A PROJECT REPORT ON

DESIGN AND IMPLEMENTATION OF AN INVENTORY MANAGEMENT SYSTEM FOR ABU ABEER AL GAWABRY TRADING

By

18F18274, IMMANUEL SAM SUHIRTHARAJ

Guided by

PUTTASWAMY M R

A Project report submitted in partial fulfillment of the requirements for the award of

BSc of Computer Science in **Data Analytics**

MIDDLE EAST COLLEGE

Knowledge Oasis Muscat, Muscat, Oman

June 2022

A PROJECT REPORT ON

INVENTORY MANAGEMENT SYSTEM FOR ABU ABEER AL GAWABRY TRADING

By

18F18274 IMMANUEL SAM SUHIRTHARAJ

June 2022

DECLARATION

I, ***IMMANUEL SAM SUHIRTHARAJ***, hereby declare that the work presented herein is genuine and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet, etc.) has been acknowledged with the main report to an item in the references or bibliography lists.

Copyright Acknowledgement

Student ID Student Name Signature

Student Name	**Student ID**	**Signature**
IMMANUEL SAM SUHIRTHARAJ	18F18274	

APPROVAL FORM

The project report entitled DESIGN AND IMPLEMENTATION OF AN INVENTORY MANAGEMENT SYSTEM FOR ABU ABEER AL GAWABRY TRADING submitted by, 18F18274 IMMANUEL SAM SUHIRTHARAJ, is approved in partial fulfillment of the requirements for bachelor's degree in Data Analytics.

Supervisor

PUTTASWAMY MR

Computing

Date

Examiner

Name

Computing

Date

ACKNOWLEDGEMENT

I'd want to offer my profound thanks and gratitude to everyone who has helped me and supported me throughout this project, as well as in the preparation and drafting of this report. The source of information for this study was provided by a faculty member of the Department of Computing and the Head, Faculty of Department. The supervisor deserves praise as well for reviewing and double-checking the structure of this report, as well as providing us with help and advice throughout the project.

ABSTRACT

This project sufficiently depicts the Inventory Management System to determine the feasibility and convenience of a fully functional system. The fundamental concept is to follow the trade from the storage of inventory and sales registers, with additional highlights to help comprehend the data.

In businesses that handle transactions involving consumer goods, an inventory management process is important for quality control. A significant retail business might run out of inventory on a critical item if inventory is not managed correctly. When it's time to refill, a good Inventory Management System will alert the merchant. An inventory system is also useful for tracking larger shipments automatically.

Counting each item by hand is likely to result in a mistake. The use of an automated Inventory Management System can aid to lower the risk of human error. An Inventory Management System at a retail business also facilitates in the tracking of retail product loss, offering valuable insights store revenues and the need for theft-prevention devices.

The approach taken to complete this project is DSDM methodology. This model is easy to comprehend and provides a beneficial method to proceed with the project.

TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

LIST OF ABBREVIATIONS

1. WBS- Work breakdown structure
2. SWOT- Strength, Weakness, Opportunities, Threats
3. MS- Microsoft
4. VS- Visual Studio
5. VSCode- Microsoft visual studio code
6. SDLC- Software development Life Cycle
7. DSDM- Dynamic System Development Method
8. UML- Unified modelling language
9. DFD- Data flow diagram
10. UI- User Interface

Chapter 1: Introduction

1.1 Introduction about the organization

Abu Abeer Al Gawabry trading is a construction material supplier company that was founded in 2004 in the sultanate of Oman. The company specializes in the trading of construction materials such as plywood, products that are based on plywood, ceramic materials, metals, lumber and lumber-based products, thermal insulation products, chemicals related to construction and others.

The division of their work is mainly targeted towards offering supplies and products to the construction of large-scale contracts such as buildings, highways, bridges as well as small scale contracts

Abu Abeer Al Gawabry trading has established itself successfully in the supply and demand market in the sultanate of Oman and provide a vast and versatile range of products. The company is fully licensed and specialized in all the fields of building materials, maintenance, and refurbishments. They have a range of well-trained work force comprised of high skill workers which accumulates over a fifty years of work experience and provide the best service available and follow proper work ethics within the Sultanate of Oman.

1.2 Problem statement

There are many problems being faced by the company. The company does not have a balanced system to be used to keep a record of their inventory and their data since the current system is not systematic. Due to this reason, it is difficult for the administrator the keep a record of the inventory and data. It a threat to safety and time since the data currently being stored is in the form of a logbook and a primitive system which makes the current system unorganized.

1.2.1 The current system

The current system used by the company is primarily manual. This causes difficulty for the staff to analyze their profit. The newly developed web application will assist in managing the inventory an easy job and helps save time and keeps the system secure. It will solve most of the current issues that are being faced by the company.

1.2.2 Problem faced

The process of recording the data into the inventory takes a lot of time. It becomes a heavy workload on the employees and managers of the company to input data manually and one by one which leads to wastage of time and information and data can be lost in the process. This in turn affects the profit that can be made and result in a data loss that can pose a major threat to the organization

1.2.3 Solution to the existing problem

An automated inventory management system will be created to save a lot of time. The web application will help keep records of customer details, banks, payments ledgers with minimum manual input which helps save a lot of time. The credentials of the users are software protect which means there will be no data loss. The system can only be accesses by the authority of the admin/manager.

1.3 Project scope statement

The inventory management system web application is solely developed for the purpose of making the workflow of the organization simpler and smooth. This will benefit the Manager as well as the employees who choose to use the system.

The primary objective of this project is to deliver a stock and inventory management system that is efficient and easy to use. The web application contains two modules, for the manager and for the employees The scope of the project will focus on issues such as database, report generation, and the store's point of sale in order to meet the specified objectives.

This new system will include a database that allows for the storage and retrieval of transaction data as well as inventory data for each item in the shop, as well as the management of product releases and storage, and the summarization of point of sales. This would result in quicker work improvisation with less time and effort. This system is supposed to aid in making the appropriate decision in the process of managing inventory aligned with the sales level in the shop, since the objective of Sales and Inventory Management System is to decrease paper effort and ineffective ways of managing inventory.

1.3.1 Project objectives

- implement an inventory management system for Abu Abeer al Gawabry trading.
- provide the organization with the modern inventory system
- Help the users analyze the profit and loss in a graphical format.
- To develop an inventory management system that is simple and easy to use.
- To make a secure system that can store personal details of customers, banks ledgers and payment systems.
- Develop a web application to monitor the stock-in and stock-out in an efficient manner.
- To generate a report based on the sales and produces monthly sales logs

1.3.2 Duration of the project

The project is expected to be completely done in a duration of four months from the 15th of October 2021 to March 22nd of 2022. The initial stages that include report making began in the beginning of November and the final stages that include the development of the system will be completed in a span of 4 months

1.3.3 Cost benefit analysis

Cost benefit compares the predicted or estimated costs and the benefits (or opportunities) linked to the project choice to determine whether it is reasonable from a business perspective.

Cost benefit analysis is used for the evaluation of the benefits with respect to the cost of doing a project. The project is then checked if its worth the effort, resources and funds after finding the difference between the cost and the benefit.

Cost

Category	**Type**	**Term 1**	**Term 2**	**Total in OMR**
Hardware	Computer	OMR. 500	-	OMR.500
	Internet	OMR. 35	OMR. 35	OMR. 70
	Peripherals	OMR. 20	-	OMR. 20
	Servers	OMR. 4	OMR. 4	OMR.8
Software	Microsoft Word	OMR. 10	-	OMR.10
	Microsoft Windows	OMR. 20	-	OMR.20
	Product license	OMR. 10	-	OMR. 10
Total				OMR. 638

Table 1: Cost table

Benefits

Quantitative benefits	Term 1	Term 2	Total
Cost of service	OMR. 30	OMR. 30	OMR. 60
Productivity Gains	OMR. 2000	OMR. 2300	4300
Savings from business improvements	OMR. 700	OMR. 720	OMR. 1420
Savings from optimized information flow	OMR. 32	OMR. 33	OMR. 55
Total			OMR. 5835

Table 2: benefit table

Profit made- OMR 5197

1.3.4 Software and programs used.

1. **Microsoft visual studio code**

The entire system will be coded using Microsoft visual studio code.

Microsoft visual studio code is a free open-source software that is used to edit code. It is made my Microsoft that runs on MacOS, Windows and Linux operating systems. It has so many outstanding features that make coding and debugging easy.

2. **Django**

Django is a master level web framework on python which will enable us to develop a sophisticated web application that has a clean and professional design. It makes the tasks of web development easy. Django is a free and open-source software.

3. **Python**

Python is a high-level programming language that is object oriented. The project will be build using python because python supports tons of modules and packages for making software development easy. Python is also the most productive programing language.

4. **Bootstrap**

Bootstrap is an open-source software that is used for designing web pages using HTML and CSS frameworks. It has JavaScript and CSS framework that comes along with it. Bootstrap has five version. The project will be build using Bootstrap version 4.9 since it is the most stable.

1.3.5 Functionality of the system proposed

The inventory management system web application is built using the python Django framework. The system uses Django framework, JavaScript, and CSS as the backend and python as the front-end. The system enables the users to check the products, manage the suppliers, check the categories, managing the orders and makes the modifications of them easy.

The management level users can add, delete, edit, and delete items. This is only possible if the person logging in is an employee or manager. The system is simple and organized in a very simple manner and any user can implement it ease.

1.3.5.1 Features of the system.

1. **Login function:** The login function is used for the admin/manager and employees to login to the system. The admin/managers have the privilege to add new employees with their own privileges into the system
2. **Add item category:** This function allows the employees and manager/admin to create a category for the items which makes it easier to group them since there are different types of categories for building materials such as tools, paint etc.
3. **Add product:** This function allows the users to add a product by name, category and by the date it was added.

4. **List of products:** The list of all products added appears here. It also has some modules inside it such as:
    - **Update product:** This allows the users to update the information previously input into the product list page
    - **Stock in:** This allows the users to add more stock based on the name and category, cost price of the item, percentage of buying the item date the stock came in, quantity of the stock, selling price of the item
    - **Stock out:** This allows the user to custom add the stock that leaves the inventory based on the product category and name, quantity of the stock leaving and date the stock will leave.
5. **Add customer:** This allows the users to add and save the personal information about the customers which include name, second name, personal pin of the customer, city, mobile number, alternate mobile number, nationality, and their complete address
6. **Customer list:** This function helps view the list of customers along with some more function such as:
    - **Update customer details:** Updates the customer details based on the information used to add them
    - **Customer ledger:** add the credit and debit information of customers based on the date and financial details.
    - Customers can be searched based on name and pin
7. **Banking system:** This function is used to store the bank information of the company based on the bank name, branch of the bank, account number and date
8. **Bank List:** This is used to update bank details
    - Banks can be searched based on name and account number
9. **Add expense:** This function is used for adding any expense that the company faces along with the amount required, date of the requirement and details of the requirement.
10. **Expense List:** The expense list shows all the expenses and the function to update them or delete them

11. **Sales function:** This is the most important function of the system. The item can be added for sales purposes.
12. Once the name of the item is selected from the drop-down menu, the stock, price, and the total amount will already be calculated.
13. **Billing function:** The billing function is used to create the invoice.
    - There are two options to choose from the billing function, cash, or installments. Once the received amount from the customer ledger and in cash is received.
    - There will be an option to create invoice. This allows the employee to view the invoice, save it in pdf format and print it.
    - A new customer can be created from the page too while billing.
14. **Invoices list:** This function shows the list of all the invoices based on the date, invoice number, customer name, type of payment, quantity of the order, subtotal, and grand total.
    - View invoice: view the invoice in a printable format which further allows to print invoice or create another invoice.
    - View payment type: View the installments and also delete the installments
    - Search: Invoices can be searched using the name, invoice number and date added.
15. **Logs:** This function is used to show the logs of the sales based on daily basis and monthly basis.
    - **Daily basis:** shows the item name, number of invoices for the item, and the stock that has left the company. This can be searched using the search function based on the name and can be filtered using the date.
    - **Monthly basis:** Shows the sales on monthly basis based on item name, number of invoices for the item, and the stock that has left the company. This can be searched using the search function based on the name and can be filtered using the month
16. **Report:** This shows the monthly report based on the sales per month and date in a visualized manner. It also gives an overview of the total sales in the current month

based on date, total customers, total quantity of orders, grand total, total paid in cash.

17. **The admin panel:** The admin has various function from the backend perspective. They can do all the function that an employee can but also give special privileges to the user of the system such as give them some special user permissions like making employees admin, adding new users to the system.
18. **Low quantity notification:** when an item in the inventory is low, there will be a notification saying the stock is less
19. **Logout:** after using the system, users can logout

1.4 Stakeholder

1.4.1 Definitions

Stakeholder: In a software development project, a stakeholder is an entity that is involved in a project or affected by it. Stakeholders can exist outside the development circle or outside it. The primarily focus on the finished end of the project. The stakeholder of a project can be involved directly or indirectly

External stakeholder: External stakeholders are not involved directly with a project. They affected by the outcomes of the project.

Examples of external stakeholders are creditors, suppliers, and the public entities

Internal stakeholder: The interest of internal stakeholders aligns inside the project. They have a direct relationship to the project development.

Example: Employer, manager, investor etc.

1.4.2 Stakeholder analysis

S.no	Stakeholder type	Stakeholder	Stakeholder Impact
1	External stakeholders	• College • Customers • Internet provider	The outcome of the web application is affected indirectly by the external stakeholders based on the requirements. They are not directly associated with the project but are related.
2	Internal stakeholders	• Employees • Admin/manager • Developer	The internal stakeholders are directly associated in the project

1.5 Interfaces

1.5.1 External interfaces

The web application will be compatible with most of the common internet browsers such as Google Chrome, Apple Safari, Microsoft Internet Explorer, Opera and Mozilla Firefox

Minimum Requirements

- A computer
- Web browser
- Windows or Mac operating system that is updated to modern standards
- Internet connection

1.6 Aims and objectives

1.6.1 Stating Aim

The primary aim of the project will be designing and developing an inventory material management system for, and organization called Abu Abeer al Gawabry trading.

Abu Abeer Al Gawabry trading currently has a system that is outdated and can only do basic

functions like adding, updating, deleting items along with basic calculation functions.

The project will give the company a new system that will be used to add item category, add product, view list of products, edit the list of products with more functions such as stock in, stock out, add customer information, update customer information, banking system, expense system, sales, stocks, and reports.

1.6.2 Objectives

The proposed system will make everything better and will give the system

- To develop an inventory management system that is simple and easy to use.
- To make a secure system that can store personal details of customers, banks ledgers and payment systems.
- Develop a web application to monitor the stock-in and stock-out in an efficient manner.
- To generate a report based on the sales and produces monthly sales logs

1.7 Contribution of the project

The proposed system will make the workflow of the company efficient and will be beneficial for the staff, manager, and the customers to do better business and save a lot of time. This will motivate more companies to automate their process and make the task performed on a day-to-day basis easier and reduces workload for the employees and managers.

The project will provide the company with a new system that will be used to add item categories, add products, view product lists, edit product lists, and perform other tasks such as stock in, stock out, add customer information, update customer information, banking, expense management, sales, stocks, and reports.

1.8 Report outline

In chapter 2, there will be a literature review based on the selected topic. Five similar applications that already exist will be critically analyzed using CU Harvard style referencing and finally a feasibility analysis will be conducted.

Chapter 2 will talk about the background of the project. It will give some insight into why the project is being initiated. It also contains similar systems that are already in use along with their advantages and disadvantages along with comparison. It also has a literature from few articles based on inventory management and VSCode. Feasibility analysis is also done in this chapter along with the explanation of different types of feasibility with respect to the system

Chapter 3 contains the analysis of different methodologies that are suitable for commencing forth with the project. Different methodologies will be analyzed with SWOT analysis and comparisons will be done. Chapter 3 will also contain a different data visualization to prove that the project is required. It will also contain different diagrams such as use case diagram, sequence diagram, network diagram, context diagram etc. Screenshots of the initial prototypes of the system will also be provided

Chapter 4 is based on designing the system. This will contain use case diagrams and sequence diagram of the system being implemented along with screenshots

Chapter 5 is about developing and evaluating the system. This includes information about the system and screenshots explaining what each page does. This also includes testing the system for security issues and flaws.

The final chapter is chapter 6. This includes conclusions and recommendations of the system. It will contain the practicality of the system and recommendation of similar system for future use. Ethical impacts of the system will also be included. Sustainability aspects of the project will also be included

Chapter 2: Literature Review and Feasibility Analysis

2.1 Background of chapter

The project focuses on providing the selected organization a very simple method to analyze the inventory they have along with maintaining customer and bank ledgers. The proposed system is much more effective for storing and retrieving data which makes it less prone to data manipulation.

The chapter contains a list of similar existing system that already exist in the modern world. They will be analyzed based on the advantages and disadvantages.

This chapter will explain why the project was started in the first place. It also includes similar systems that are now in use, as well as their benefits and drawbacks, as well as a comparison. It also includes a bibliography of a few works on inventory management and VSCode. This chapter also includes a feasibility study as well as an explanation of the various types of feasibility with relation to the system.

2.1.1 Structure of the chapter

This chapter focuses more on the literature review and feasibility analysis of the system. Section 2.2 will provide information on five similar systems that are currently in use along with citations. Section 2.3 will include literature reviews that are available based on the software's required for the system and the relationship of the system to the project will be explained. Section 2.4 is the feasibility analysis which will talk about the hardware and software requirements and the additional costs.

2.2 Similar systems

1. Silver Inventory system

Silver Inventory System is an internationally used application, particularly by small to medium-sized businesses in need of a mid-range, cost-effective independent stock application with a reasonable set of features. Silver Inventory System is a complete and cost-effective application. It features a user-friendly, easy-to-understand UI that may be utilized without the need for training. The system employs a point-of-sale panel that allows for real-time purchases and transactions. User access functions are available in the security system and can be managed based on access level or role. Unauthorized users are prevented from gaining access using secured access. The user has the option of exporting reports as Excel spreadsheets. In addition, it wires charges for ordered products and purchase orders while holding inventories for sales orders.

2. Zoho inventory management system

Zoho is one of the most well-known distributors of small-scale business applications and solutions. e-Commerce enterprises, online sellers, retailers, wholesalers, and small and medium-sized organizations can all benefit from it. Zoho has 45 Software as a Service (SaaS) products that help manage businesses from start to finish.

Advantages and disadvantages of Zoho inventory management system

Advantages	**Disadvantages**
Has varieties of distribution functions	Forecasting is unavailable
Can track inventory in other warehouses	No bill and ID support
The system also provides trading and purchasing sorting capabilities.	Some functions are available in certain countries
Supports tags and barcodes	Supports FIFO costing
Has a built-in marketing took	Makes the items placed first a priority over the subsequent ones

Table 3: Advantages and disadvantages of Zoho

3. Sortly inventory management systems

Users can visualize their inventory by using images rather than words on an accounting page to view it. Each item can have up to eight photographs. Users can also set up Auto Updates and Alarms, set up automated notifications to help you keep track of inventory levels, returns, and so forth. Include personalized labels and remarks.

Features

- Index – Ability to complete inventory to a beautiful, natural application and keep track of subtleties by adjusting Sortly to the user's specifications.
- Oversee - Update inventory by physically shifting things between kinds or using the Sortly app to scan QR codes/tags.
- Track - Gain a better understanding of where things are moving and what is being used by receiving alerts on low stocks and important dates.
- Report - Generate simple PDF or CSV reports that show current inventory status or previous movement.

Cons

One of the biggest flaws with this software is that it does not send out instant notifications. When specific items are pulled from stock, it does not alert the administrator. In addition, the application system's care should not be overburdened. Maintenance should not be too time consuming. It is difficult to locate assistance on a regular basis.

4. Cin7

Cin7 is a fully automated point-of-sale system and inventory management system designed to meet the needs of a variety of business sizes and sectors. The platform is completely cloud-based and includes cutting-edge capabilities that help you sell and distribute your products more rapidly and efficiently. It integrates all of your inventory and allows you to

manage numerous sales channels from a single platform, providing you a greater understanding of how your stores and online sales are managed across all of your locations.

Features

- Streamlines workflow: Cin7 is designed to be flexible. It ensures that stock levels match orders regardless of how or where a company stores or sells its goods.
- Financial tracking: It allows to connect goods to the most popular eCommerce apps and handle transactions through the platform.

Cons:

- Limitations for customization
- Updates keep making the software buggy
- Cost of the software keeps going up more frequently

5. **BrightPearl**

Brightpearl is a multichannel retail management solution that allows businesses to manage all of their critical processes in one location, including orders, inventory, accounting, reporting, and customer data. In addition, the app provides real-time information on client purchasing habits, profitability by SKU and channel, cash flow, inventory, and other topics.

Features

- Capable of doing more than 20,000 orders per hour
- Provides multicurrency support
- Good information about product in hand and stocks
- Real time insight of orders that are new

Cons

- Difficult to change or update single items since it works in a group of products
- Search feature is irresponsive sometimes
- Doesn’t have a good customer support system

2.3 Literature Review

Every day, people across the world engage in a plethora of sales transactions, resulting in a steady flow of significance that defines the character of our economies. Buying and selling things has been going on for ages, so it's only logical that an inventory management system has existed in some form or another. In most circumstances, sales refer to a transaction between two parties in which the customer receives things (physical or digital), services, or assets from the merchant in exchange for money.

According to Sheikh (2018), an organization's inventory is the supply of raw materials, partially finished goods known as work-in-progress and finished commodities it has on hand to meet its operating requirements. It is a significant investment as well as a potential source of wastage that must be carefully managed. A business's inventory is defined as a stock of goods kept on hand in expectation of possible demand. The minimum amount of inventory that must fall for an order to be placed to resupply a product.

Jonsson & Mattson (2016) explained that inventory management systems are crucial for cutting costs while increasing revenues, as well as meeting customers' requirements by guaranteeing that there is enough stock in the appropriate quantities, quality, and location. Inventory management systems must be implemented to ensure that inventory is appropriately handled. Inventory management systems relate to a set of policies and controls that manage inventory levels, analyze inventory that will be maintained, use raw materials for production, and deliver finished items.

People have been buying and selling items for ages; thus, inventory management has at the very least, always been in some manner. Obviously, there were no computers or bar code readers 300 years ago, but humans have always attempted to streamline the trading process by adopting new technologies. Inventory management is always expanding. The need for increasing efficiency and precision drives the evolution, as it does many other processes. Today, suppliers can track and manage inventory and shipments using real-time, cloud-based technologies.

According to a study done by Navaro (2012), Sales and Inventory System, a computer is a general-purpose device that may be configured to perform finite-set arithmetic or local

operations. Computers play a significant role in our society today due to technological advancements.

Technology has a great impact on our lives nowadays. Inventory, on the other hand, refers to the resources, unmanufactured commodities, and finished goods that are part of a business's ready-to-sell resources. Merchants had to keep a record of purchases and keep an eye on how many things were sold and how many were remained that day.

The proposed system will prove beneficial for the managers. According to a study titled “Monitoring and inventory for discovery mall” by Gaela & Libo (2009), The system will be responsible for physically counting products, inventories, and computing inventory summary, proper inventory management which will in turn reduce the difficulties of the manager in processing inventory. Which means that the system will keep track of commodity and product availability to eliminate understocking, overstocking, and running out of supplies.

Above is the brief introduction to the history and modern use of inventory management systems.

The following will explain about the software’s and languages used for the proposed system

Plainer (2020) stated in their research paper that visual studio code began as a project with the goal of creating a code editor that could be used in the web. In terms of stability and functionality, VSCode has been constantly improved.

VSCode is a dynamic software which allows integrating multiple programming language such as Python, Java, C, C++ etc. Visual Studio Code's core feature is a lightning-fast source code editor that's ideal for day-to-day use. Syntax highlighting, bracket-matching, auto-indentation, box-selection, snippets, and more are all available in VS Code, which supports hundreds of languages. You can traverse your code with ease thanks to intuitive keyboard shortcuts, easy modification, and community-contributed keyboard shortcut mappings.

According to Gore & Singh (2021), Django is an open-source web application framework written in the Python programming language. Because of its ability to develop quickly, in today's market, Django is in high demand. Building any type of application takes less time.

The notion of service-oriented computing underpins Web Services technology. Web services are standards that link and share business operations over the network, allowing applications from various vendors, languages, and platforms to communicate with one another and with customers. Web apps are applications that are accessible using a Web browser over a network and are written in browser-supported languages (e.g., HTML, JavaScript). Web apps rely on Web browsers to run and include a variety of well-known applications such as online retail sales, online auctions, and webmail (Fedahgi, 2011)

2.4 Feasibility analysis

A feasibility study is essentially an analysis of a planned project plan or method's effectiveness. This is accomplished by examining aspects such as technical, economic, legal, operational, and time feasibility.

1. **Technical feasibility:** It specifies how a product or service will be delivered, including transportation, business location, required technology, supplies, and personnel.

2. **Economic feasibility:** The period during which a break-even financial model of the business venture is developed based on all costs associated with taking the product from idea to market and achieving sales sufficient to satisfy debt or investment requirements is known as the economic feasibility step of business development.

3. **Operational feasibility:** The capacity to use, support, and accomplish the required activities of a system or program is known as operational feasibility. Everyone who produces, operates, or utilizes the system is included. The system must meet a business need to be operationally feasible.

2.4.1 Hardware requirements for the project

Category	Type	Cost
Hardware	Computer	OMR. 500
	Internet	OMR. 35
	Peripherals	OMR. 20
Software	Microsoft Word	OMR. 10
	Microsoft Windows	OMR. 20
	Python VSCode	-
	Product license	OMR. 10
Total		OMR 595

Table 4: Hardware requirements

- **Hardware requirements**

The primary requirement of the project is the computer where all the work required will be done.

Internet is also a necessity since the software will be a web application.

Peripherals such as mouse, keyboards and printers will be necessary to work with.

- **Software requirements**

The coding will be done on Microsoft windows platform using VSCode and python language. The web application can be developed on any platform as long as its compatible with python 3.10 and Django.

- **Technical Feasibility**

The system will be developed using tools like

- Visual studio code
- Python 3.5
- Bootstrap 4.9
- Django

These software's are free and easily available. The only requirement to run it is a modern-day web browser that can run web applications

- Economic feasibility

This system provides a better solution for the selected organization by adding the required necessities. The solution offered by this system will provide a better stock and inventory management system. The web application costs little to nothing make but can provide the organization with more benefits.

