - 13	T C - 14	T C - 15	T C - 16	T C - 17	T C - 18
UC-1	✓																	
UC-2		✓																
UC-3			✓															
UC-4				✓														
UC-5					✓													
UC-6						✓												
UC-7							✓											
UC-8								✓										
UC-9									✓									
UC-10										✓								
UC-11											✓							
UC-12												✓						
UC-13													✓					
UC-14														✓				
UC-15																		
UC-16															✓			
UC-17																✓		
UC-18																	✓	
UC-19																		✓

8. RESULTS/OUTPUT/STATISTICS

8.1 %completion

The project is 96% complete. More modules can be integrated but project is completed according to requirements.

8.2 %accuracy

The project accuracy is 98%. There are no test cases for backup management.

8.3 %correctness

All the requirements are satisfied and the project is 100% correct.

9. CONCLUSION

Integrated business model is completed and ready to use in industry. We can add more modules easily as we used MERN stack for development. Project can be used online and also within the premises of business as per requirements. You can generate multiple reports and keep track of business current position. Payments can be managed very easily and all the entries produced are always correct. Mr. Usman (Chartered Accountant) helped us a lot in maintaining the ledgers and accounts.

10. FUTURE WORK

In future we may add some more modules in project like Tax Calculator, biometric attendance system, AI to predict sales and barcode readers for easily organizing sales and purchases. Project will be marketed on social media and personal blogs.

11. BIBLIOGRAPHY

11.1 Books

Nil

11.2 Journals

Nil

11.3 Articles

Nil

11.4 Research papers

Nil

11.5 Other References

Nil

12. APPENDIX

12.1 Glossary of terms

UMT	University of Management and Technology
IBM	Integrated Business Model
Rankoli	A vanity factory is Johar Town, Lahore
ERP	Enterprise Resource Planning
UC	Use Case
TC	Test Case

12.2 Pre-requisites

To use the system user must have following:

- Laptop/Desktop/Mobile
- Internet Connection
- Login Credentials
- Admin Permission