- **Operational feasibility.**

The system will be very easy to use. It does not have much hardware and software requirements. The system will be easy to use by everyone in the organization. The minimum requirement to operate the system is basic computer skills and a web browser.

Chapter 3: Methodology

3.1. Introduction

The primary target of the project is to provide Abu Abeer al Gawabry trading a suitable and reliable inventory management system. This system will effectively manage the inventory of the company.

The project will give the company a new system that will be used to add item category, add product, view list of products, edit the list of products with more functions such as stock in, stock out, add customer information, update customer information, banking system, expense system, sales, stocks, and reports.

To gather and evaluate user requirements to offer enough information to the researchers about what the system users want the system to do. To test and validate the developed system to verify that it meets all the system's criteria. Implementing a prototype of the designed system that achieves the system designs and gives a true taste of the actual system.

3.1.1 Structure of the chapter

This chapter contains 8 parts.

Section 3.2 will contain the SWOT analysis which will help to strategically plan the benefits and the weakness of the project.

In section 3.3, a comparative analysis will also be done in a tabular format to compare the models.

The stages of the methodologies will be explained in section 3.4 and the relativity of the project with respect to the model will be explained.

It will also contain justification of project methodologies and explanation for all of them and justifying the selected methodology.

Section 3.5 will be the introduction to project planning. Every important aspect of project planning will be explained in this section. This will include tables, figures, and diagrams

Methods of data collection and analysis of the data will be done in section 3.6. The data collection method including interviews, surveys will be justified and explained.

Section 3.7 will talk about the physical requirements to move forward and develop the system required for the project. This will be explained in a tabular form.

The project design and illustration including the prototypes will be explained in section 3.8. Screenshots of the initial stages of the project will be attached.

3.2 SWOT analysis for methodology

The meaning of SWOT is strength, weakness, opportunities, and threats. This method is implemented by project managers to analyze the criteria of the project with respect to the name. This helps in assessing the key objectives of the project and help to see if there is any room for improvement

- **Waterfall model (Linear methodology)**

Waterfall Model was the first Process Model to be introduced. It is often referred to as linear-sequential life cycle model. It is simple to understand and execute. In this model, each step is executed before proceeding on to the next, and the stages must not coincide.

The Waterfall model is the simplest SDLC methodology for software development.

This model illustrates the software development process as a methodical flow of events. This means any phase of the development process would proceed only after the previous is done. The stages in this waterfall model do not overlap.

Strengths	Weakness	Opportunities	Threats
There is no planning required	Requirements must be known prior to development	Requirements are known	The model is highly restrictive
Is easy to apply for small projects	Progress cannot be tracked properly	Technology is easy to understand	Not economic and practical
Objectives are well defined and easily understood	Customers cannot preview the product during development	Can be implemented in similar projects	Integration is only addressed at the product
Every phase is predetermined with its input and output	Not suitable for large scale projects	Definition of the project is stable	Not applicable for complex and modern projects
Non-technical and easy to adapt	Difficulty in integration	Can be ported to a new platform	Hard to reverse errors

Table 5: SWOT analysis for Linear methodology

- **Iterative model**

The iterative process begins with a small application of a small set of software requirements and continues to develop flexible versions until a complete system is built and ready for use.

The iterative life cycle model does not attempt to start with a complete set of requirements. Instead, the development begins with the specification and implementation of a small part of the system, which will then be considered for alternatives. This process has been repeated, with each model repetition leading to a new version of the program.

Strength	Weakness	Opportunities	Threats
Deployment of ongoing functions is rapid	Resource required is more	Development can be planned parallelly	Project is highly dependent on risk analysis
Results are achieved early	Management required is high	Inexpensive for changes	Cannot provide definition of the system without increments
Risk management is easy	Not suitable for small projects	Initial operation time is less	Not suitable for changing environments
Better for large projects	Product is unknown	Risks identified from each increment can be used for the next increment	Because not all requirements are acquired at the start of the complete life cycle, system architecture or design challenges may occur.
Customer feedback is available	High skilled resources required for risk analysis	Testing and debugging is easy	More complex to manage

Table 6:SWOT analysis for iterative methodology

- **Agile model**

Agile means to something that is fast or versatile. A software development method is proposed based on iterative development is referred to as a "agile process model." Agile methodologies breakdown projects into smaller iterations or sections and try to minimize planning. The scope and requirements of the project are defined at the start of the development phase. The number of iterations, duration, and scope of each iteration are all explicitly determined ahead of time.

Strength	**Weakness**	**Opportunities**	**Threats**
Gives an insight to how the final product looks	Not suitable for architecture projects	High collaboration between customer and stakeholders	Lack of predictability
Increases productivity due to iterations	Difficult to work with in large projects for large organizations	Detection of problems is easy	Difficult to pin down problems
It is adaptive	The team should be present together to work	There is more room for improvement	Keeping engagement between everyone to make it work
More flexible	Limited project planning and tracking	More transparent which helps customers and stakeholders	Record keeping is difficult
Delivery is rapid	Documentation is limited	Feedback is immediate	Doesn’t follow a strict schedule and project can go wrong if not followed properly

Table 7: SWOT analysis for agile methodology

3.3 Comparison of models under each methodology

Here the best model from each methodology will be compared and the most suitable model for the development of this project will be selected

V-Shaped (Waterfall)	Spiral Model (Iterative)	DSDM (Agile)
Feedback from user is unavailable	Feedback from user is unavailable	Feedback from user is available
Cannot be changed once the project has initiated	Can be changed once the project has initiated	Can be changed once the project has initiated
Very low predictability	Very low predictability	High predictability
Easy to understand	Average to understand	More complex
Basic usability skill required	Average usability skill required	Most used model
Cost is low	Expensive	Very expensive
No customer priority	Average customer priority	High customer priority
Resource organization is available	Resource organization is not available	Resource organization is not available

Table 8: Comparison of methodology

From the above comparison, it is fair to say that DSDM methodology will be the best model for this web application development

3.3.1 Justification of methodology

Dynamic Systems Development Model (DSDM) has been selected for the project. DSMD focuses more on the aspect of the projects including the satisfaction of the stakeholders and the customer. The project environment mandates a focus on the broader business goal as well as all elements of the solution that develops to fulfill that need. DSDM has a long history of delivering successful Agile projects in a variety of organizational situations, and has proven to be entirely scalable, functioning efficiently in small, simple organizations, big, complicated organizations, and highly regulated environments. It's also been proved to work equally well for IT and non-IT projects, such as business change initiatives.

3.4 Application of methodology

3.4.1 Stages of the methodology

The methodology of DSDM for agile software development that is supported by the user's continuous interaction in an iterative and progressive development. It's a well-defined conceptual and technological support framework, generally consisting of actual objects or software modules, that may be used to organize and develop.

The stages of the methodology are as follows

1. **Pre-project phase**

Pre-project meetings take place at the executive level, when business challenges are discovered, applications are chosen, these applications are prioritized, a budget is set aside, and team building begins.

The pre-project phase guarantees that only the most acceptable projects are initiated and set up appropriately. The first project planning for the feasibility study is completed once it has been confirmed that a project will go forward, that finance is available, and so on. The feasibility study kicks off the project properly.

The feasibility and business studies are carried out one after the other. They provide the ground rules for the remainder of the development process, which is iterative and incremental, and they must be finished before any additional work on a project can begin.

2. **Project life cycle phase**

This includes

   a. **Feasibility study**

The feasibility of developing the application is evaluated during this phase, and decisions are taken appropriately. It considers the available team, the budget, the required functionality, and the potential of implementing the functions within the resources available. This study's output includes a model (or a prototype) and reports that detail how all of the feasibility criteria were satisfied.

b. Business study

The comprehensive business analysis of the intended system is completed in this phase. The system's information needs are identified, and the business requirements are described at a high level. Once this is completed, the desired system's fundamental architectural structure is ready. The team investigates the project's business elements.

3. Functional model iteration

This is one of the life cycle's two iterative stages. The major goal of this phase is to iteratively create the prototype and obtain feedback from users to extract the intended system's needs.

The requirements obtained in earlier steps are incorporated into Functional Models. To achieve so, functional prototypes of the requirements are generated to show the user how the program will perform.

The prototype is refined by demonstrating it to a user, receiving feedback, and making modifications as needed.

4. Design build and iteration

The major goal of this phase is to enhance and improve the functional models created in the previous phase so that they completely meet the demands of the user. To do this, prototypes of the initial model will be created.

The software components created during functional modeling are fine-tuned until they fulfill a certain standard. The result of this step is a thoroughly tested system that is ready for use.

5. **Implementation**

Ultimately, in the implementation phase, users check that the prototypes meet their criteria, and the implementation, as well as the training of future users so they can operate the program, takes place.

End-user feedback is obtained on a regular interval to maintain that business expectations are satisfied and that the perfect solution is provided to meet the end users' needs.

3.4.2 Describing the application of the methodology to the system of the current project

1. **Feasibility and business study**

A feasibility study was conducted to ensure the project is viable and to justify that the project is economically feasible.

All the hardware and software requirements of the project were evaluated alone with a cost estimation study and an operational feasibility study.

The results showed that the system is very feasible

2. **Functional model iteration**

Initial prototypes of the system have been created and feedback from the users have been taken in the form of questionnaires, surveys, and interview.

3. **Design build and integration**

The initial prototypes that have been created and provided to the organization. Continuous review of the system

4. **Implementation**

The project is implemented after following all the criteria which make sure the web application is ready for deployment after making sure it passes all the criteria required

3.5 Project Planning

3.5.1 Introduction to project planning

A project plan, also known as a project management plan, is a document outlining how a project will be followed through, monitored, regulated, and finalized. This document outlines the program objectives and scope, and it serves as an official guideline for the project team, the larger organization, and stakeholders.

It is a compilation of multiple diverse documents that is generated throughout the project planning phase. It's more than just a schedule or a task list, although both elements are included. The project management plan is formally established at the beginning of the project and then adjusted as necessary throughout its lifetime.

3.5.2 Communication plan

A communication plan is a framework for how conveying important information will take place with ongoing project information to key stakeholders in project management. The communication strategy will assist the team in deciding who should receive warnings and when to inform project participants. The team will explain which channels participants should use and when in the communication strategies, as well as how often different information should be provided and who should respond to each channel.

Goal	Tool of communication	Audience	Frequency
Project review	In person meeting, email	Project supervisor Stakeholders	Weekly
Testing project	On campus meeting	Project manager Project supervisor Stakeholders	Weekly
Document submission	Online	Project supervisor	Weekly
Review the system	On campus meeting	Project supervisor	Weekly
Project updates	Email	Project supervisor	Weekly

Table 9: Communication plan

It is very important to approach and communicate with stakeholders regarding information about the project. This makes sure the project is viable and makes sure it can be deployed without any setbacks

3.5.3 Resource plan

Project managers employ a resource management strategy to manage their resources. A resource management strategy is often used to handle the most crucial resource in any project

Each resource is allocated a job and duties when the project manager creates a resource management strategy. It is vital to stress, however, that team members should be included in the planning process. This permits resources to be engaged and can bring project insight that might otherwise go unnoticed by someone who isn't a subject matter expert.

Resource	**Type**	**Functionality of resource**
Hardware resource	Personal Computer	Backbone of the project. used for every important and necessary task
	Internet	Required to find necessary information to make the project possible
	Storage space	Required to save all the necessary files
Software resources	Microsoft Visual studio code	For making the project and coding purposes
	Microsoft word	Required to make the report
	StarUML	Tool used to make system diagrams
	Python 3.5	Front-end coding
	Microsoft PowerPoint	To create presentation

Table 10: Resource planning

3.5.4 Risk management plan

Risk is defined as an occurrence that has a chance of occurring and this may have a direct or indirect effect on a project if it ever does occur.

Risk management is a continuous process that can last the length of the project. It entails risk assessment planning, diagnosis, evaluation, measurement, and schedule predictability. Many of these procedures are adjusted as attack vectors are recognized over the lifecycle of the project. The aim of risk management is to reduce the probability and impact of occurrences that are unfavorable for the project. On the other hand, any incident that has the potential to add value should be taken advantage of.

Risk	Type of risk	Chance of occurring	Impact	Mitigation strategy
Project Failure	Project deferral risk	Low	High	Implement better project planning strategy
Project exceeds cost	Cost risk	Low	Medium	Examine cost benefit again and make sure it doesn't exceed requirements
Time limit exceeds	Time risk	Low	High	Follow proper time management
Technological risks	Technological Risk	High	High	Use up to date software's
Low client involvement	Stakeholder risk	Low	High	Organize meetings with client

Table 11: Risk management plan

From the risk management plan, it is evident that the proposed web application is a low-risk project.

3.5.5 Work breakdown structure (WBS)

A typical productivity tip for making the work more reasonable and approachable is to break it down into smaller tasks. The Work Breakdown Structure (WBS), which is one of the most important project management tools, is the instrument that utilizes this approach for projects. It incorporates scope, cost, and schedule baselines on its own, guaranteeing that projects are in track.

Task Name	Duration	Start	Finish	Predecessors
Design and implement a inventory management system	**113 days**	**Fri 10/15/21**	**Tue 3/22/22**	
Pre-project phase	**12 days**	**Fri 10/15/21**	**Mon 11/1/21**	
Project overview	3 days	Fri 10/15/21	Tue 10/19/21	
Scope of project	2 days	Wed 10/20/21	Thu 10/21/21	3
Aim of project	1 day	Fri 10/22/21	Fri 10/22/21	4
Objective of project	1 day	Mon 10/25/21	Mon 10/25/21	5
Stakeholder analysis	2 days	Tue 10/26/21	Wed 10/27/21	6
Data collection	3 days	Thu 10/28/21	Mon 11/1/21	7
Feasibility study	**16 days**	**Fri 10/15/21**	**Fri 11/5/21**	
Literature review	2 days	Fri 10/15/21	Mon 10/18/21	
Problems faced	2 days	Tue 10/19/21	Wed 10/20/21	10
Feasibility analysis	3 days	Thu 10/21/21	Mon 10/25/21	11
SWOT analysis	1 day	Tue 10/26/21	Tue 10/26/21	12
Operational feasibility	2 days	Wed 10/27/21	Thu 10/28/21	13
Cost benefit analysis	3 days	Fri 10/29/21	Tue 11/2/21	14
Risk assessment	3 days	Wed 11/3/21	Fri 11/5/21	15
Functional model iteration	**13 days**	**Fri 10/15/21**	**Tue 11/2/21**	
Proposed methodology	2 days	Fri 10/15/21	Mon 10/18/21	
Select methodology	1 day	Tue 10/19/21	Tue 10/19/21	18

Functional model iteration	3 days	Wed 10/20/21	Fri 10/22/21	19
Design build & implementation	4 days	Mon 10/25/21	Thu 10/28/21	20
Risk management	3 days	Fri 10/29/21	Tue 11/2/21	21
design build and iteration	**11 days**	**Fri 10/15/21**	**Fri 10/29/21**	
Requirement of system	3 days	Fri 10/15/21	Tue 10/19/21	
Security measures	4 days	Wed 10/20/21	Mon 10/25/21	24
Implementing the system	4 days	Tue 10/26/21	Fri 10/29/21	25
development phase	**16 days**	**Fri 10/15/21**	**Fri 11/5/21**	
Coding phase	8 days	Fri 10/15/21	Tue 10/26/21	
testing the system	4 days	Wed 10/27/21	Mon 11/1/21	28
final evaluation of system	4 days	Tue 11/2/21	Fri 11/5/21	29
Post-project phase	**9 days**	**Fri 10/15/21**	**Wed 10/27/21**	
Conclusion	3 days	Fri 10/15/21	Tue 10/19/21	
Recommendations	4 days	Wed 10/20/21	Mon 10/25/21	32
Legal matter	2 days	Tue 10/26/21	Wed 10/27/21	33

Table 12: WBS

3.5.6 Gantt chart

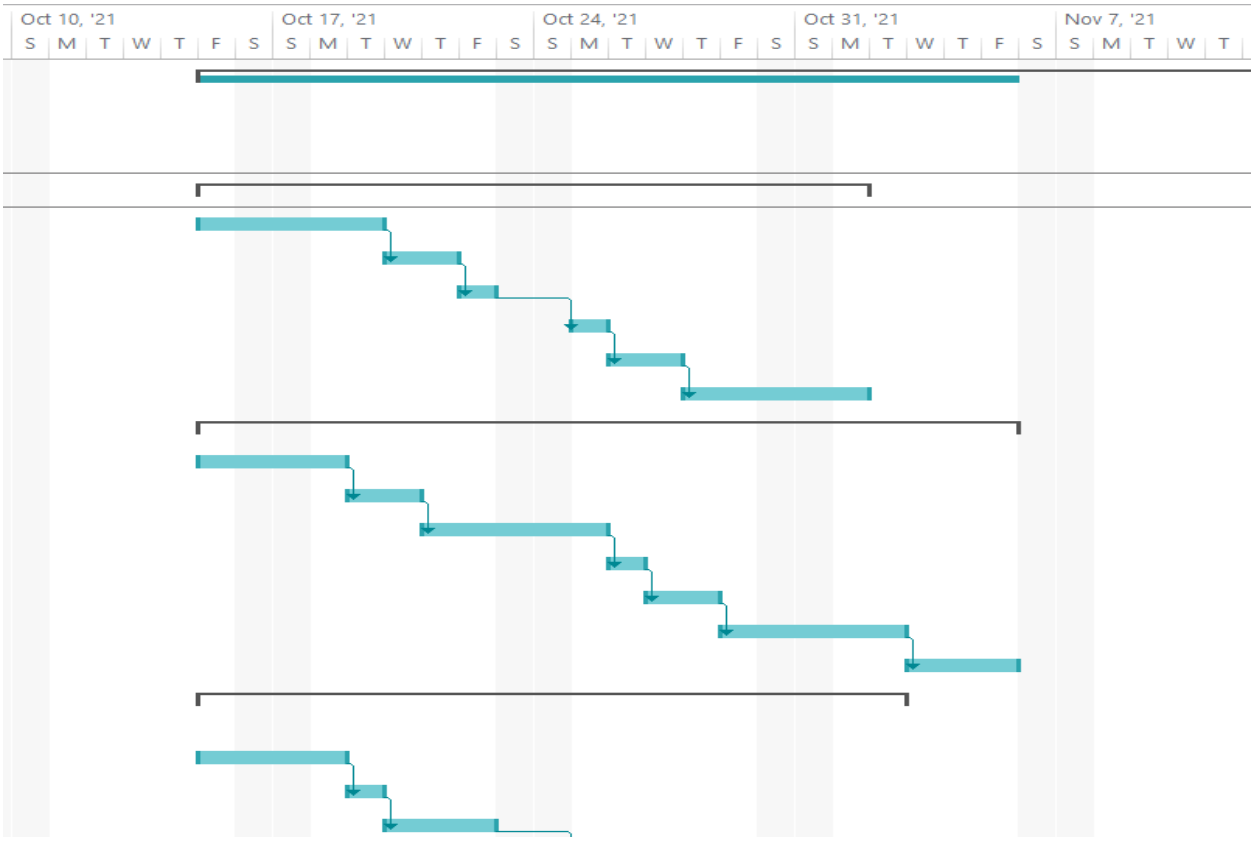


Figure 1: Gantt chart

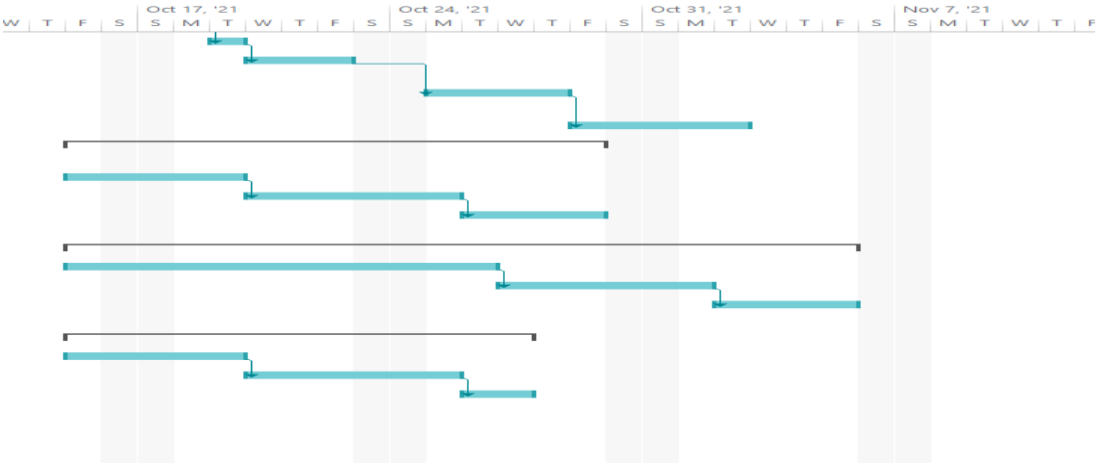


Figure 2: Gannt chart 2

3.5.7 network diagram

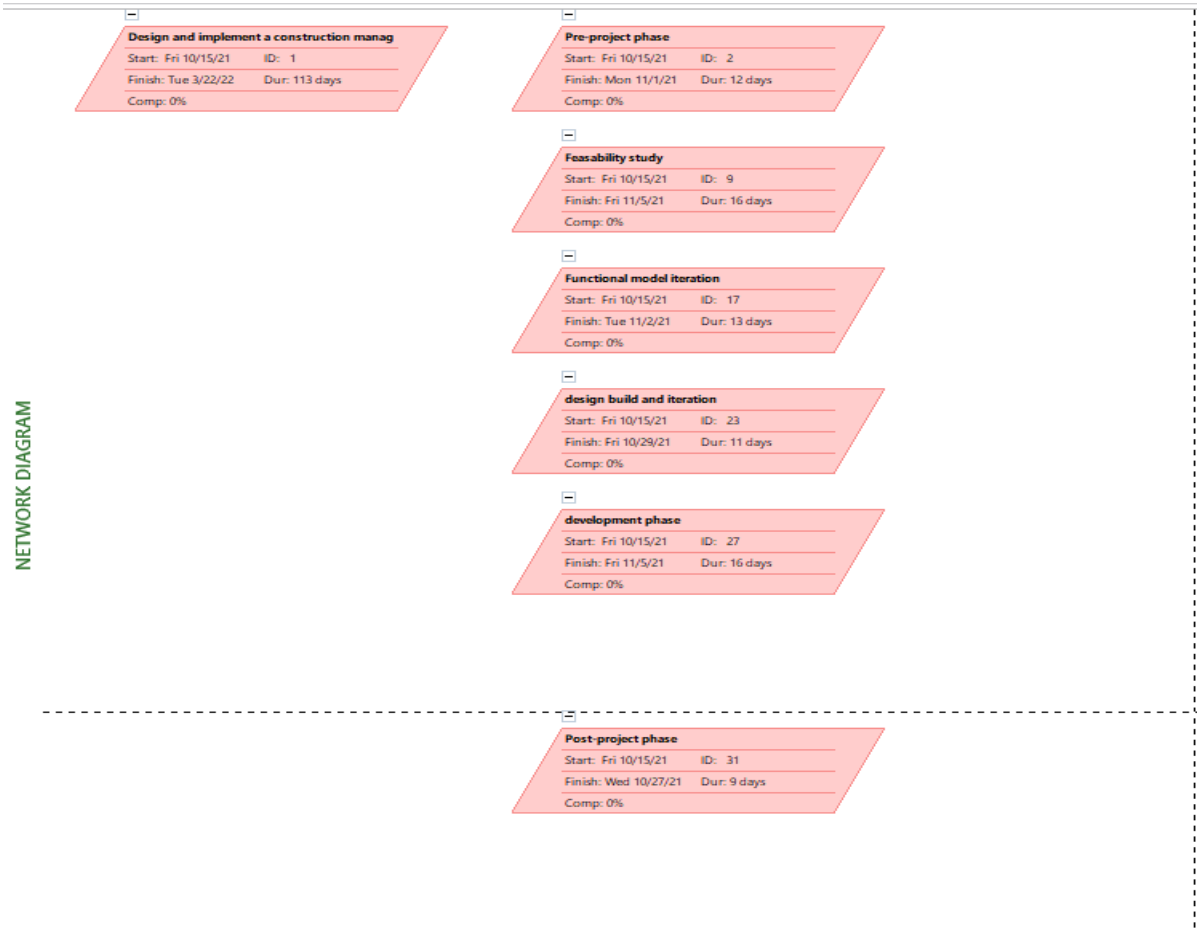


Figure 3: Network diagram

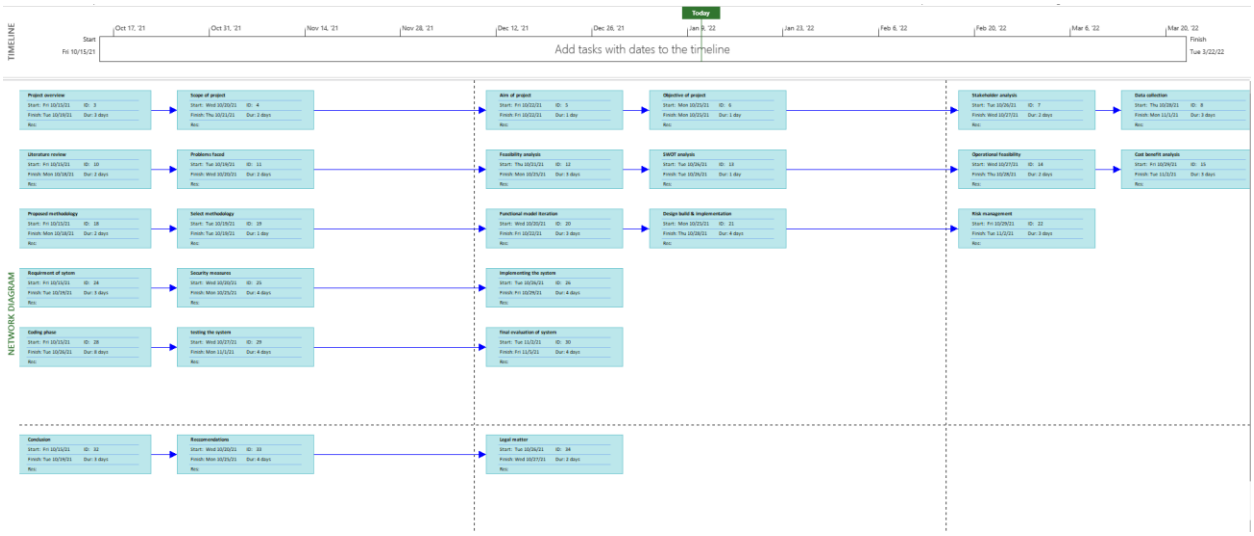


Figure 4:Network diagram continuation

3.6 Data collection and analysis

3.6.1 Describing data collection methods

The procedure of obtaining, evaluating, and analyzing precise findings for study using conventional approved procedures is known as a data collection. Based on facts collected, a researcher can analyze their hypothesis. Regardless of the topic of investigation, data collecting is typically the first and most crucial phase in the research study.

two methods of data collection were used for this study

- Interview- An interview is defined as a formal chat between two or more people in which questions are asked of a person in order to obtain the needed responses or answers (wisestep,2020)

The interview process allows the interviewee to get more information from a personal perspective of the employees and manager of the company. This helps in retrieving details that cannot be retrieved from any other data collection methods. This gives an accurate screening because the process is completely face-to-face.

- Survey- A survey is a method of research that is used to collect data from a group of members who are willingly ready to give responses.

3.6.2 Justification of data collection methods

- **Interview**

Interview helps in understanding the requirements of the client in real time and can provide much information possible

An interview is defined as a formal chat between two or more people in which questions are asked of a person in order to obtain the needed responses or answers (wisestep,2020)

Objectives of interview is to get a deeper understanding of the primary users of the system who are the managers and the employee of the system. This makes sure that the system can fulfill all the requirements that the company needs

- **Survey**

Survey helps in getting the opinions of the employees and the general people in term of what is better for the project and system that is being developed and gives an idea on how the system should be made. This makes sure the web application will full fill the requirements of the users of the system. The survey included questions about the current system in place as well as the requirement purpose of a new system. A few personal questions were also asked to the respondents. Questions asked also included the security systems of the current system and usability of the system. Most of the replies justified the implementation of a new web-based inventory system.

3.6.3 Presenting the findings

- **Interview**

An interview was conducted at the site of the company. They were not currently satisfied with the current inventory management system they had. It was primarily conducted to provide an insight on what the system was supposed to be practically and visually. Which in turn gives a better understanding of the project.

- **Survey**

Q1) The first question in the survey, 22 responses were submitted

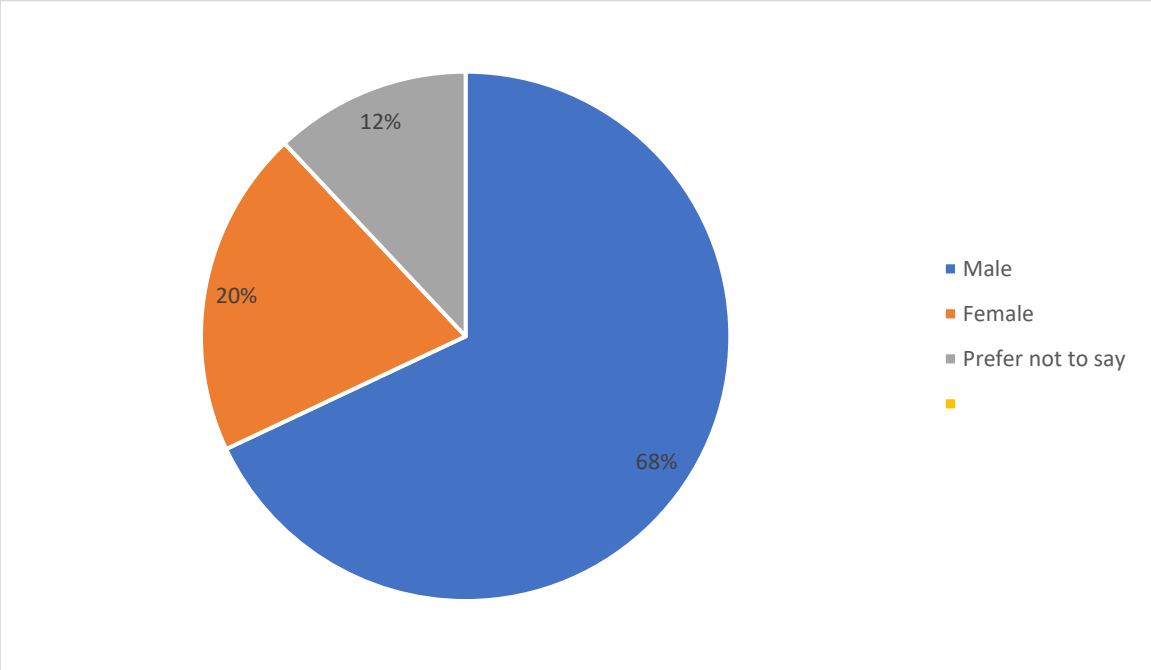


Figure 5: Survey data visualization

From the data provided by the responses, 68% of them were male, 20% were females and 12% prefer not to say

Q2) They second question asks about the experience with inventory management systems.

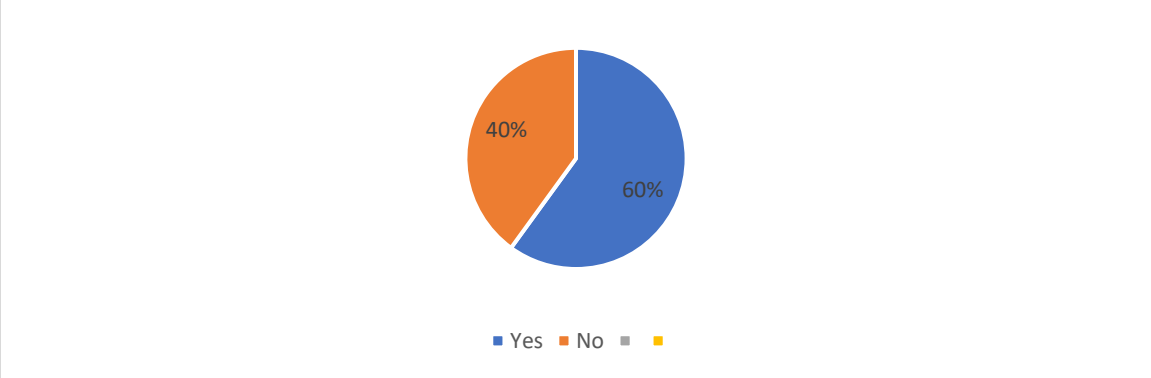


Figure 6: Survey data visualization 2

20 responses were acquired from this question, 60% of them are experienced with inventory management system whereas 40% don't have any experience with it.

Q3) The third question ask the users to rank the system in place. 20 responses were received

1- Very satisfied and 5- Unsatisfactory

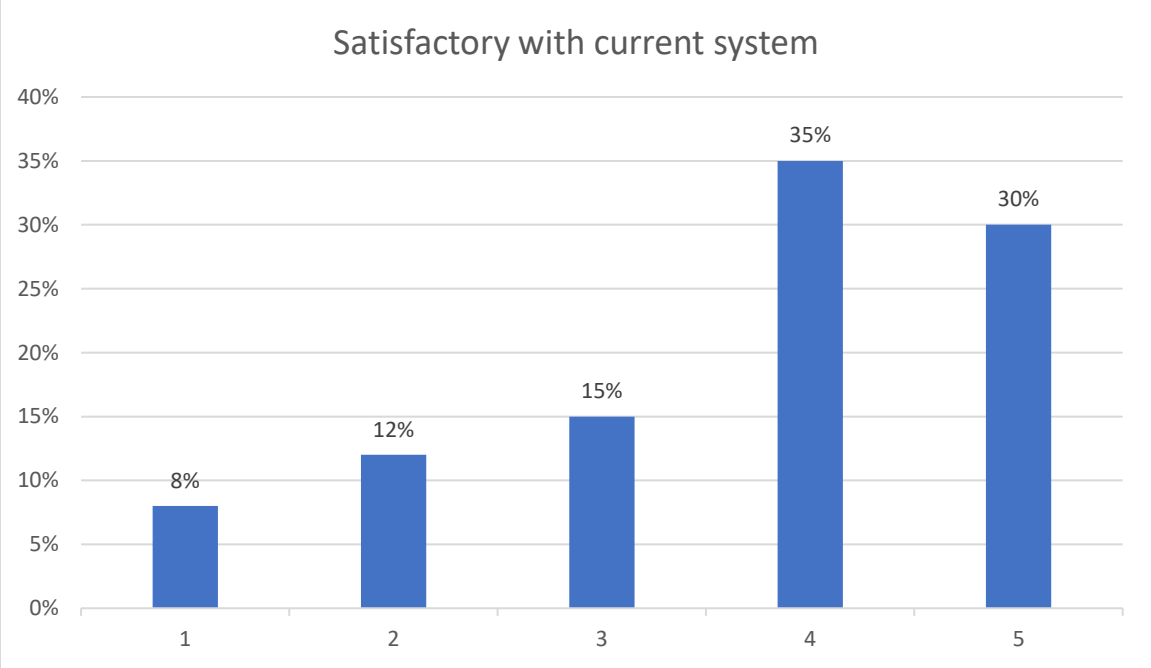


Figure 7: Survey data visualization 3

From the above information, it is very evident that more people are unsatisfied with the current system that is in place in total, 65% of the users are not happy with the current system and only 8% are very satisfied with it which is a very small number.

Q4) The fourth question is based on the security issues on the current system

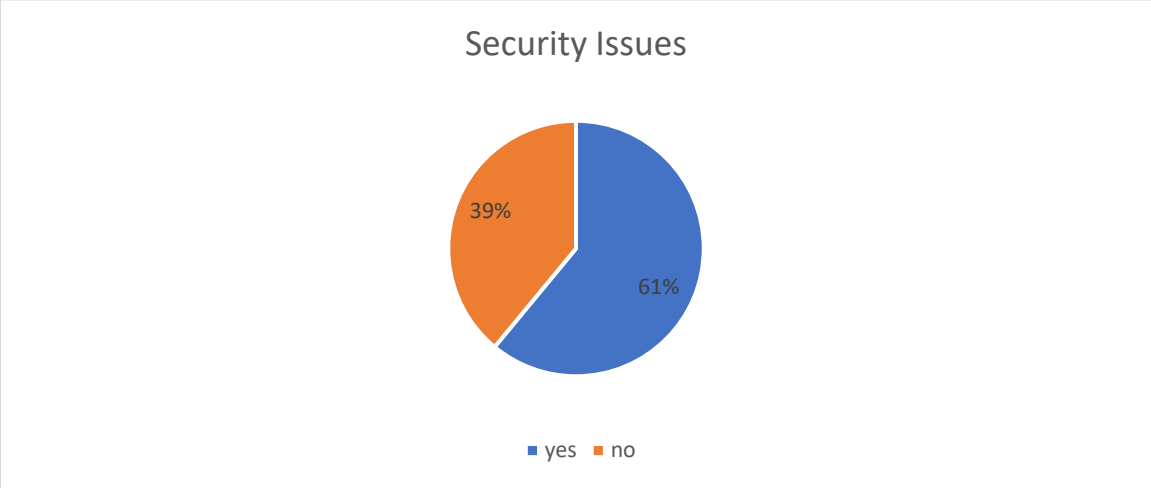


Figure 8: Survey data visualization 4

From the data gathered, 61% of the users feel there is a security concern from the current system that is used which shows it needs security improvement

Q5) The fifth question is based on the usability of the current system and to see if the users find any hardships in using it

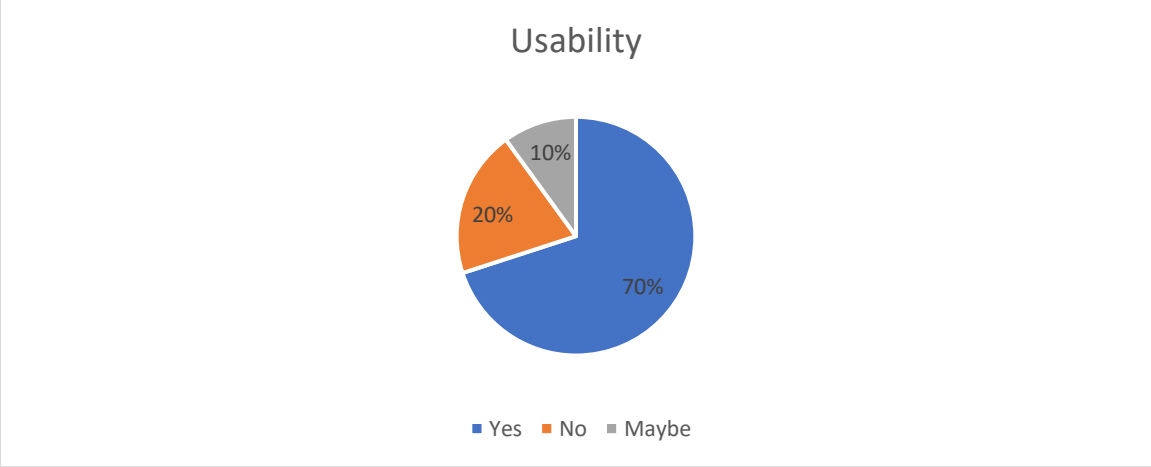


Figure 9: Survey data visualization 5

From the above results, 70% of the users find the current system complicated only 20% are satisfied with the current system

Q6) The sixth question asks the users if they want a better system which is superior than the current on

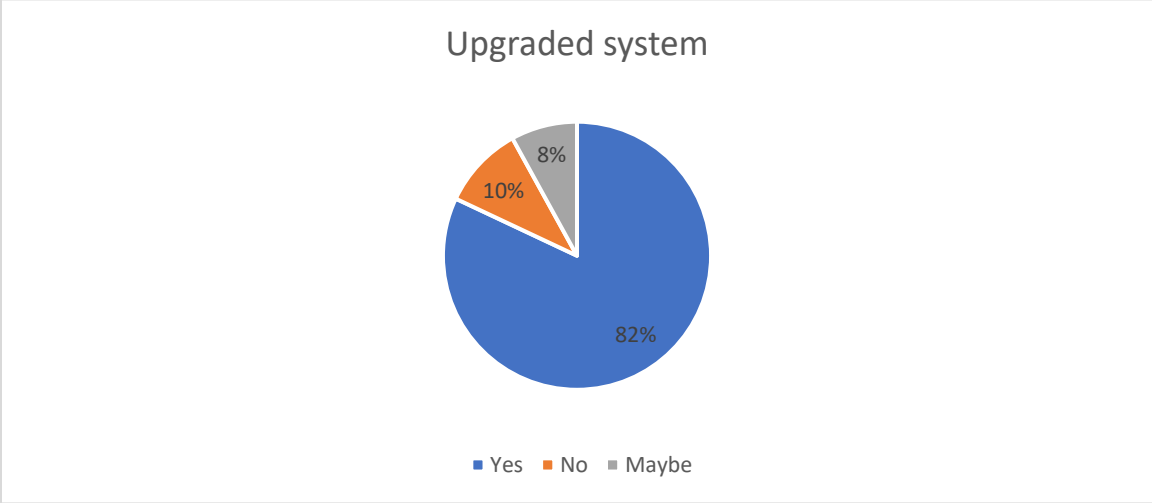


Figure 10: Survey data visualization 6

Q7) The seventh question ask about the user difficulty with the current system

1- Is very hard and 5 is very easy

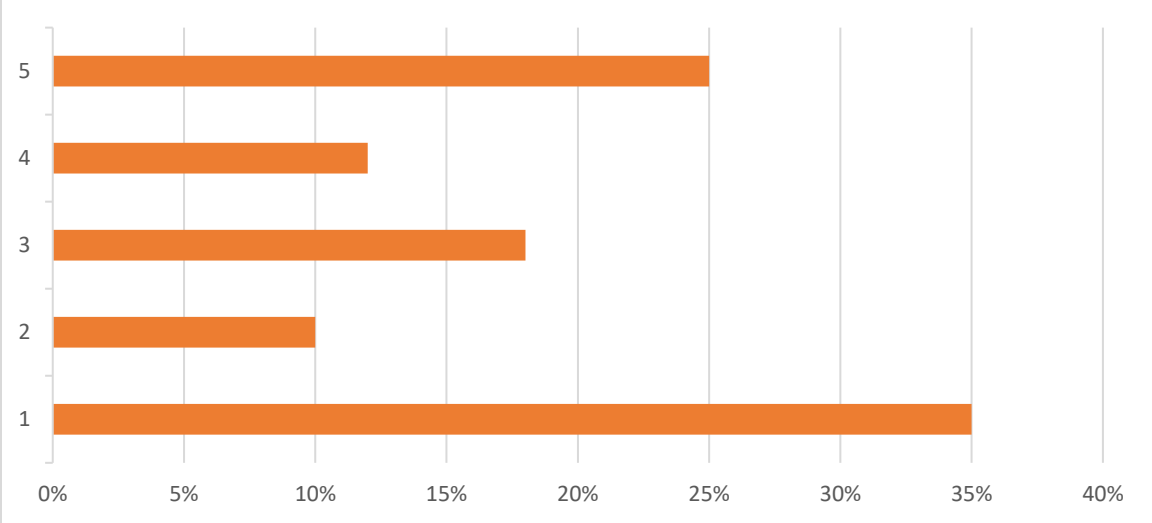


Figure 11: Survey data visualization 7

From the above visualization, majority of the user find the current system hard to use. Only 25% of the find it very easy to use but there are more people who have a difficulty in using it.

3.6.4 Interpreting the findings

From the statistics and data provided above, it is clear that the users require a better and enhanced system that upgrades the current system.

Most of the responses recorded were in favor of a new system.The proposed system will prove highly beneficial for the users of the system.

3.7 Requirements

3.7.1 External interface requirements

In order to make the web application, there a few external interfaces required.

Sr.no	Interface	Requirement
1	User interface	• Pc • Laptop • Monitor • Browser
2	Software interface	• VS Code • Bootstrap • CSS • HTML • Browser
3	Hardware interface	• PC • Hard disk • Mouse
4	Communication	• Internet

Table 13: External requirements

3.7.2 Functional requirements

A Functional Requirement is a document that explains the function that the software need to provide. It refers to the software system or a subsystem of one. A functionality is nothing but the resources, actions, and results of a software application. A calculation, manipulation of data, business activity, interactivity, or any other specific functionality that defines what function a process is supposed to perform can all be included.

Requirement ID	**Requirement: The following requirements should be able to be working for the system to be functional from all aspects**
FR-001	The system should take in the appropriate login information
FR-002	The employees and admin should be able to add categories of product, Add product, and view a list of products they can update and add stock
FR-002	Employee and admin should be able to add customers, update customer details and input customer ledgers
FR-003	Employee and admins should be able to enter bank details, add credit and debit amounts and update them.
FR-004	Employee and admins should be able to add the list of sales expenses to the system and can delete it.
FR-005	Employee and admins should be able to create and download invoice.
FR-006	Purchases made should be able to get accepted as cash or invoice
FR-007	Daily and monthly logs of stock should be shown in a tabular format
FR-008	Monthly reports should be visualized in a sales graph
FR-009	Users who are logged in should be able to logout
FR-010	Admin should be able to access their functions

Table 14: Functional Requirements

3.7.3 Security Requirements

Requirement number	Requirement Description
SR-001	Only the admin can add employees
SR-002	Django password hash should be working
SR-003	Backend which includes personal information can only be manipulated by admin
SR-004	User password should be strong
SR-005	Cannot be breached

Table 15: Security requirements

3.8 Illustrations

3.8.1 Data flow diagrams

A data flow diagram (DFD) illustrates how information flows through a process or system. It shows input data, output, storage facilities, and pathways between each destination utilizing predefined symbols.

3.8.1.1 Level 0 DFD

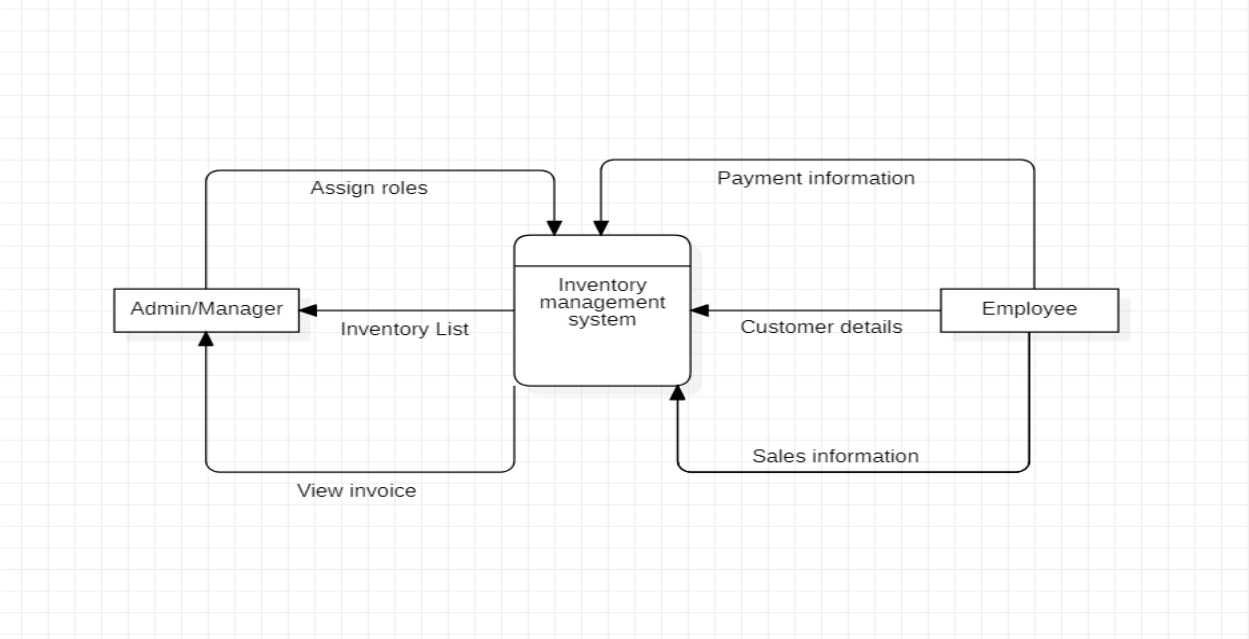


Figure 12: Context diagram/Level 0 DF

3.8.1.2 Level 1 DFD

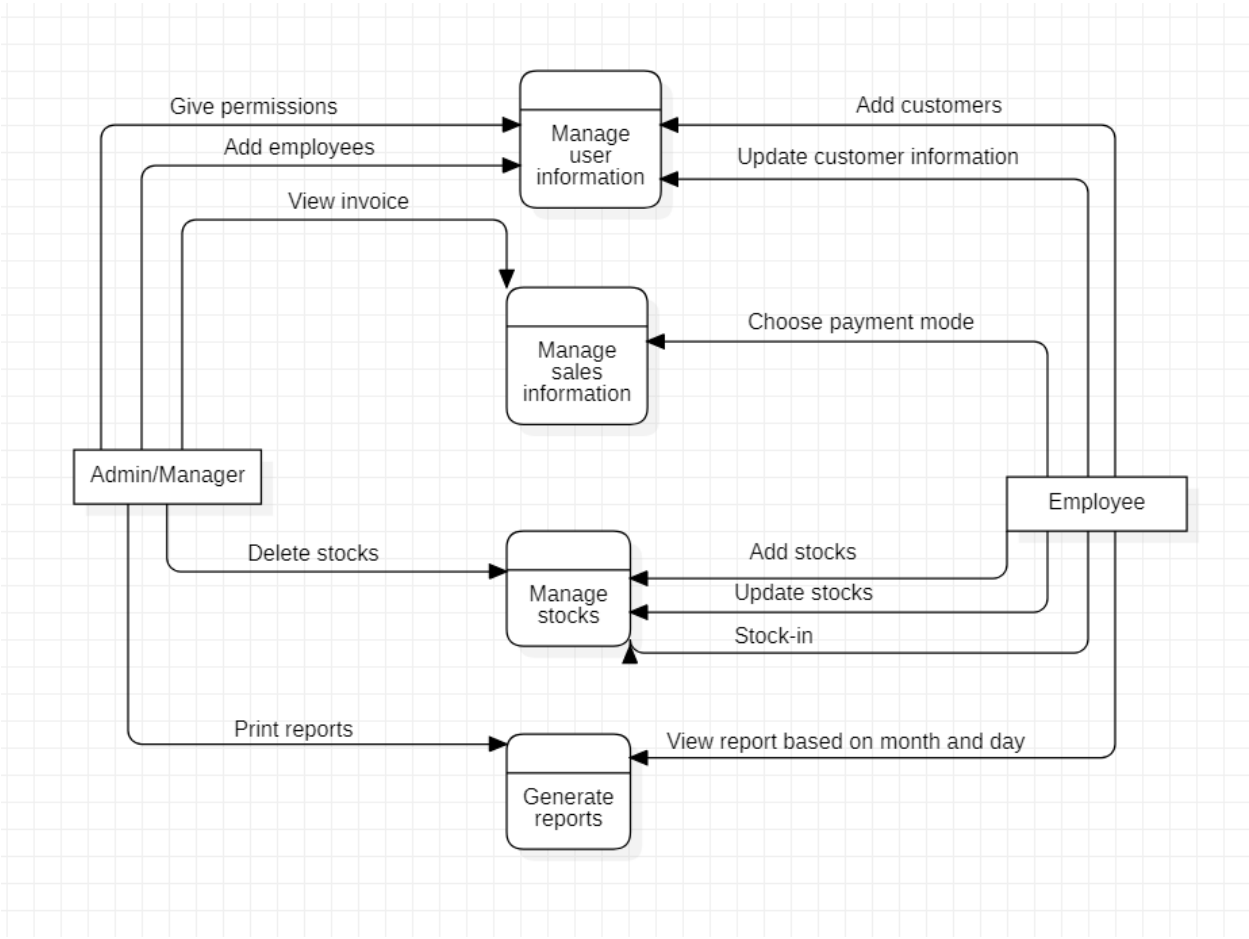


Figure 13: Level 1 DFD

3.8.3 System flowchart

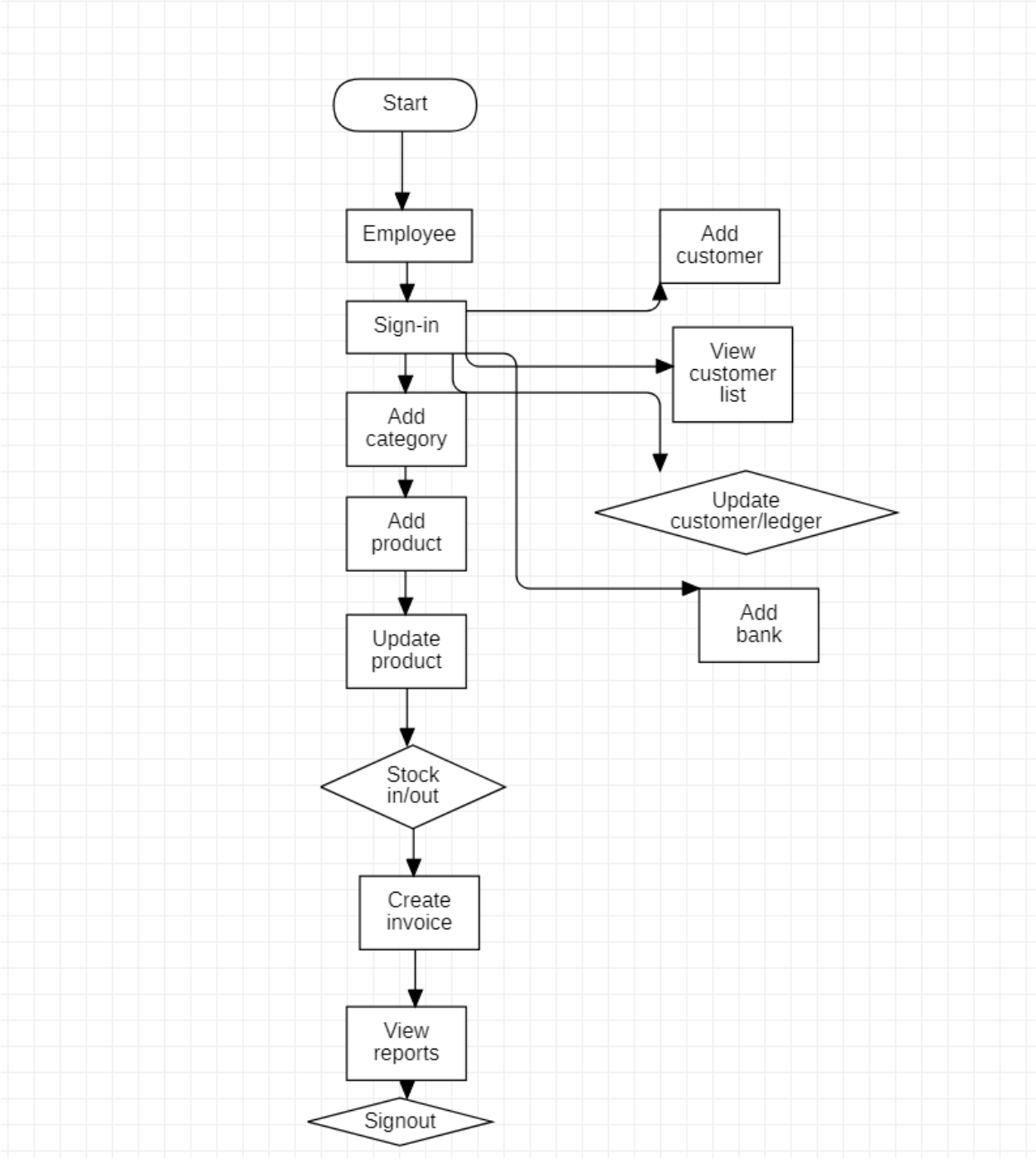


Figure 14: System flowchart

3.8.4 Entity relationship diagram

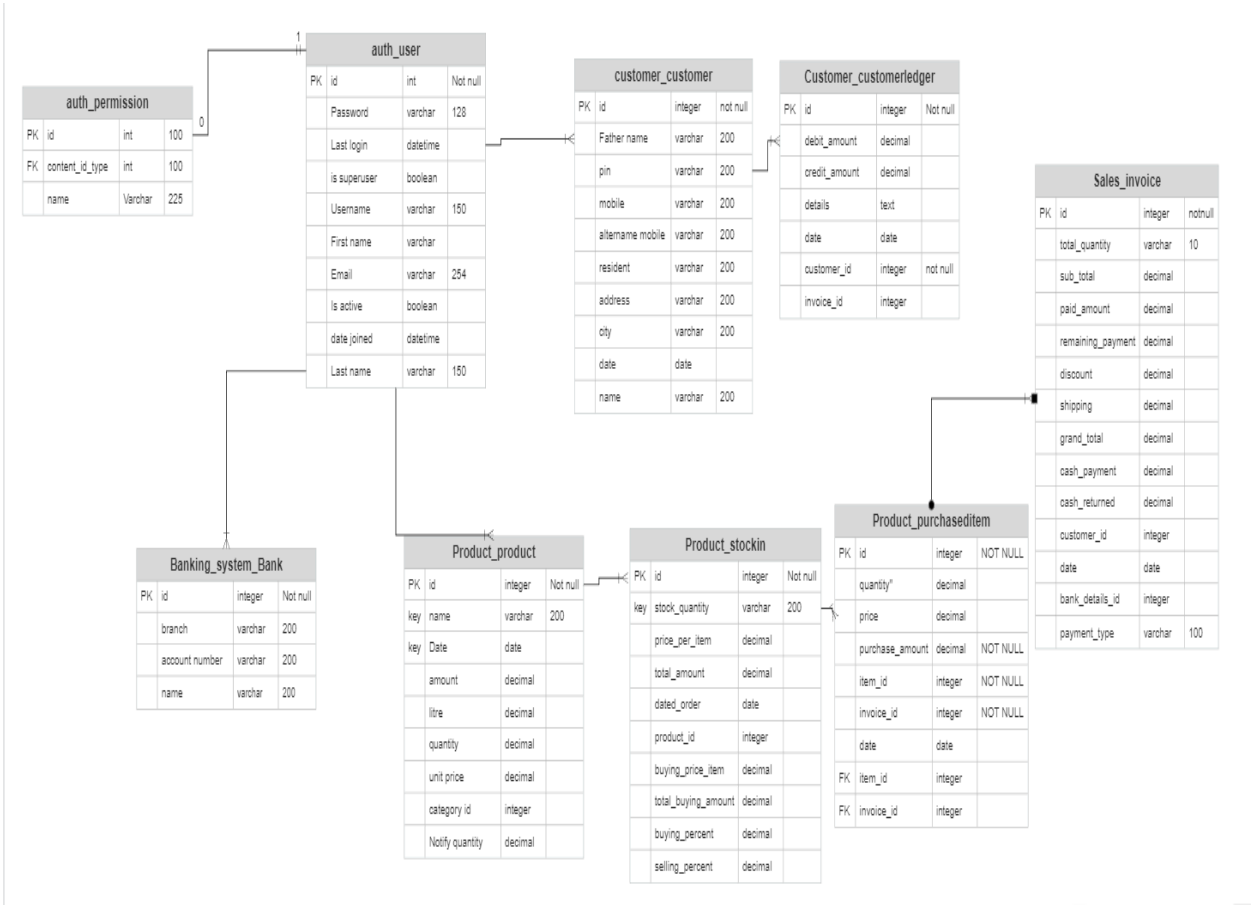


Figure 15: ER Diagram

3.8.5 class diagram

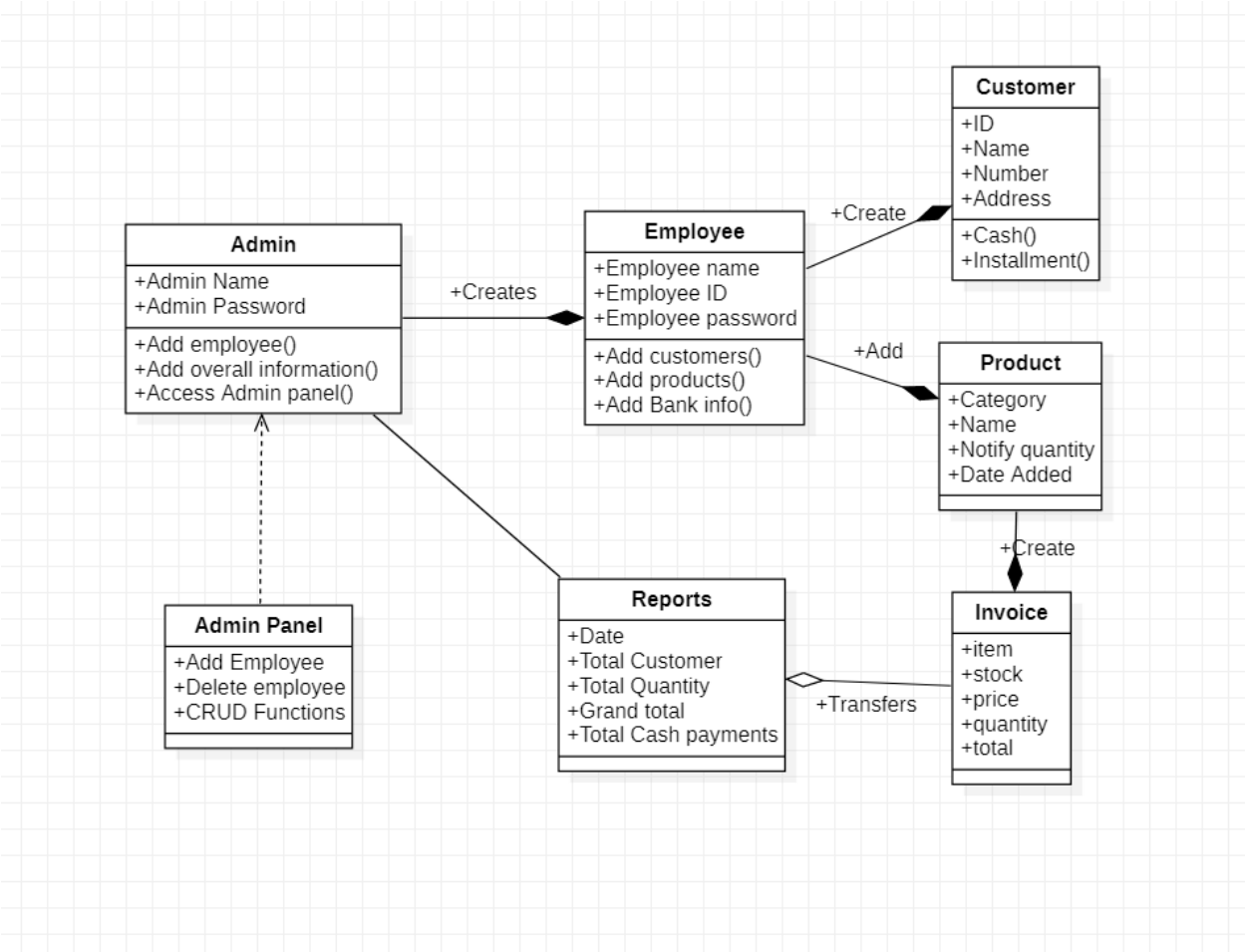


Figure 16: Class diagram,

3.8.6 Use case diagram

A use case diagram is a visual representation of the characteristics of a system and its users. It is often represented as a graphic representation of interactions between the various parts in a system. Use case diagrams illustrate the activities that happen in a system and how they flow, but they do not specify how certain events are handled.

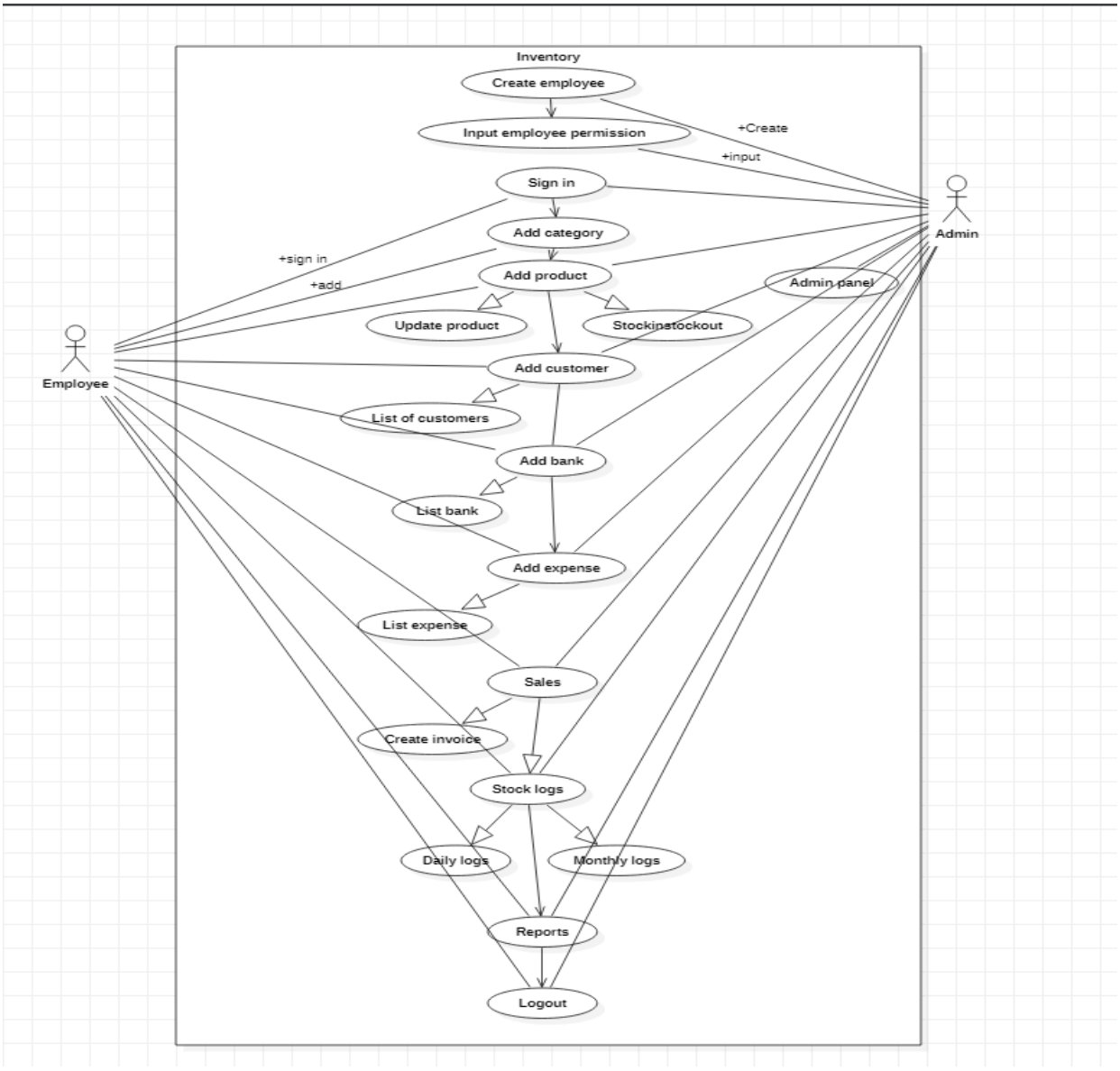


Figure 17: Use case diagram

3.8.6.1 Sequence diagram

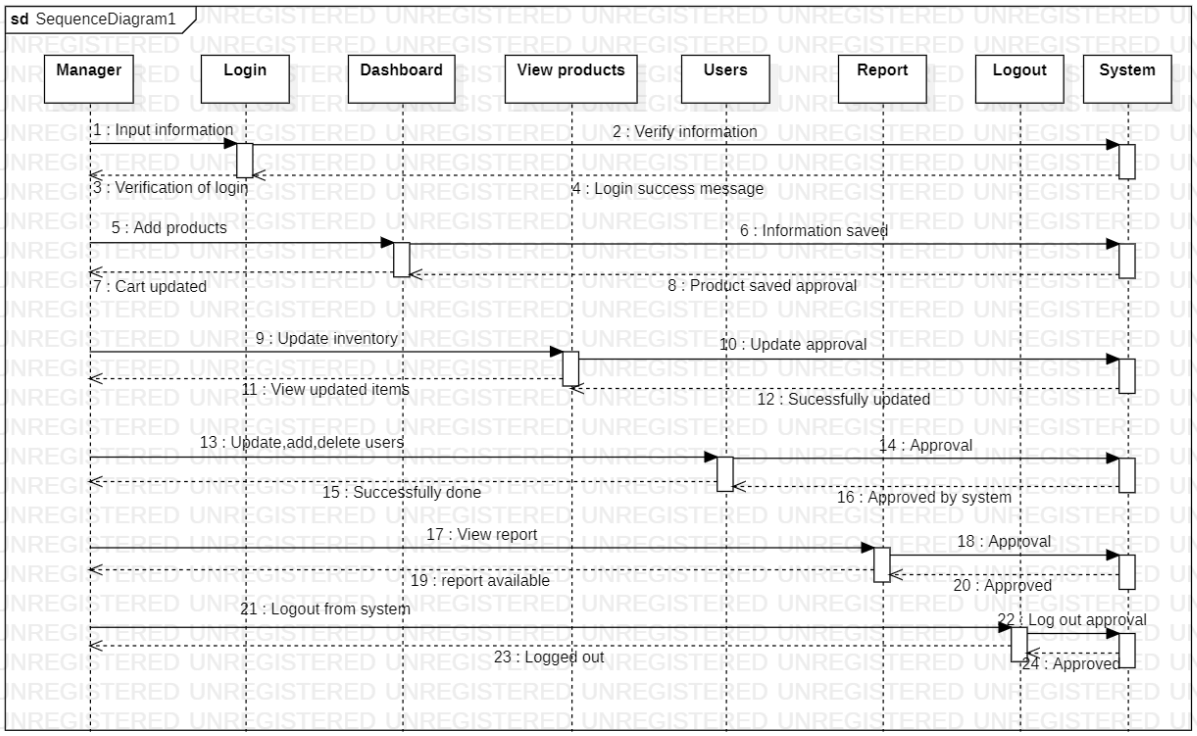


Figure 18: Sequence diagram

3.9 Initial prototype

3.9.1 Login UI

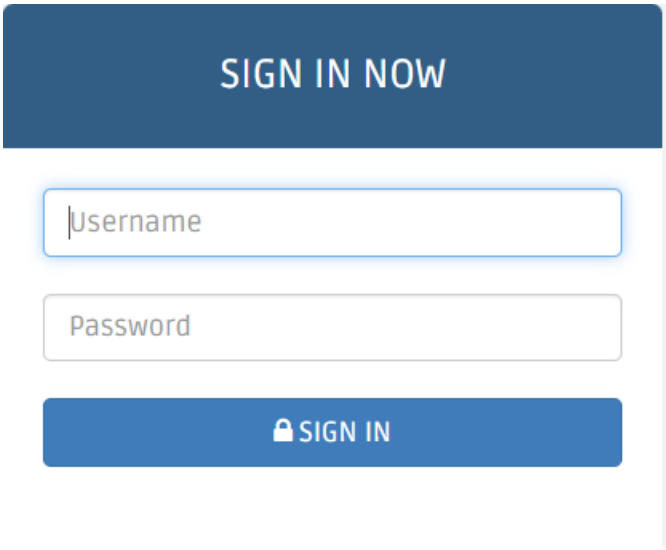


Figure 19: Login UI

This is the initial login UI. It does not contain any logos or any special features. This will be updated when the system gets an update.

3.9.1 Add category page

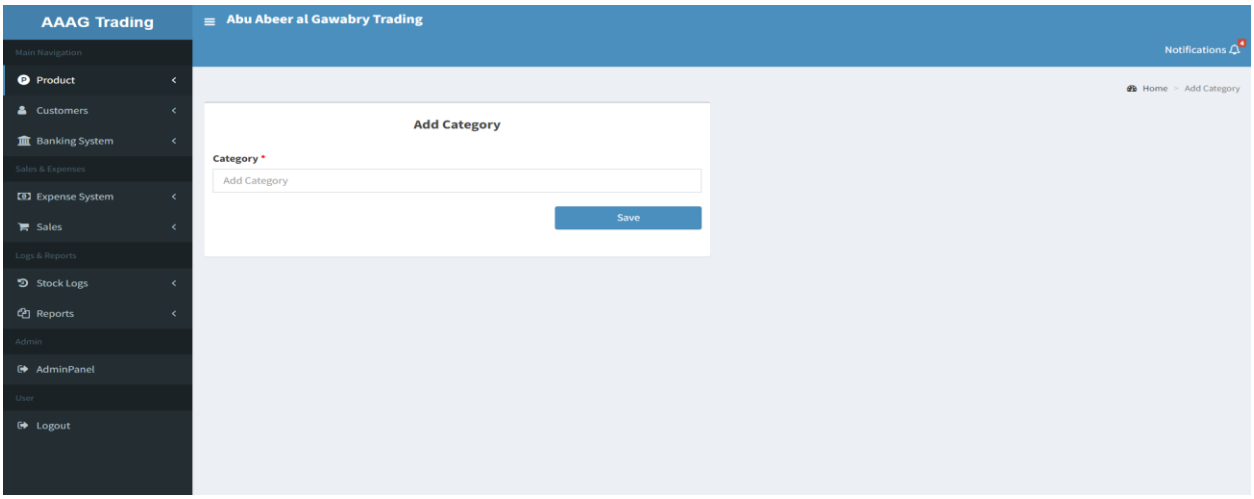


Figure 20: add category

This is the add category page. The templates do not have any CSS or animations, yet which will be added.

Chapter 4: Design

4.1 Introduction to chapter

This chapter will talk about illustrating the system which will include an updated use-case diagram with use-case specifications, test case, class diagram, sequence diagram, ER diagram.

The tables and views in the application will be explained. Methods to protect the data will also be included.

Screenshots of all the pages and explanation of each page will also be added.

The different designs of the system will be further explained based on the updated system after the system has gone through some changes

Section 4.2 will illustrate the designs, 4.2.1 contain the use case diagram of the system

4.2.2 contains the flowchart of the system along with the explanations

4.2.3 contains the sequence diagram.

The use case specification of the login function is in section 4.2.4

Section 4.2.5 contains the entity relation ship diagram which shows the relationship between different tables

Section 4.3 contains the data dictionary. It shows the structure of the tables present in the database.

Section 4.4 explains the purpose of using the diagrams

Section 4.5 contains the screenshot of each page.

4.2 Illustrating the design

4.2.1 Use case diagram

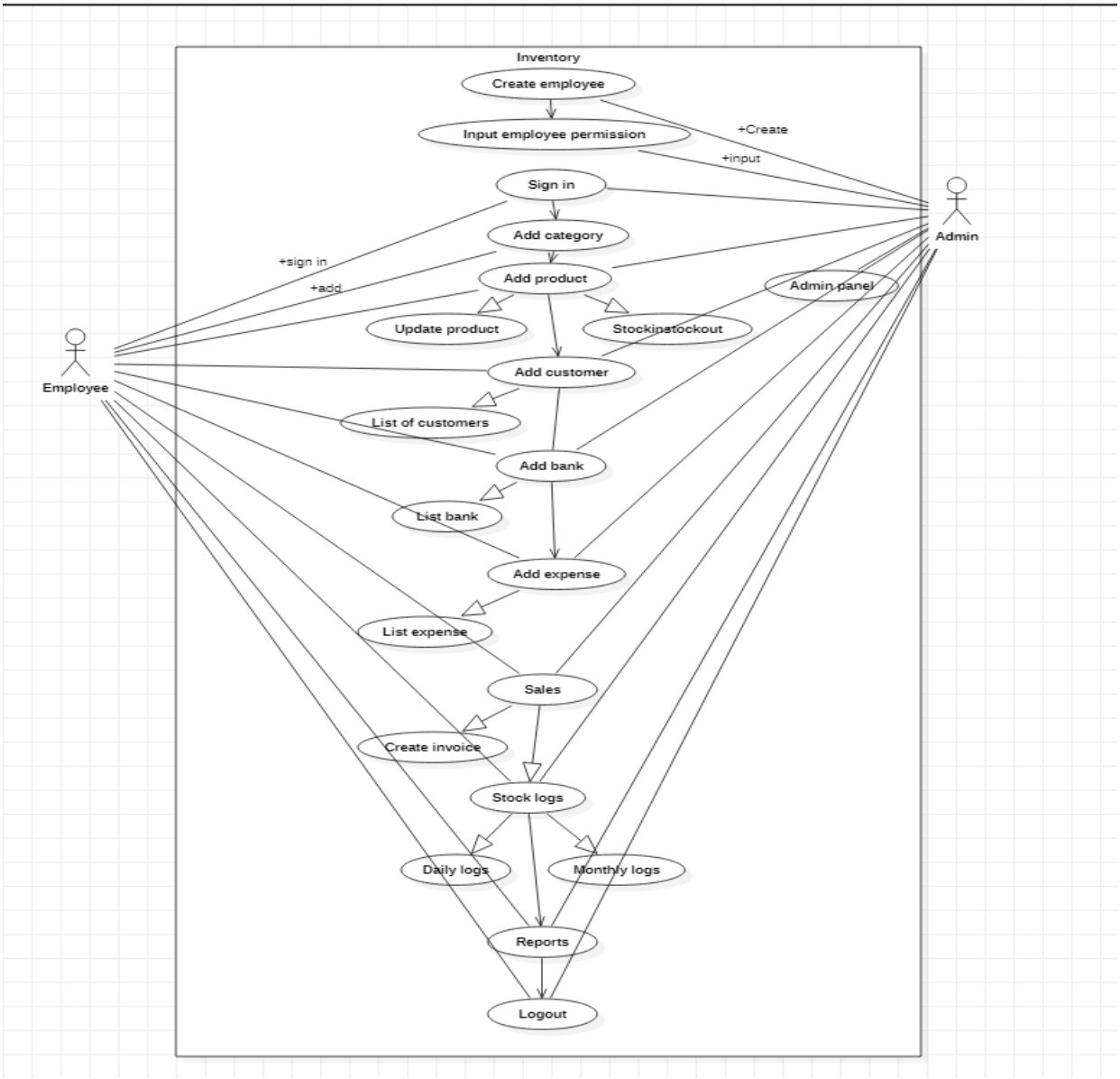


Figure 21: Use case

The above use-case diagram gives a brief description of the exterior working of the system the two actors are employees and admin/manager. The use case starts when the admin creates an employee and finalizes the permissions of the employee. They both can add categories, products and perform most of the same actions such as adding bank, viewing

the bank list, adding expenses, and viewing the expense list, list of stocks and logs, report and logout. Only the admin can assign privileges and access the admin page

4.2.2 System flowchart

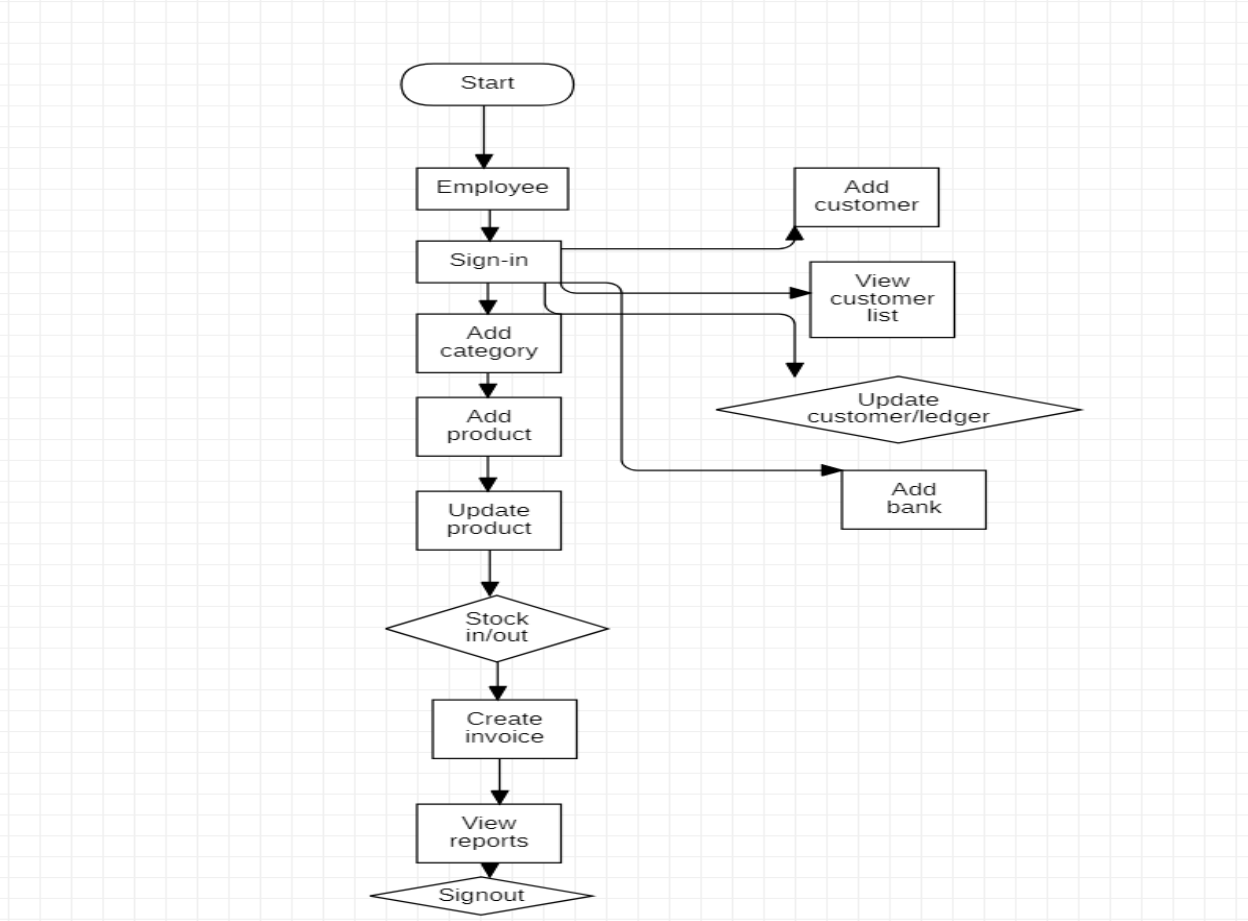


Figure 22: Flowchart

The system flowchart gives an insight into the flow and process of the system. This flowchart primarily focuses on the functions of the employee. The process starts when the admin signs. They can add category, then add product and perform certain functions in the product page. Stock in and stock out the products create invoice and view reports and finally logout.

4.2.3 Sequence diagram

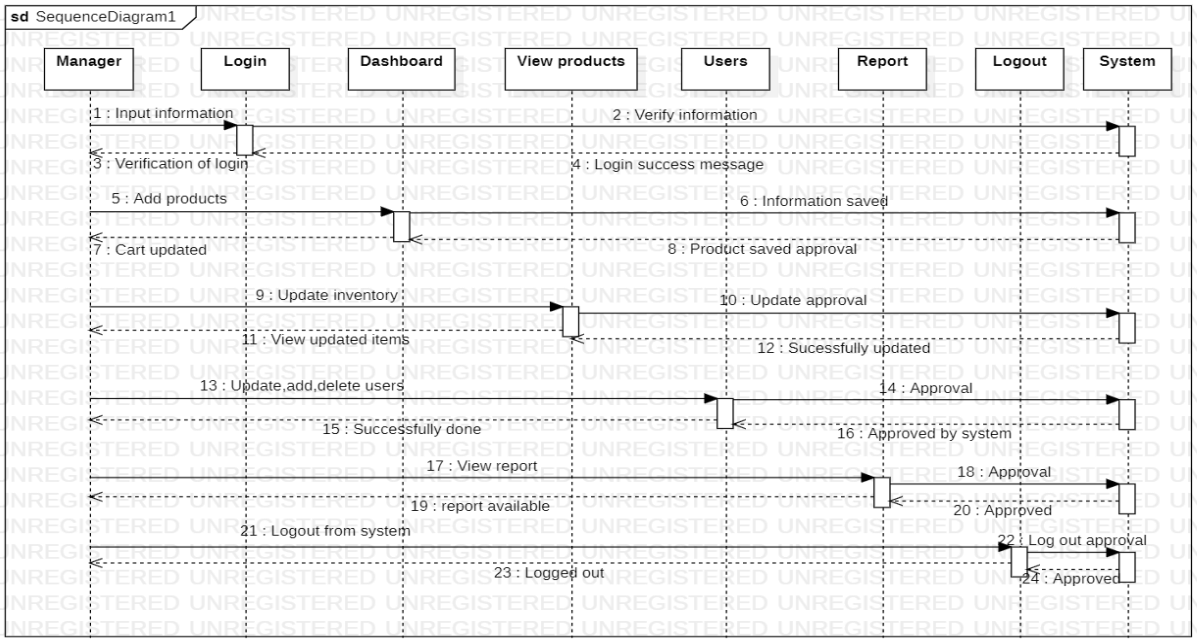


Figure 23: Sequence diagram

The above sequence diagram explains in the details about the actions that are carried out by a particular use-case. The use case in this instance is the admin/manager usecase.

4.2.4 Use case specification

Use case specification 1: Login

Name: Registration

Description: This use case is for the customer to login into the system to access various functions. The process ends when the customer successfully logged out.

Author(s): Immanuel

Actor(s): Customer

Location(s): Muscat

Status: Pathway defined

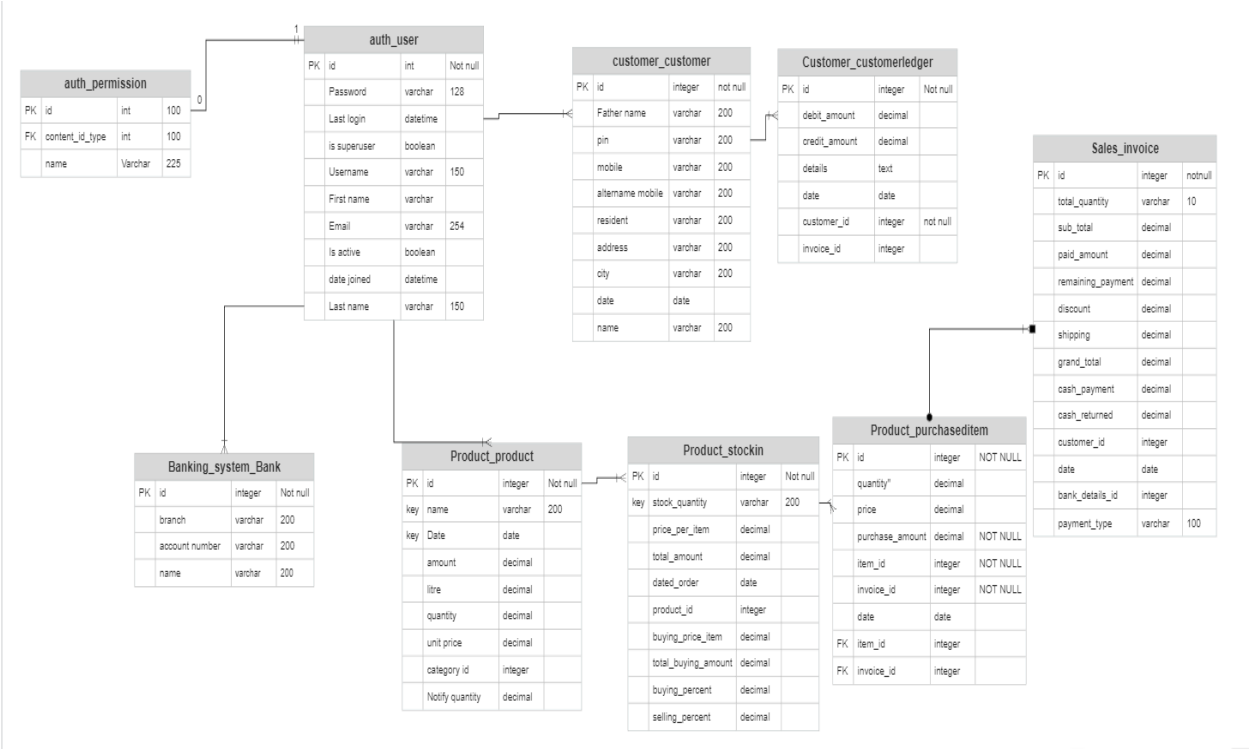
Priority: 1

Assumption: Employees need to login in order to access the system. If they don't, they won't be able to access the system. This is only possible if the customer has registered.

Precondition(s): The admin should input the employees to the system

Postcondition(s): The employee will be able to access the system

4.2.5 ER Diagram



The ER diagram shows the relationship between the different entities that are present in the data base of the system. The most important entities were selected from the database and their relationships have been mapped. The auth_permission entity gives permission to the user and the user can interact with the customer table and have access to the ledgers, bank functions. Product function, stock-in and stock-out and get the invoice based on the purchases.

4.3 Data dictionary

1. Author permissions table (auth_permissions)

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	content_type_id	integer	1	Null	0
2	codename	varchar (100)	1	Null	0
3	name	varchar (255)	1	Null	0

Table 16: author permissions

The above table structure is for the permissions that the author can give to the users

2. User information table

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	null	1
1	password	varchar (128)	1	null	0
2	last_login	datetime	0	null	0
3	is_superuser	bool	1	null	0
4	username	varchar (150)	1	null	0
5	first_name	varchar (30)	1	null	0
6	email	varchar (254)	1	null	0
7	is_staff	bool	1	null	0
8	is_active	bool	1	null	0
9	date_joined	datetime	1	null	0
10	last_name	varchar (150)	1	null	0

Table 17: user permissions

The above structure is of the information of the users that can be saved in backend of the system (Django)

3. **Adding bank table (banking_system_bank)**

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	null	1
1	branch	varchar (200)	0	null	0
2	account_number	varchar (200)	1	null	0
3	name	varchar (200)	1	Null	0

Table 18: Add bank table

This table structure shows the information that is stored while creating a new bank to save details.

4. **Bank details (banking_system_bankdetails)**

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	description	text	0	Null	0
2	date	date	0	Null	0
3	credit	decimal	0	Null	0
4	debit	decimal	0	Null	0
5	bank_id	integer	0	Null	0
6	invoice_id	integer	0	Null	0

Table 19: Add bank details table

The above table structure shows the information that is stored while creating new details for the bank after creating a new bank.

5. **Customer information (Customer_customer)**

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	father_name	varchar (200)	0	Null	0
2	Pin	varchar (200)	1	Null	0
3	mobile	varchar (200)	0	Null	0
4	alternate_mobile	varchar (200)	0	Null	0
5	resident	varchar (200)	0	Null	0
6	address	varchar (200)	0	Null	0
7	city	varchar (200)	0	Null	0
8	date	date	0	Null	0
9	name	varchar (200)	1	Null	0

Table 20: Customer information

The above structure shows the information that will be saved when creating a new customer

6. **Customer ledger table (Customer_customerledger)**

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	debit_amount	decimal	0	Null	0
2	credit_amount	decimal	0	Null	0
3	details	text	0	Null	0
4	date	date	0	Null	0
5	customer_id	integer	1	Null	0
6	invoice_id	integer	0	Null	0

Table 21: Customer ledger information

This structure shows the information in the page to add customer ledger

7. Admin log (Django_admin_log)

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	action_time	datetime	1	Null	0
2	object_id	text	0	Null	0
3	object_repr	varchar (200)	1	Null	0
4	change_message	text	1	Null	0
5	content_type_id	integer	0	Null	0
6	user_id	integer	1	Null	0
7	action_flag	smallint unsigned	1	Null	0

Table 22: Admin Log

The above table shows the structure of the admin logs. The table shows the details of the action the admin has performed with the time stamps

8. **Expense table (expense_expense)**

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	amount	decimal	0	Null	0
2	description	text	0	Null	0
3	date	date	0	Null	0

Table 23: expense table

The above table illustrates the structure of the table expenses which is used to store the expenses of the company

9. **Product table (product_product)**

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	name	varchar (200)	0	Null	0
2	date	date	0	Null	0
3	amount	decimal	0	Null	0
4	litre	decimal	0	Null	0
5	quantity	decimal	0	Null	0
6	unit_price	decimal	0	Null	0
7	category_id	integer	0	Null	0
8	notify_qty	decimal	0	Null	0

Table 24: Product table

The above table shows the list of the structure of the table in which products can be added.

10. Product category table (product_productcategory)

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	category	varchar (200)	0	Null	0
2	date	date	0	Null	0

Table 25: product category

This table shows the characteristics of the table in which the products and organized in the web application

11. Purchased items (product_purchaseditems)

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	quantity	decimal	0	Null	0
2	price	decimal	0	Null	0
3	purchase_amount	decimal	0	Null	0
4	item_id	integer	1	Null	0
5	invoice_id	integer	1	Null	0
6	date	date	0	Null	0

Table 26: purchased items

The above table shows the data dictionary of the products that are purchased in the web application. These values get saved after the item is purchased.

12. Product stock-in (product_stockin)

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	stock_quantity	varchar (200)	0	Null	0
2	price_per_item	decimal	0	Null	0
3	total_amount	decimal	0	Null	0
4	dated_order	date	0	Null	0
5	product_id	integer	0	Null	0
6	buying_price_item	decimal	0	Null	0
7	total_buying_amount	decimal	0	Null	0
8	buying_percent	decimal	0	Null	0
9	selling_percent	decimal	0	Null	0

Table 27: stock in

The above table shows the data dictionary of the stock in table which is used to add the stocks of a product after adding the product

13. Product stock-out (product_stockout)

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	stock_out_quantity	decimal	0	Null	0
2	date	date	0	Null	0
3	product_id	integer	0	Null	0
4	Buying_price	decimal	0	Null	0
5	Selling_price	decimal	0	Null	0
6	invoice_id	integer	0	Null	0

Table 28: stock out

The above table shows the structure of the stock out table which shows the list of stock that has left the company

14. Sales invoice (sales_invoice)

cid	name	type	notnull	dflt_value	pk
0	id	integer	1	Null	1
1	total_quantity	varchar (10)	0	Null	0
2	sub_total	decimal	0	Null	0
3	paid_amount	decimal	0	Null	0
4	remaining_payment	decimal	0	Null	0
5	discount	decimal	0	Null	0
6	shipping	decimal	0	Null	0
7	grand_total	decimal	0	Null	0
8	cash_payment	decimal	0	Null	0
9	cash_returned	decimal	0	Null	0
10	customer_id	integer	0	Null	0

11	date	date	0	Null	0
12	bank_details_id	integer	0	Null	0
13	payment_type	varchar (100)	1	Null	0

Table 29: sales invoice

The above table illustrates the information of the datatypes of the sales invoice which is stored in the database.

15. Installment invoice (sales_invoiceinstallment)

cid	**name**	**type**	**notnull**	**dflt_value**	**pk**
0	id	integer	1	Null	1
1	paid_amount	decimal	0	Null	0
2	description	text	0	Null	0
3	date	date	0	Null	0
4	invoice_id	integer	1	Null	0

Table 30: installment invoice

The above data dictionary illustrated the types of data stored when the customers choose instalment as payment method.

4.4 Purpose of diagrams

The purpose of various diagrams will be explained here

1. Use-Case diagram: Use-Case diagram is used for the modelling purpose of the behavior of the system. It only shows the actors and the outer activities of the system not internal.
2. Class diagram: Class diagram shows the classes, operation, and the attributes within each class of the system and shows their relationship.
3. Sequence diagram: Sequence diagram shows the interaction of the elements of the system and describe their actions.
4. Flowchart: it shows the sequence of the activities in the system
5. ER diagram: Explains how the entities of the system collaborate with each other

4.4 Database security

Security risks and attacks are becoming more prevalent as applications become more complicated and reliant on third-party libraries, among other factors. According to a Forrester analysis published in 2020, the majority of external assaults are carried out by exploiting a software vulnerability or a web application.

The databased used by this web application is sqllite3. The auditing of the database will be done using SQL server. SQL provides so many features for database. Microsoft assists in keeping the database management systems secure by releasing constant updates.

Django backend is also very protected against threats. Django has built in SQL injection protection which prevents from outside attacks. Django also has XSS protection which prevents attacks that come from the client side.

From the library point of view, only trusted libraries were imported.

4.5 Screenshots of each page

4.5.1 Login page

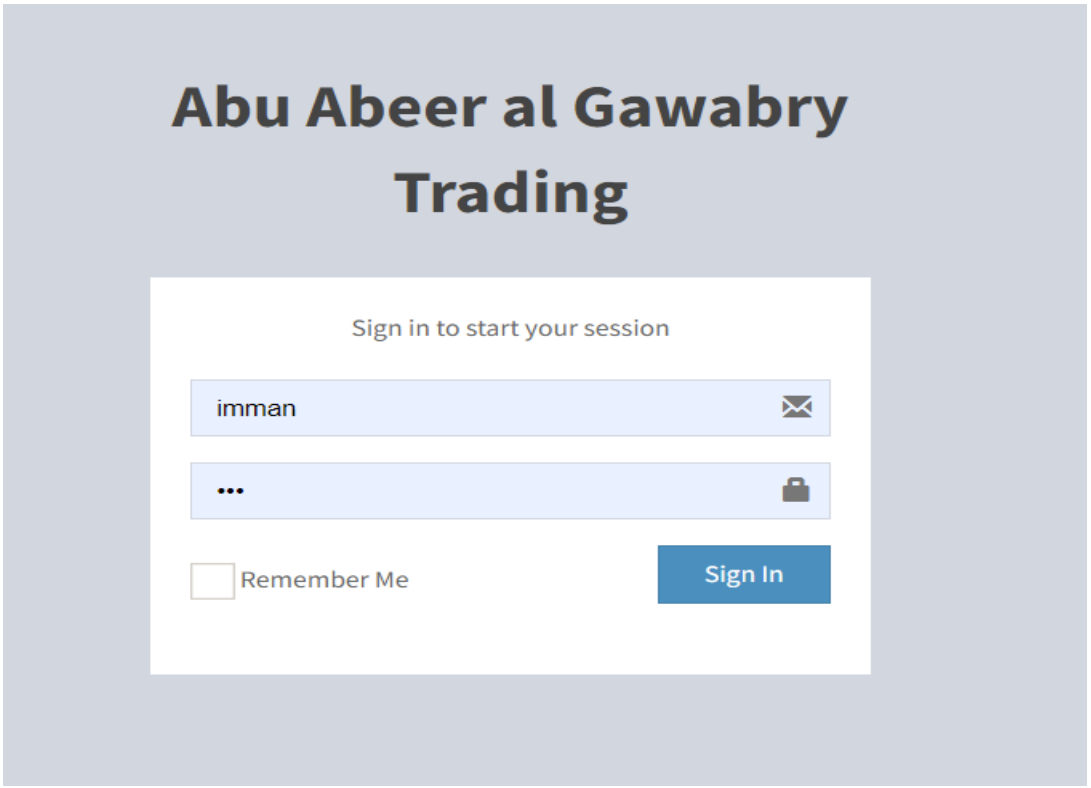


Figure 24: Login page

4.5.2 Add category page

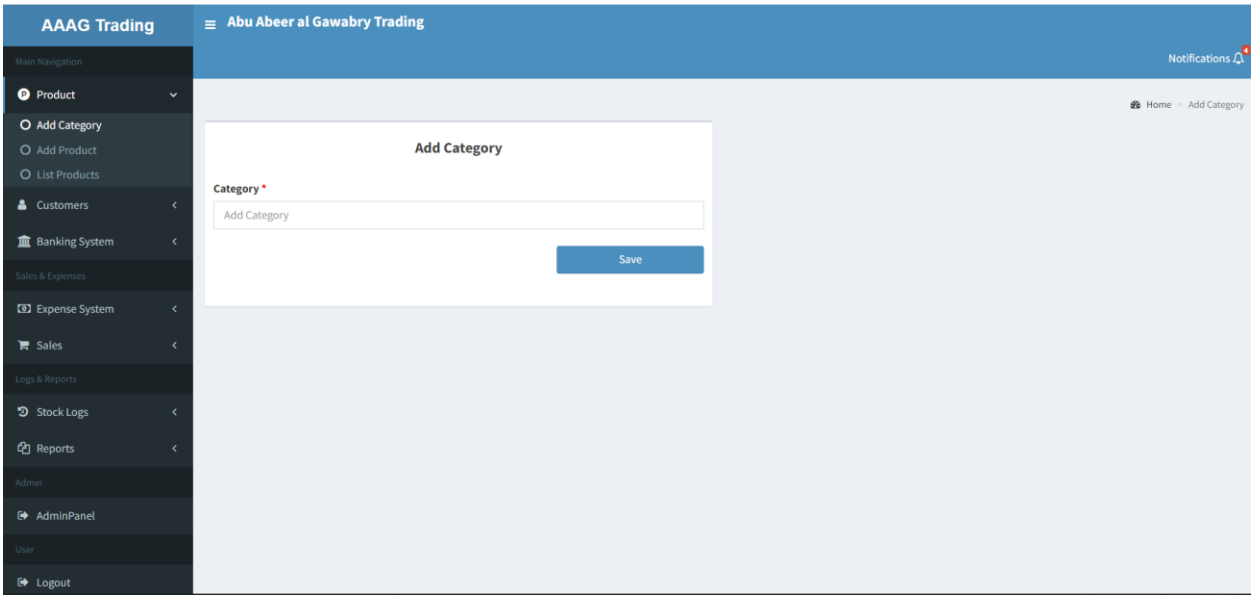


Figure 25: Add category page

4.5.3 Add product page

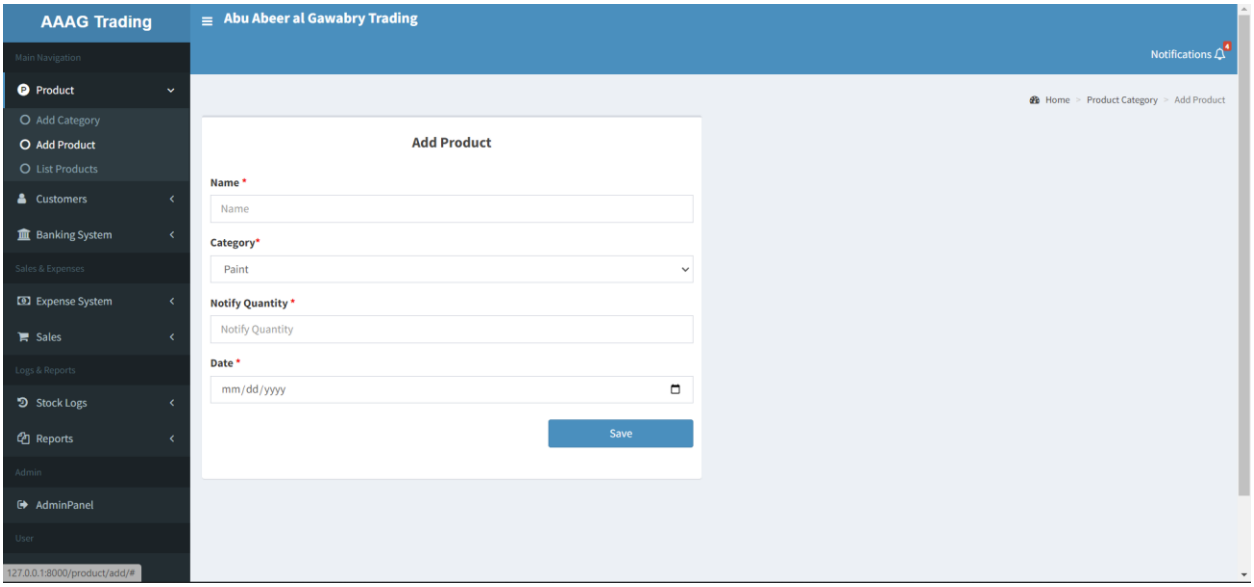


Figure 26: Add product page

4.5.4 List product page

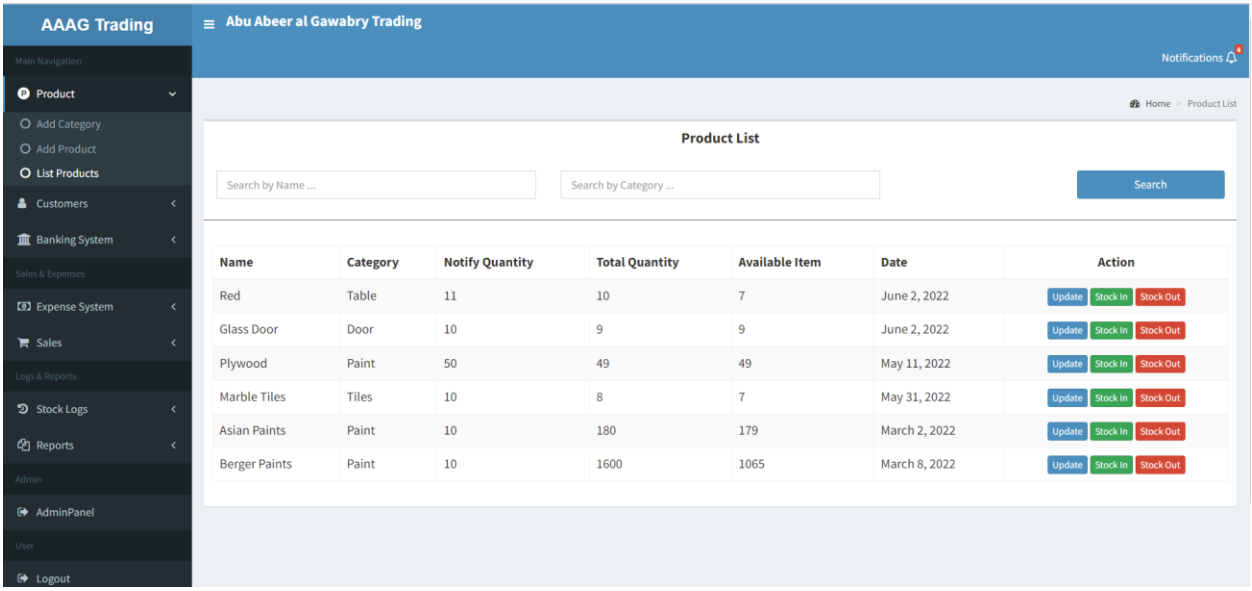


Figure 27: List product page

4.5.5 update product

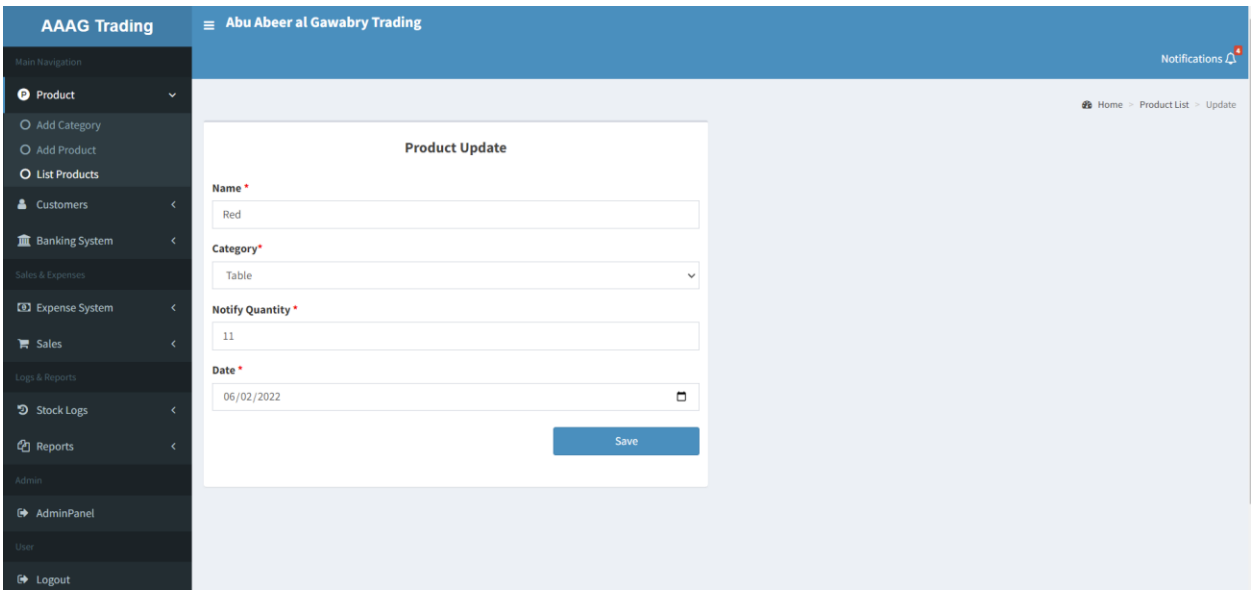


Figure 28:update product page

4.5.6 Stock in product

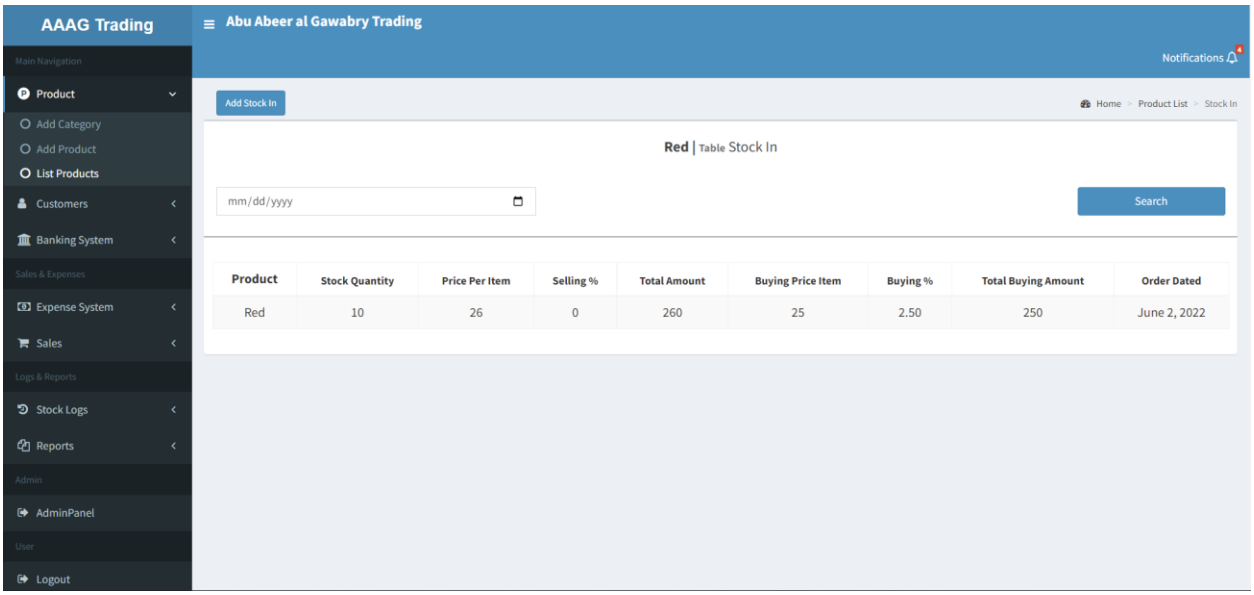


Figure 29: Product stock in page

4.5.7 Stock out product

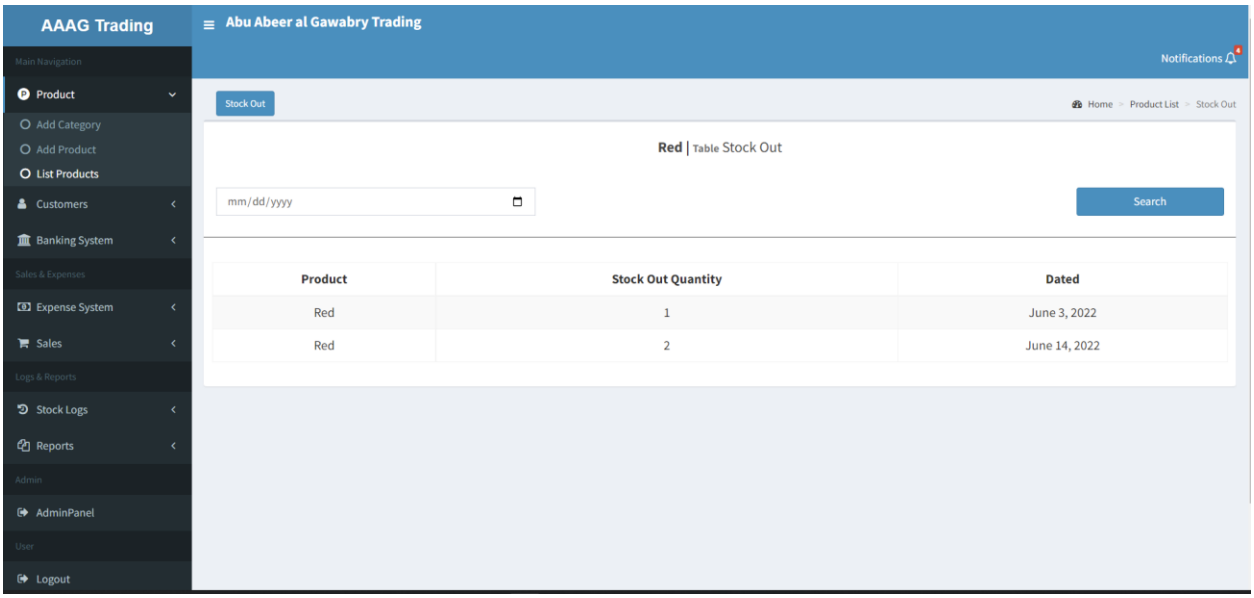


Figure 30: Product stock out page

4.5.8 Add customer

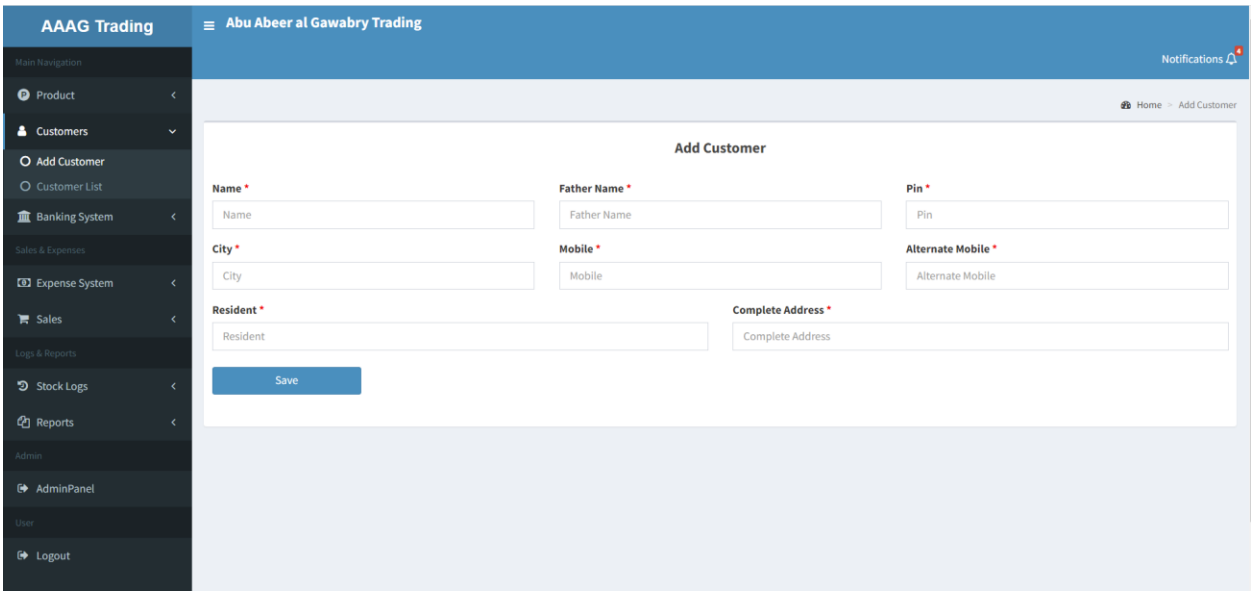


Figure 31: Add customer page

4.5.9 List customer

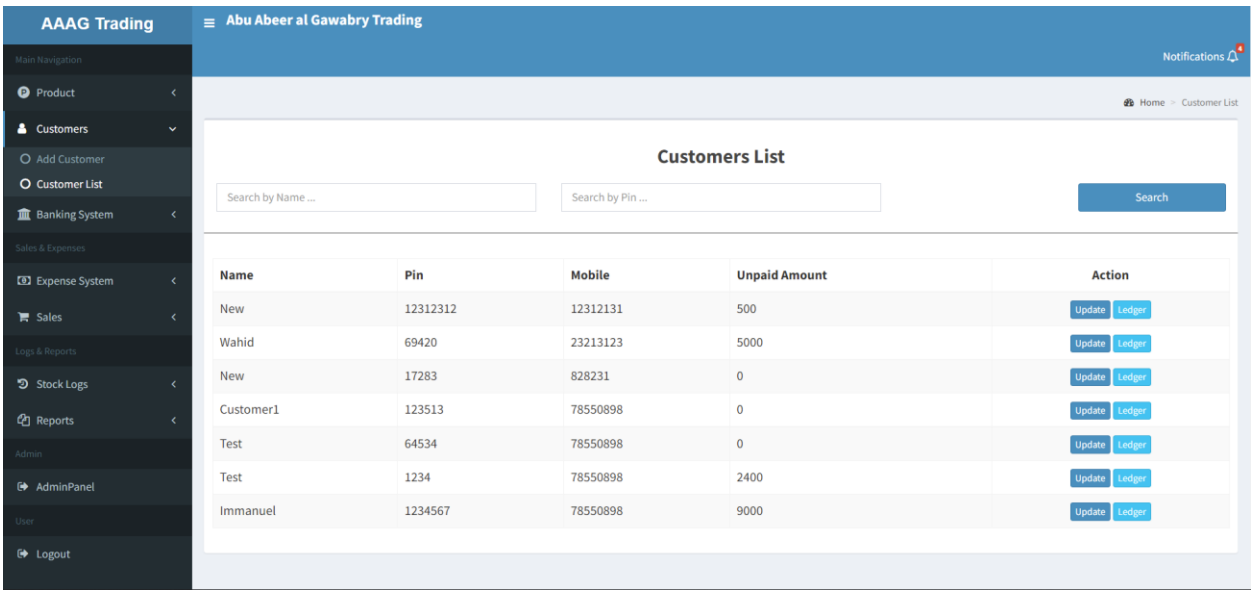


Figure 32: Customer list page

4.5.10 Customer ledger

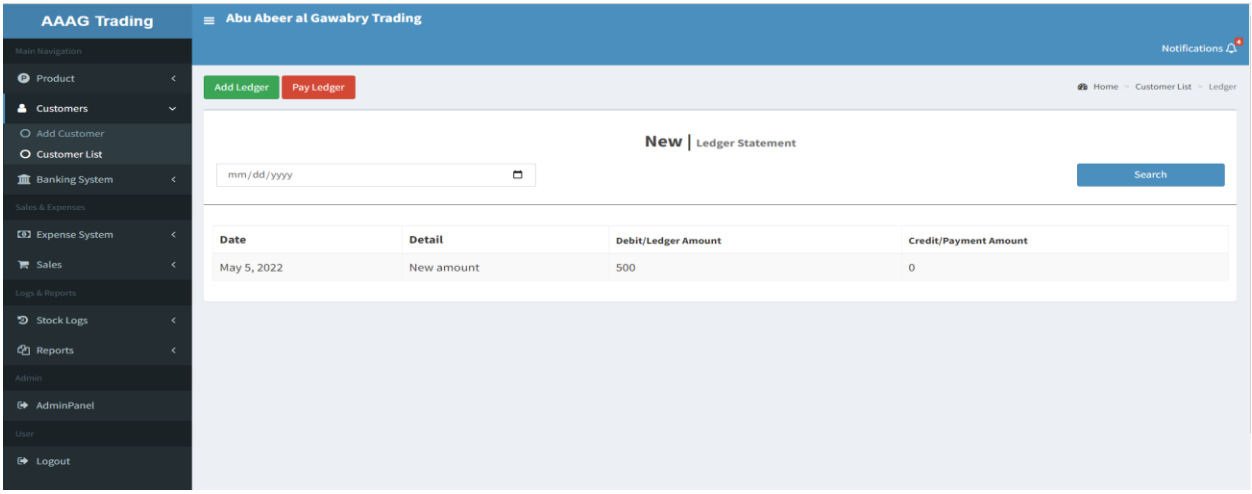


Figure 33:Customer ledger page

4.5.11 Add ledger

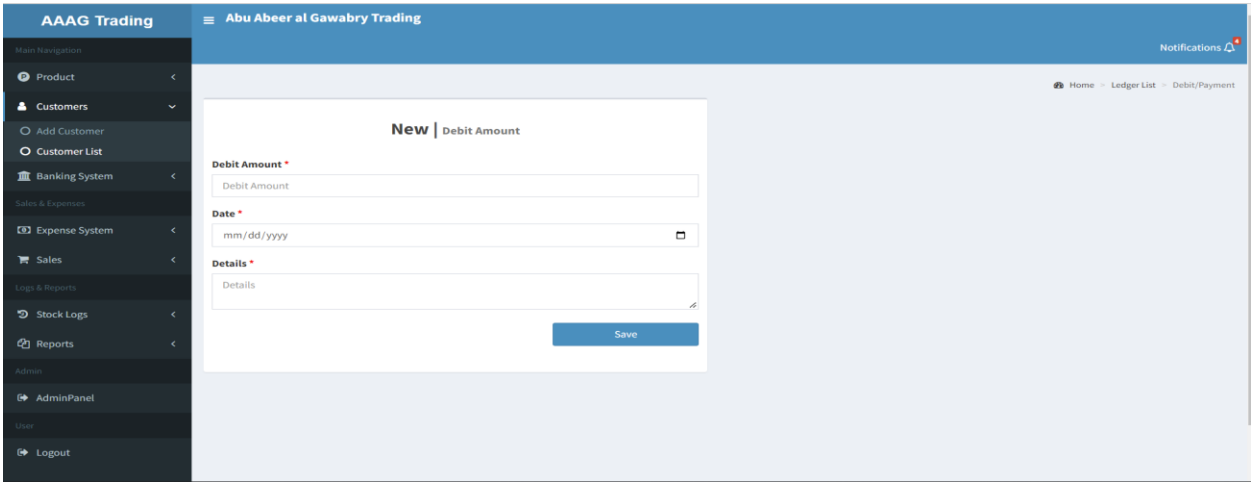


Figure 34: Add ledger page

4.5.12 Pay ledger

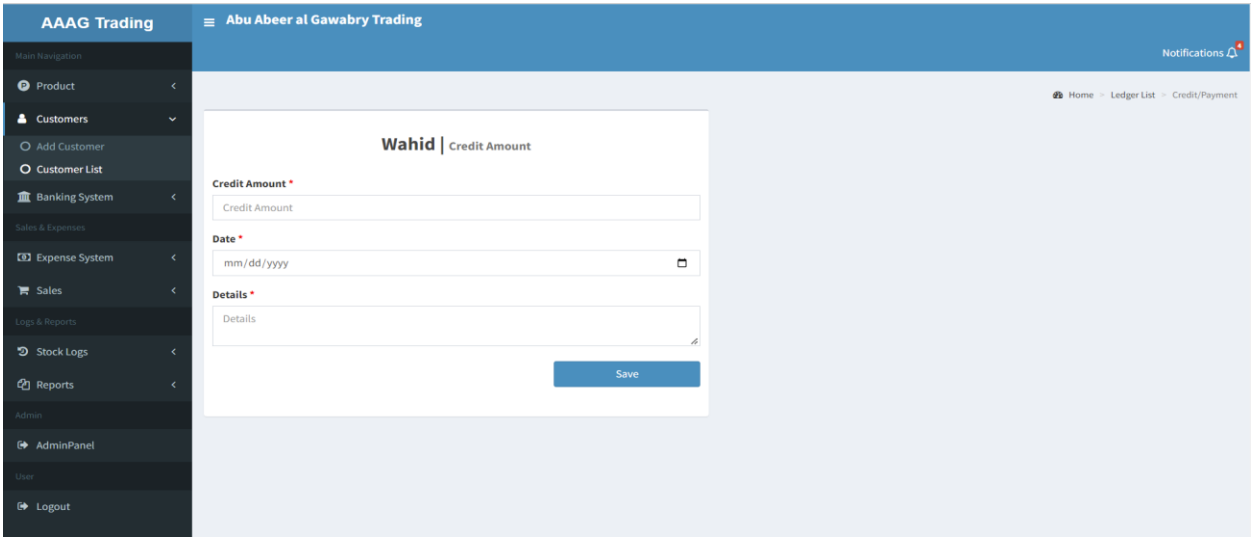


Figure 35: Pay ledger

4.5.13 Add bank

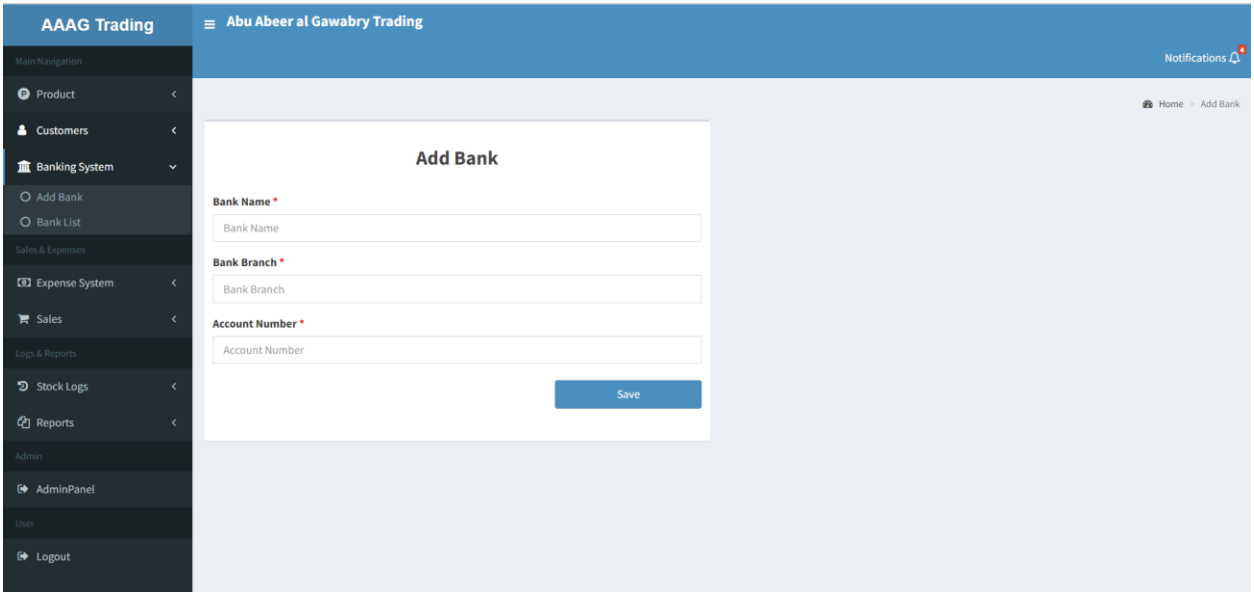


Figure 36: add bank

4.5.14 List bank

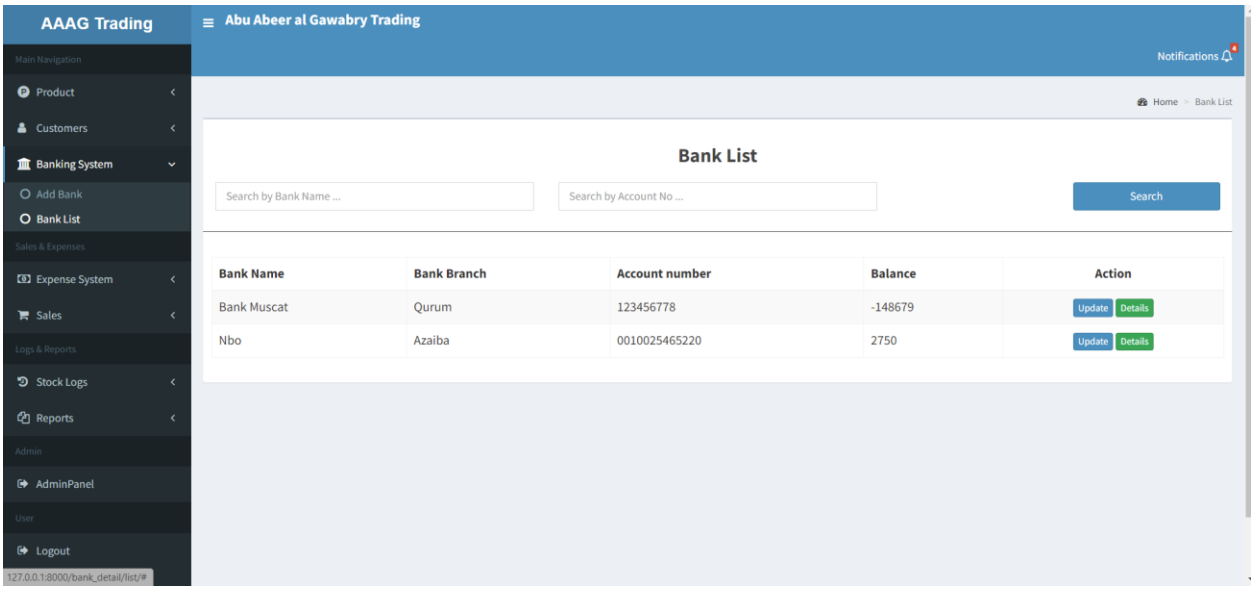


Figure 37: list bank

4.5.15 Update bank

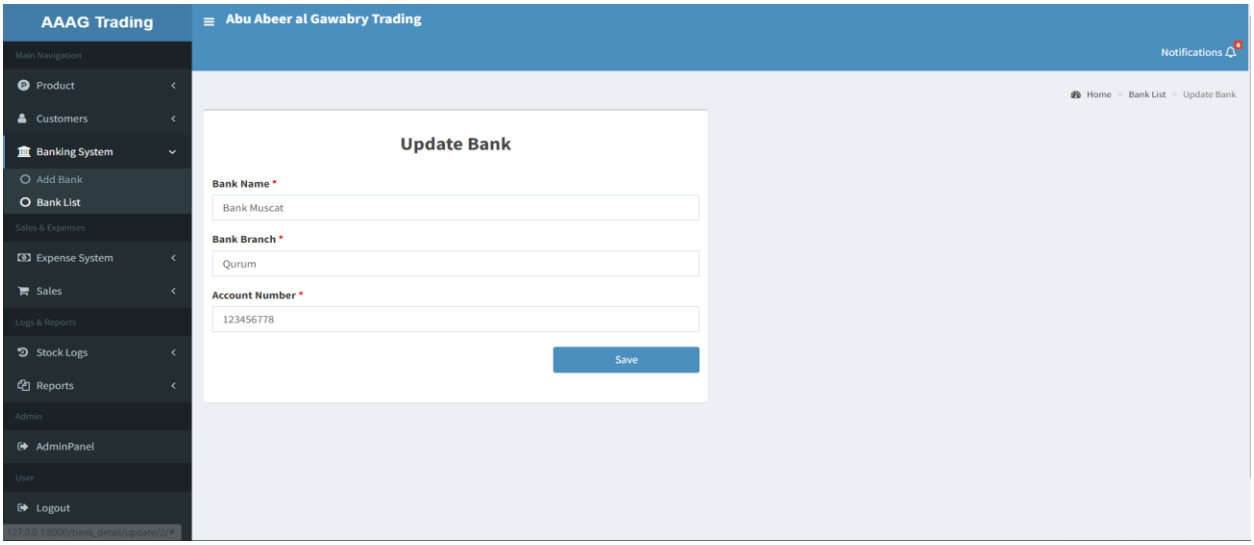


Figure 38: Update bank

4.5.16 Add debit

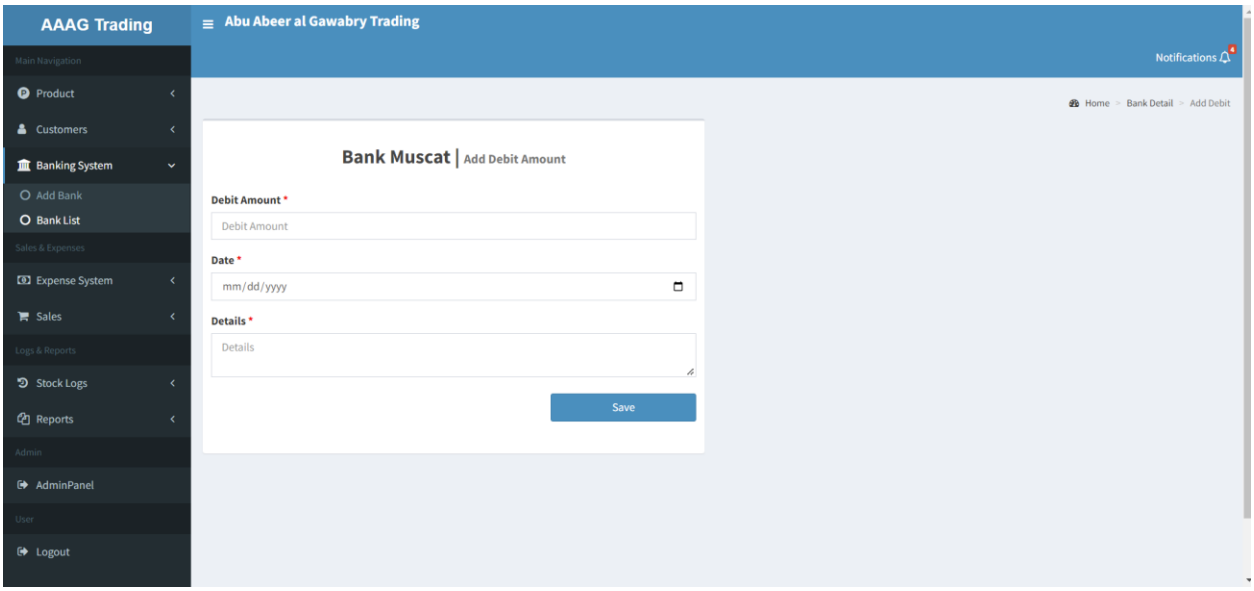


Figure 39: Add debit

4.5.17 Add credit

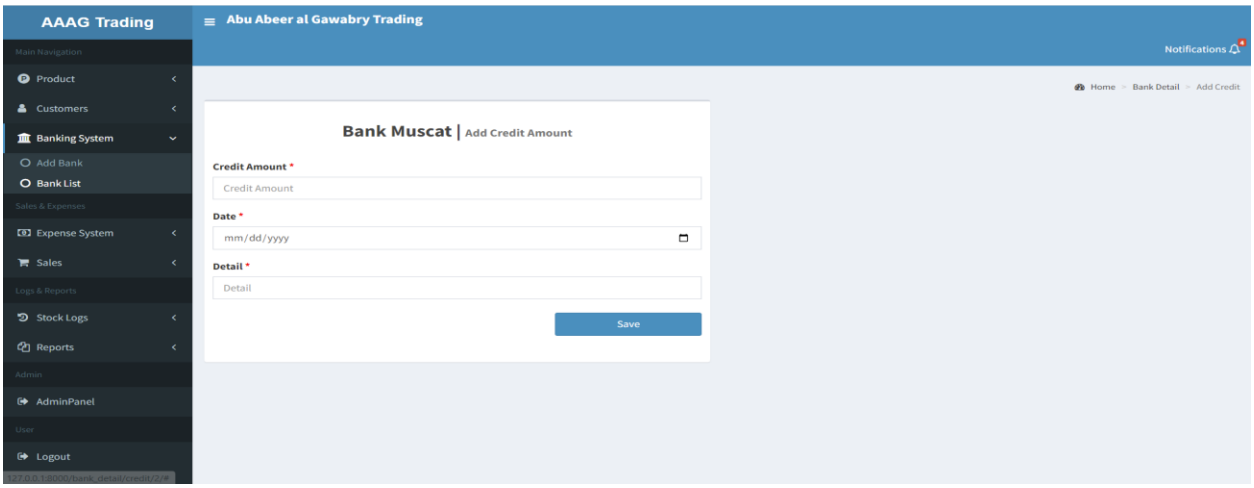


Figure 40: Add credit

4.5.18 Add expense

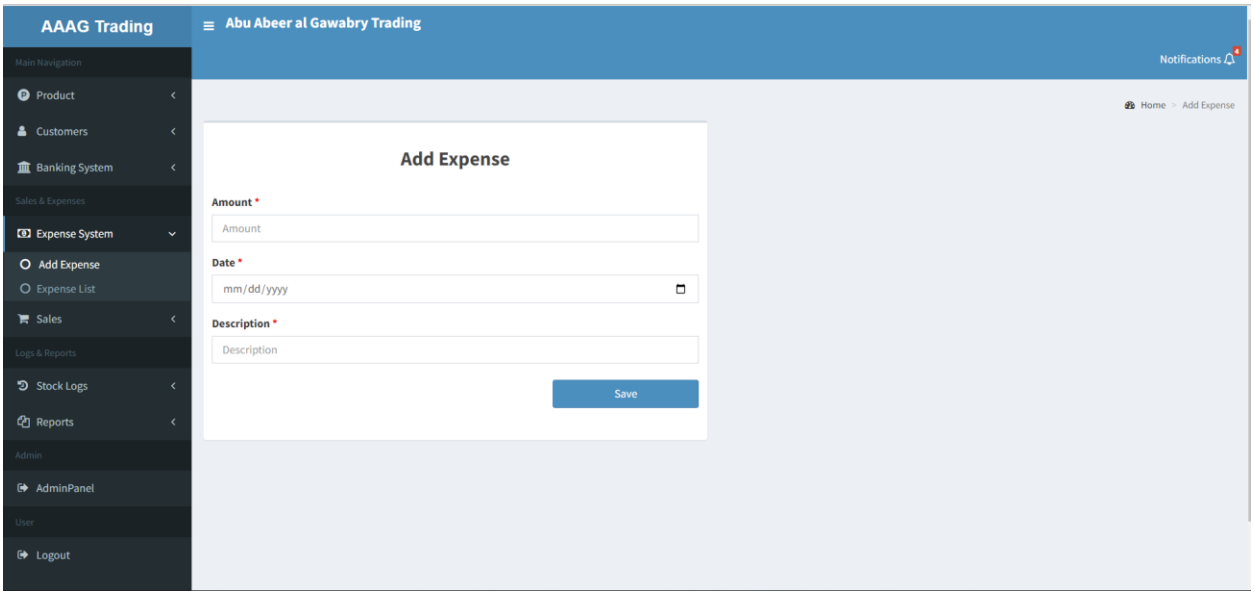


Figure 41: Add expense

4.5.19 Expense list

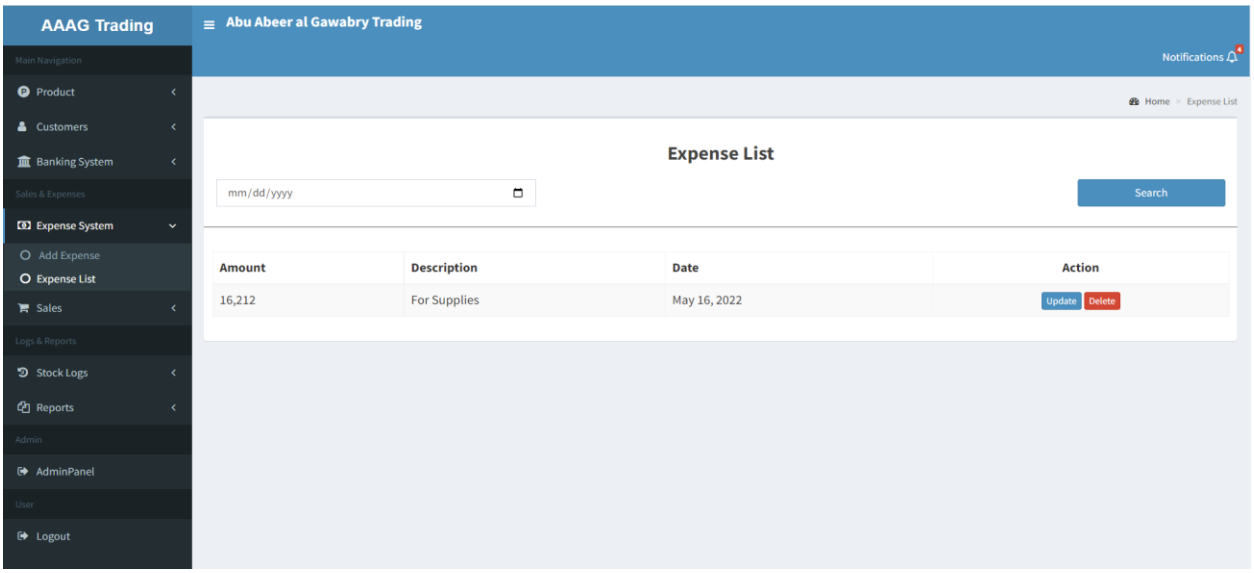


Figure 42: Expense list

4.5.20 Update expense

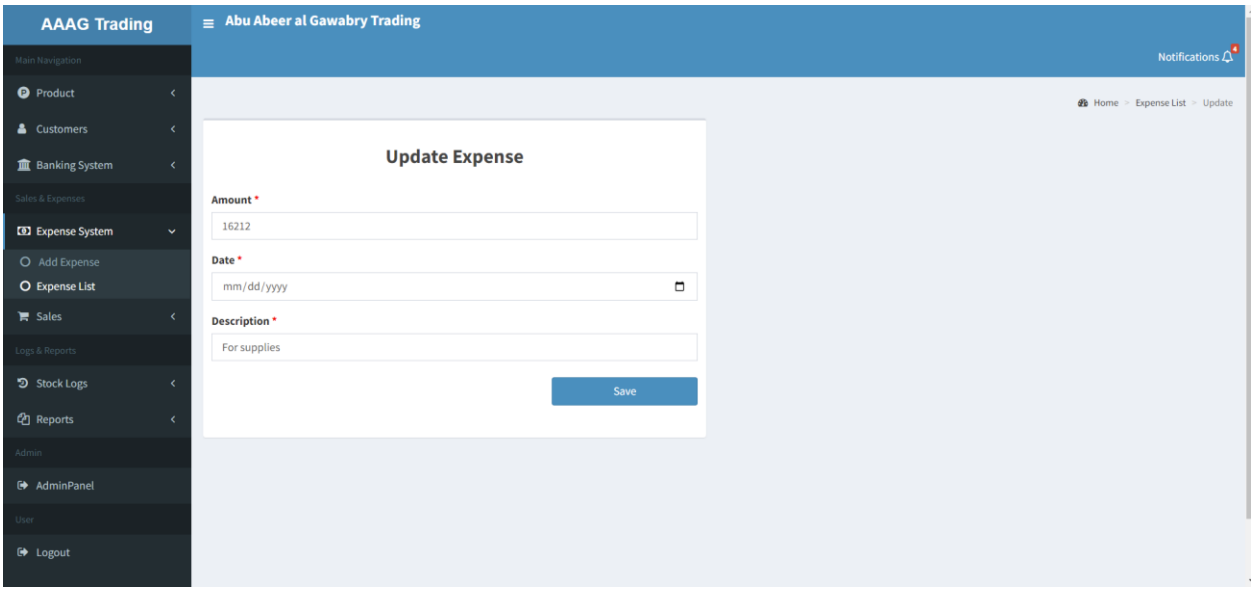


Figure 43: Update expense

4.5.21 Invoice Page

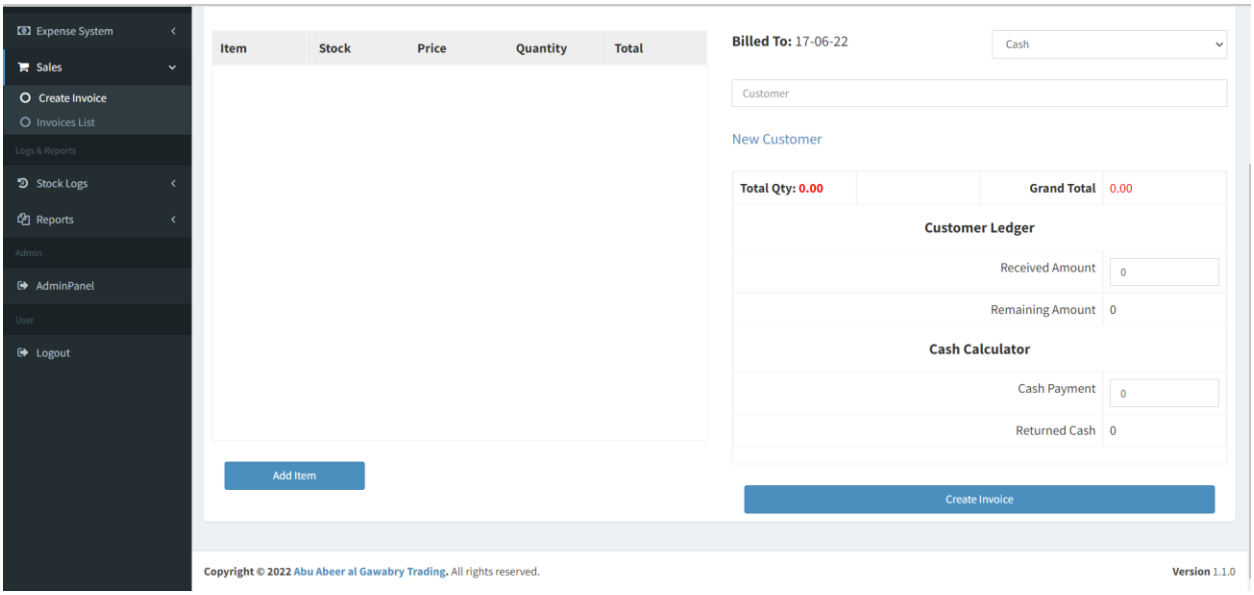


Figure 44: Invoice page

4.5.22 Invoice list

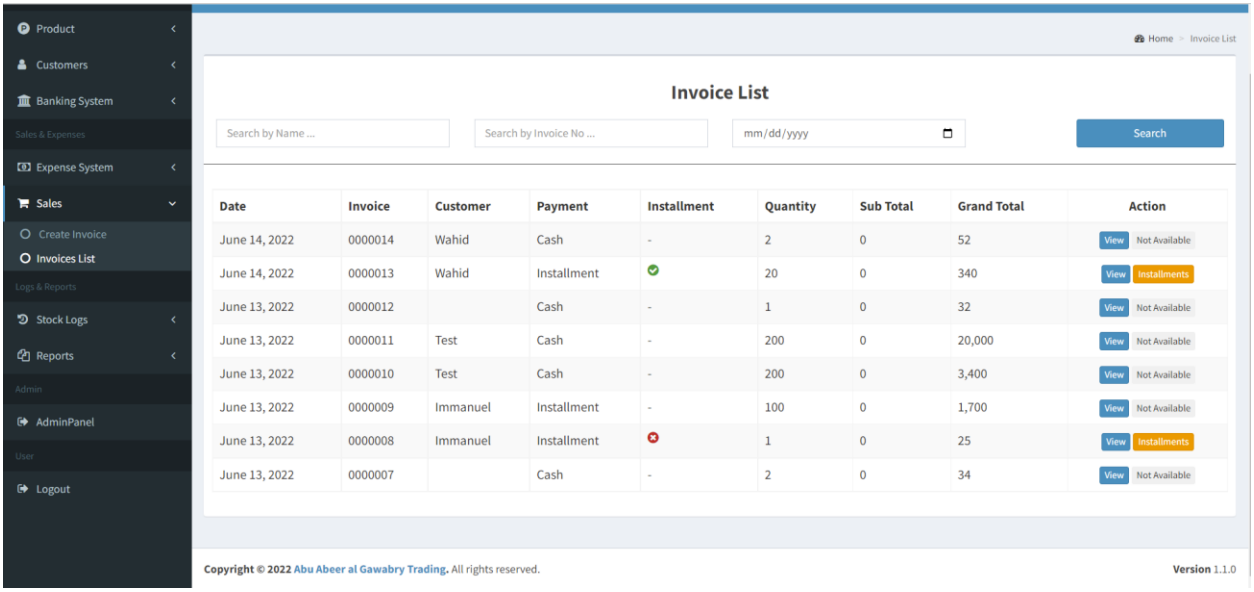


Figure 45: Invoice list

4.5.23 view invoice

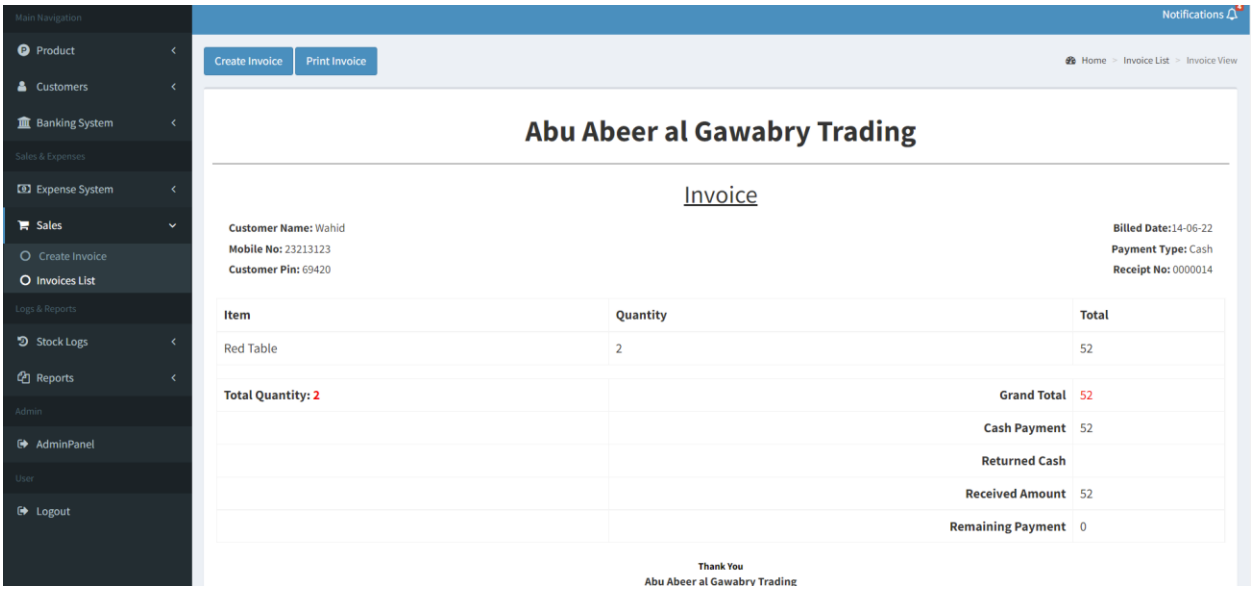


Figure 46: View invoice

4.5.24 Print voice

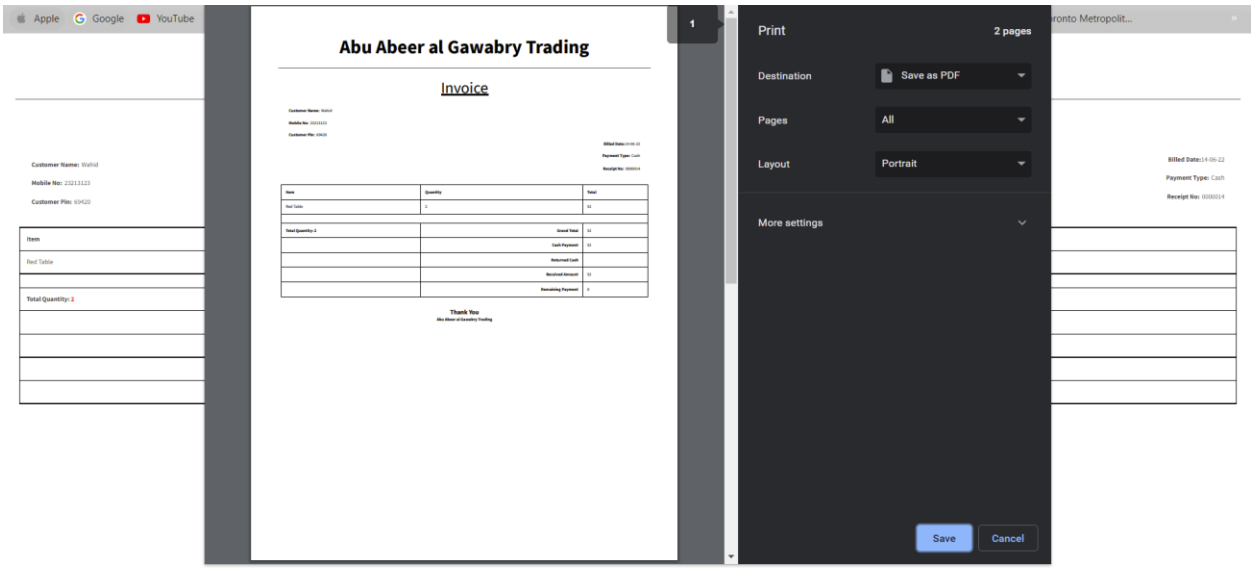


Figure 47: Print invoice

4.5.25 View installment

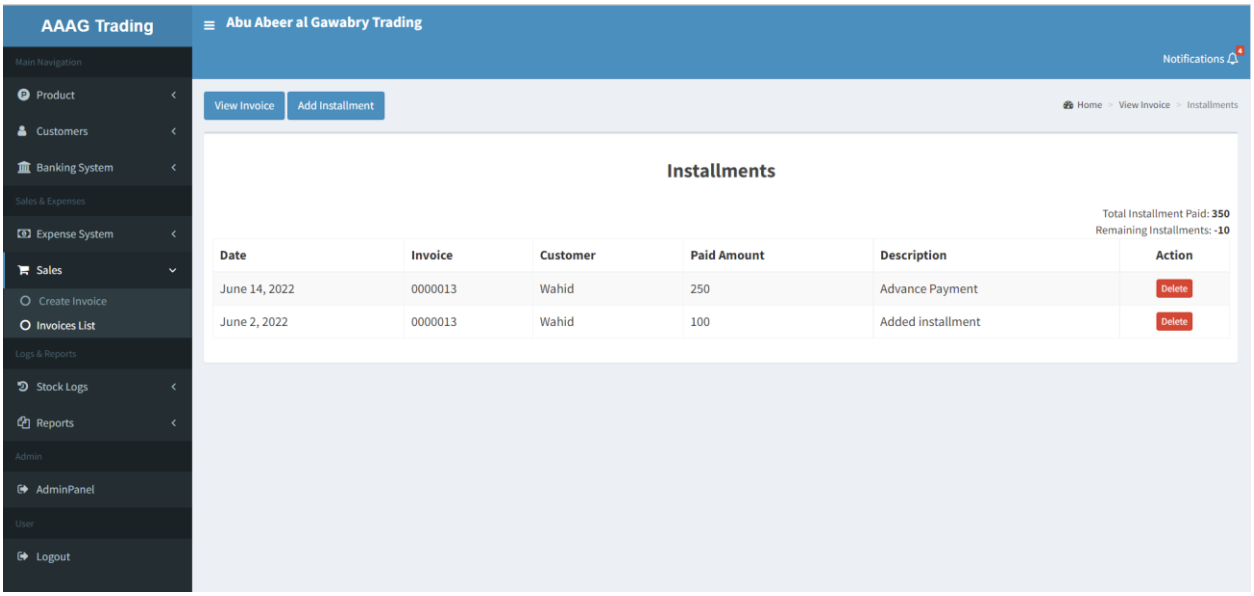


Figure 48: view installment

4.5.26 Add installment

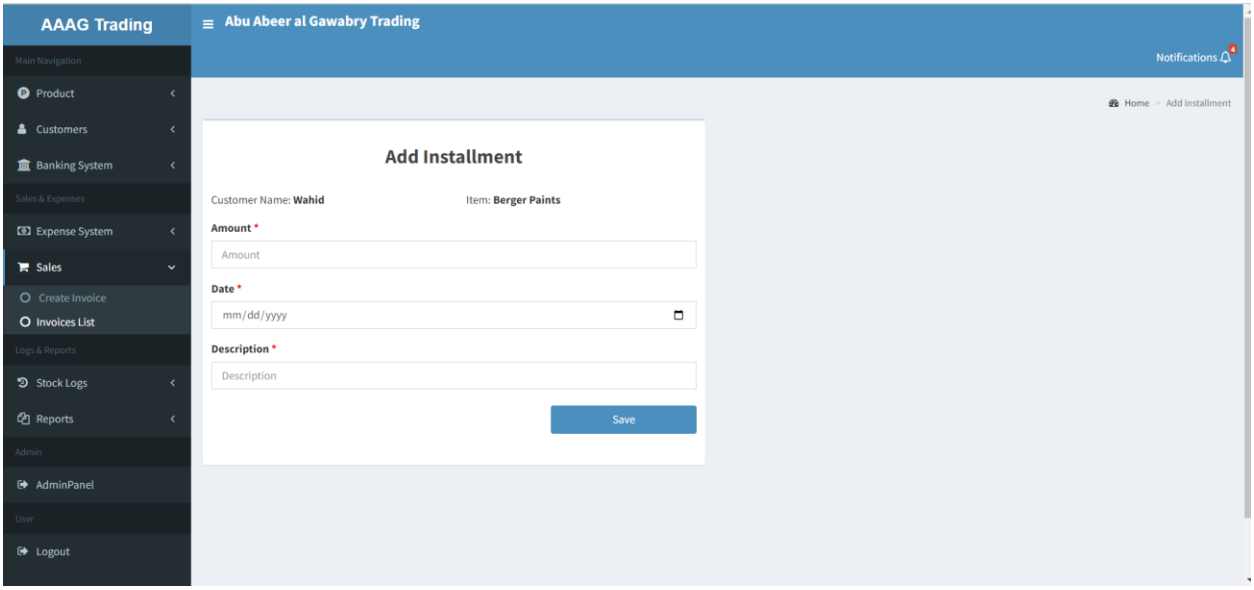


Figure 49: Add installment

4.5.27 Daily logs

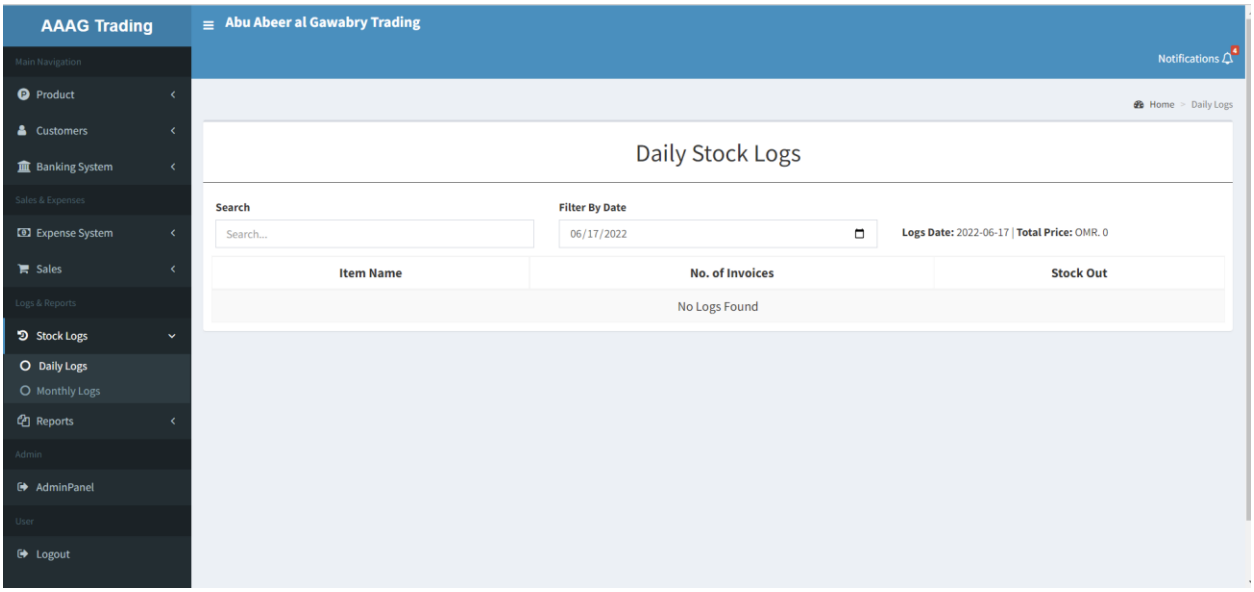


Figure 50: Daily logs

4.5.28 Monthly stocks

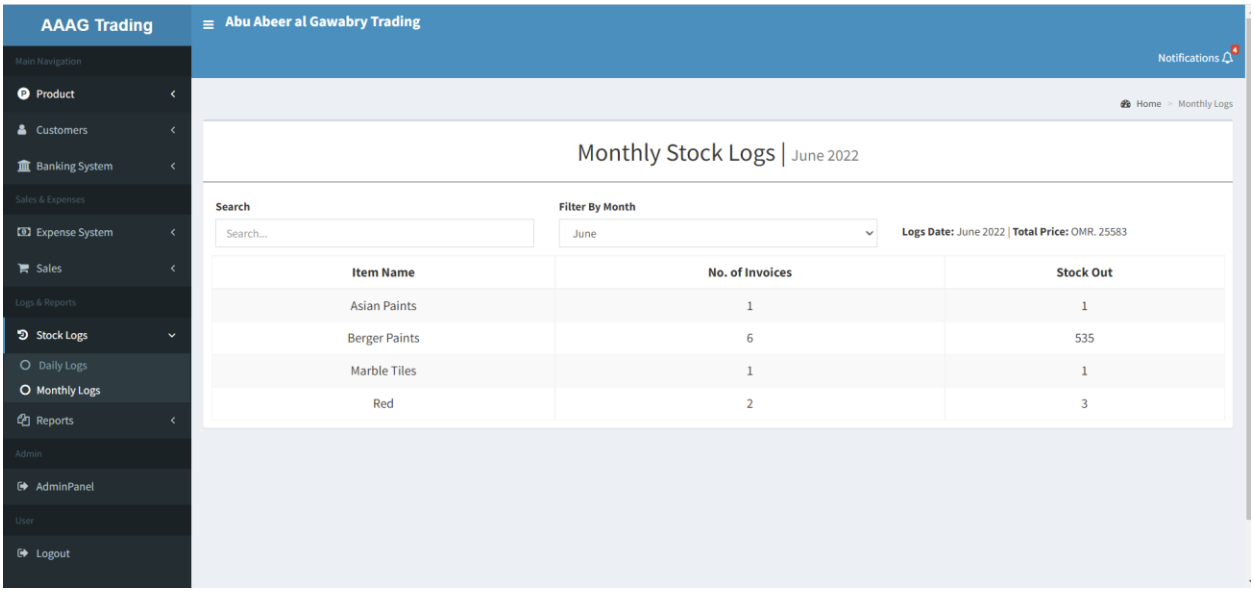


Figure 51: Monthly logs

4.5.29 Reports

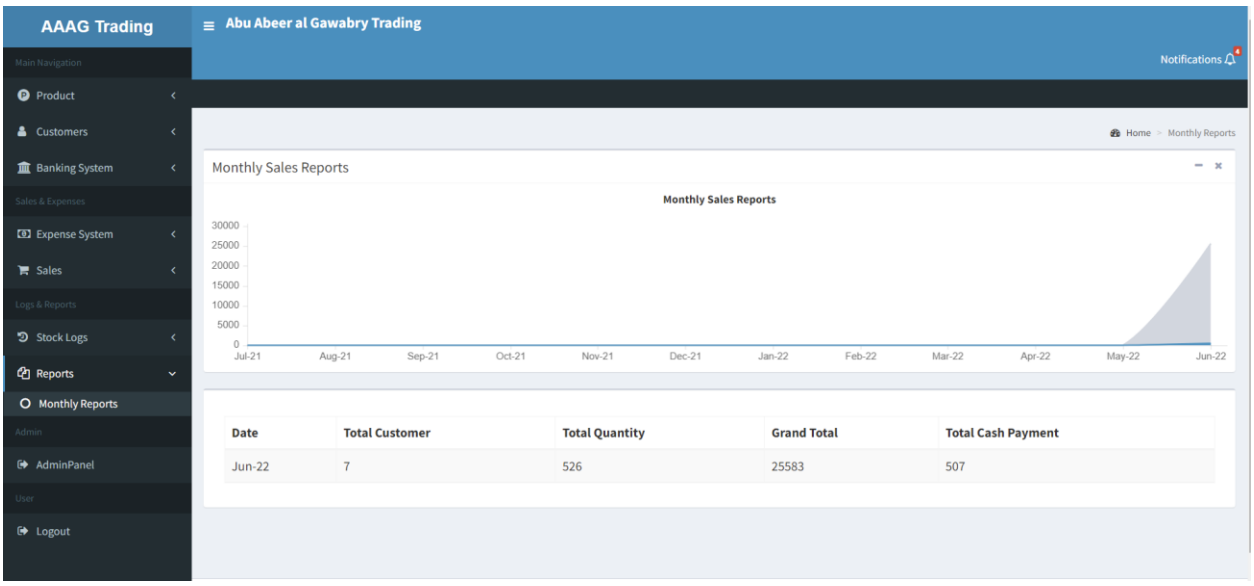


Figure 52: Month reports

4.5.30 Admin panel

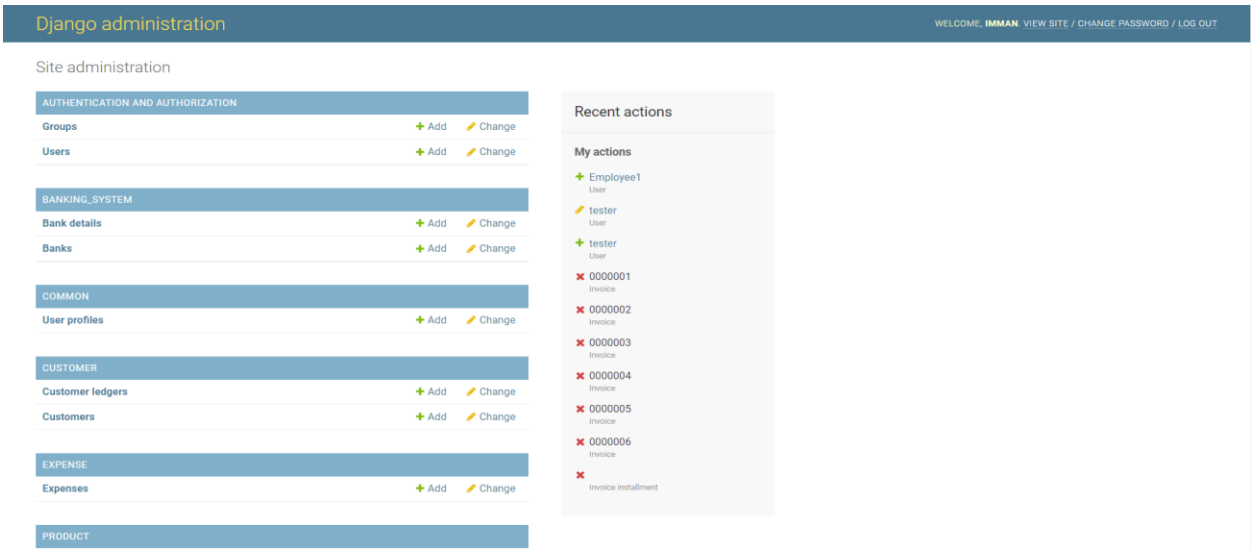


Figure 53: Admin panel

Chapter 5: Development and evaluation

5.2 Outline of the chapter

This chapter contains the explanation about each page of the system with some information regarding it. It will also contain screen shots along with the codes. The aspect of testing required for the system will be justified and the different test strategies will be explained.

Test cases will be attached in the form of tables to see the outcome of the tests performed and the desired output. Analysis of the test results will also be presented along with a critical evaluation of the system.

5.2 Description of the system

This section will contain a brief description of the system along with its screenshots and explanation of the screenshots

5.2.1 Login page

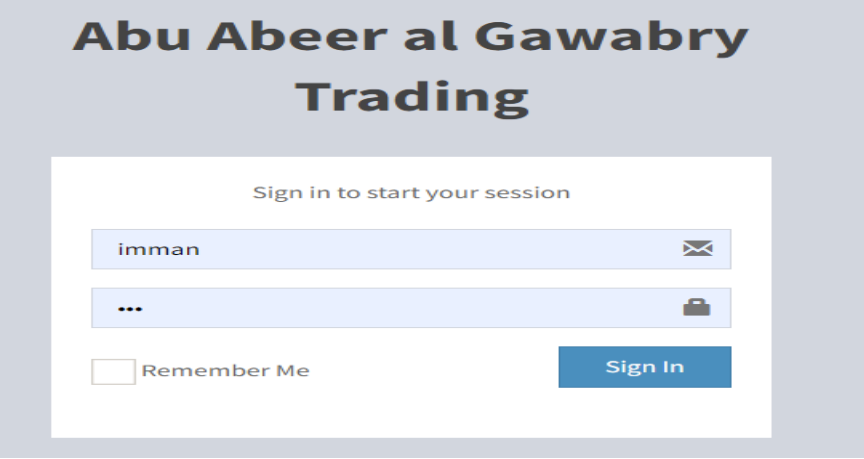


Figure 54: Login

The login page is one of the main pages of the system. The employee and admin/manager can login to the system through this page to access the main functionalities

5.2.2 Add category page

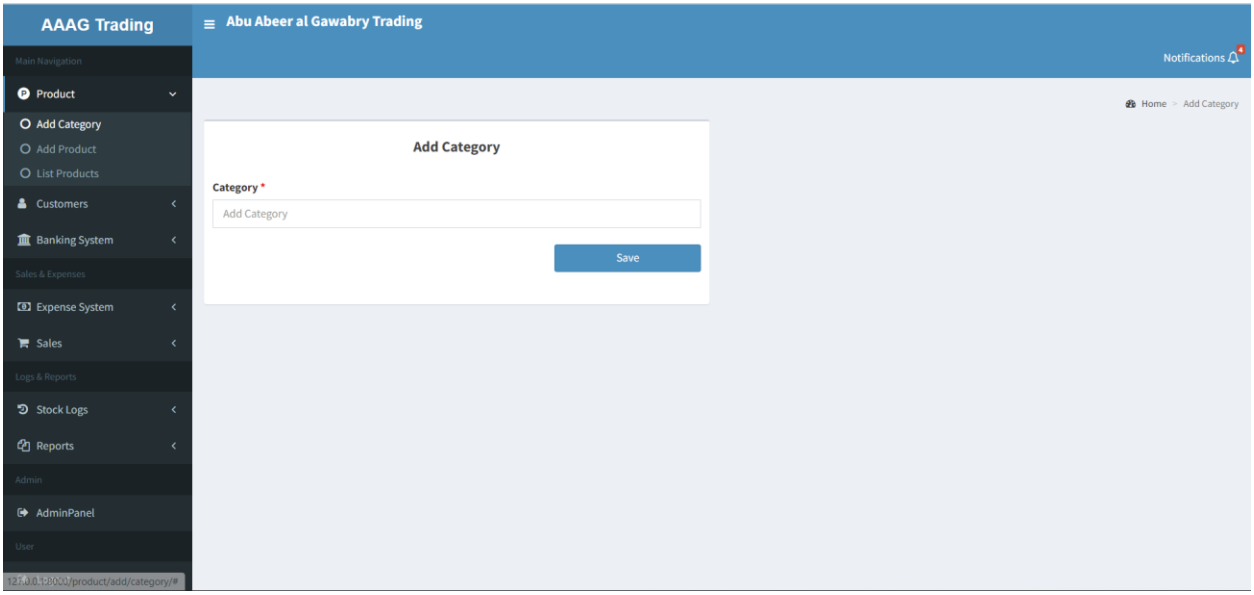
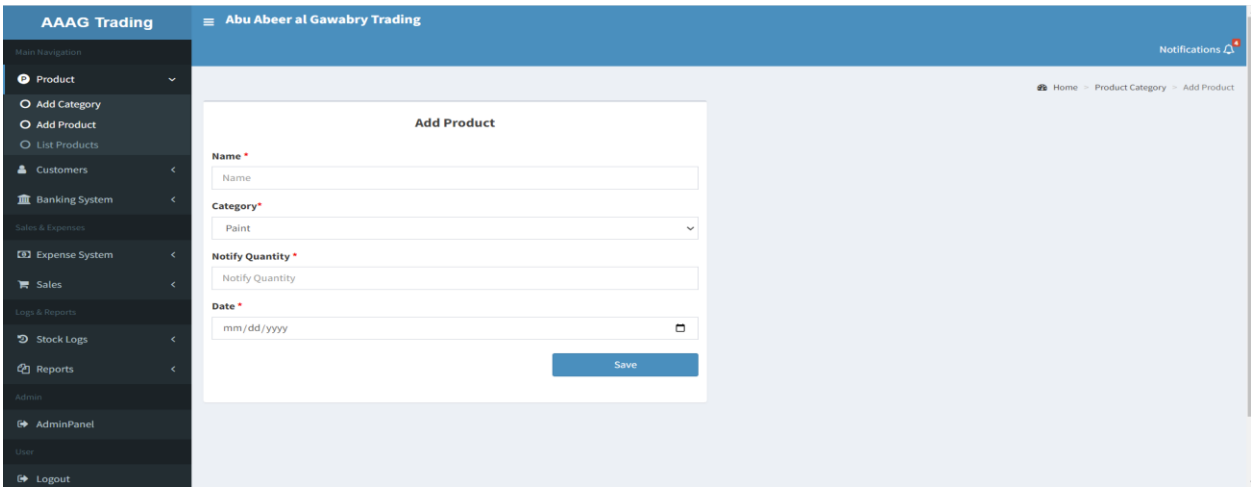


Figure 55: add category

In this page, the user can add the category in which the product belongs to. This makes it easier to organize the products in the inventory.

5.2.3 Add product



This is the page in which product is added. The product name, category, notification quantity and the date must be added to add a product.

5.2.4 List product

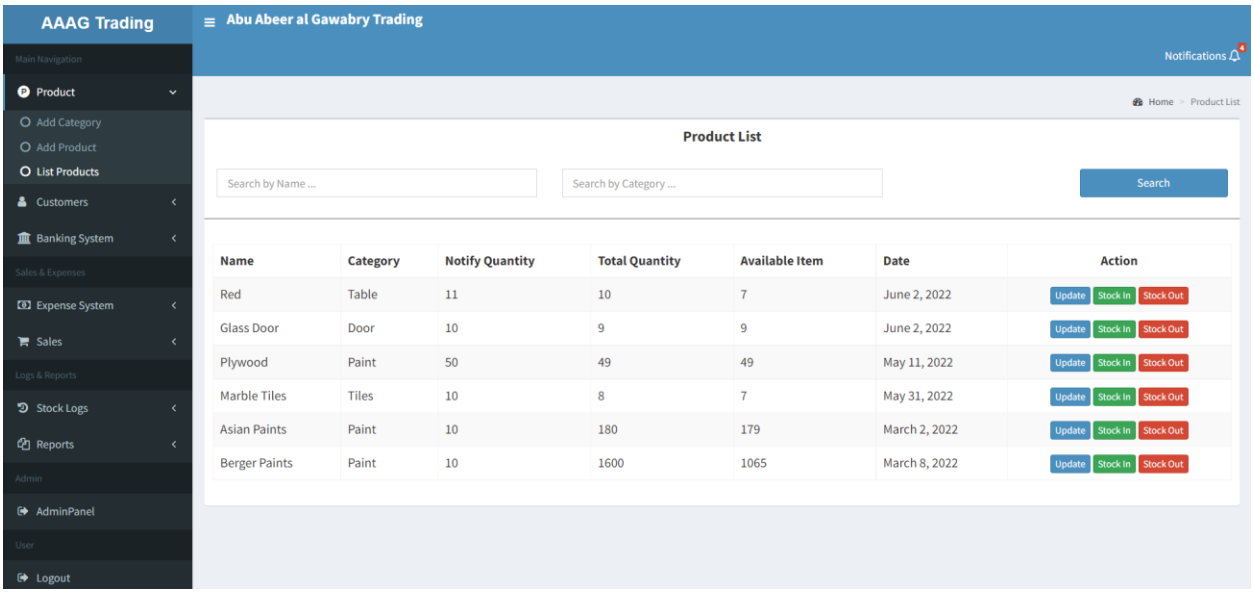


Figure 56: list product

This page shows a list of all the products available including the name, category, notification quantity, total quantity, items available and the date it was added. It also has certain actions the users can take such as updating, stock-in and stock-out.

5.2.5 Update product

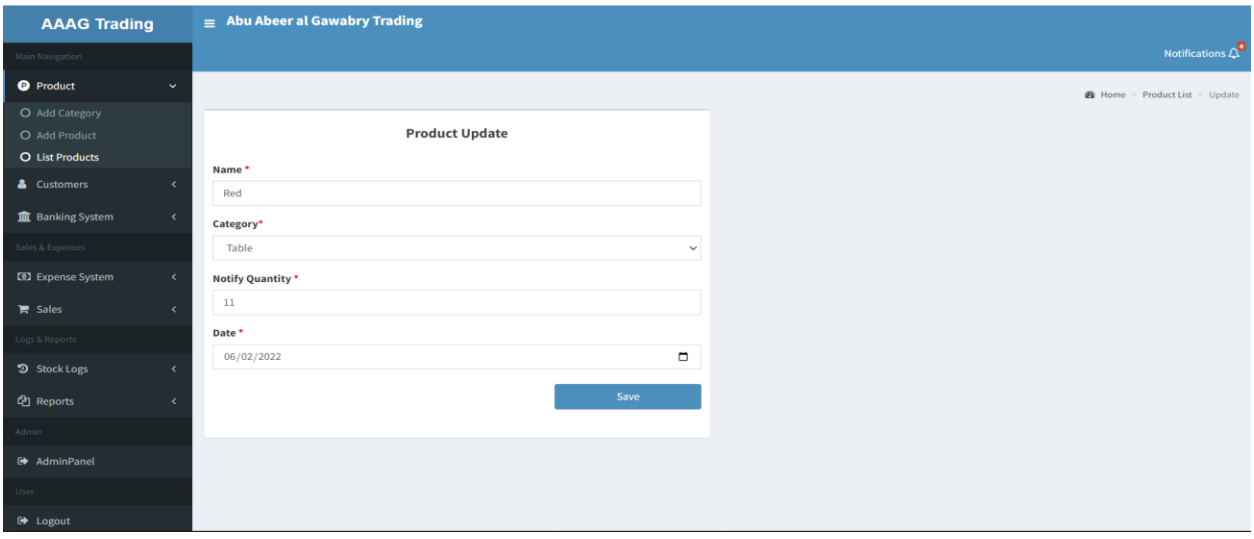


Figure 57:update

The employees can update the products on this page

5.2.6 Stock in

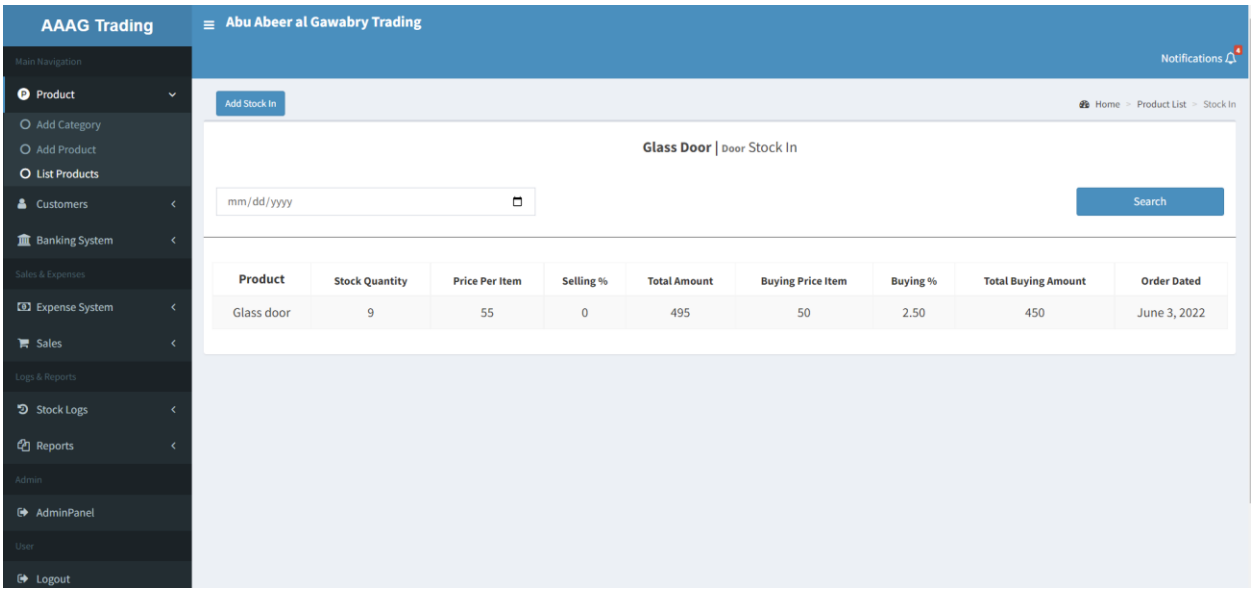


Figure 58: stock in

This page is used to add stock into the system

5.2.7 Stock out product

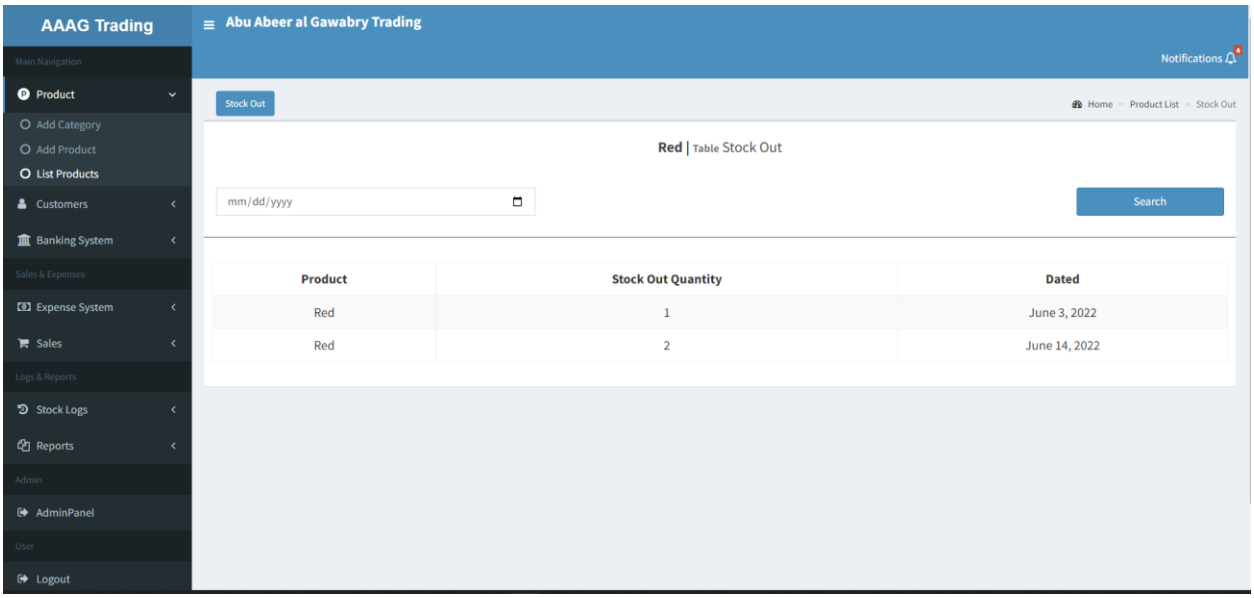


Figure 59: Product stock out page

This page is used to manually remove the existing stock from the inventory

5.2.8 Add customer

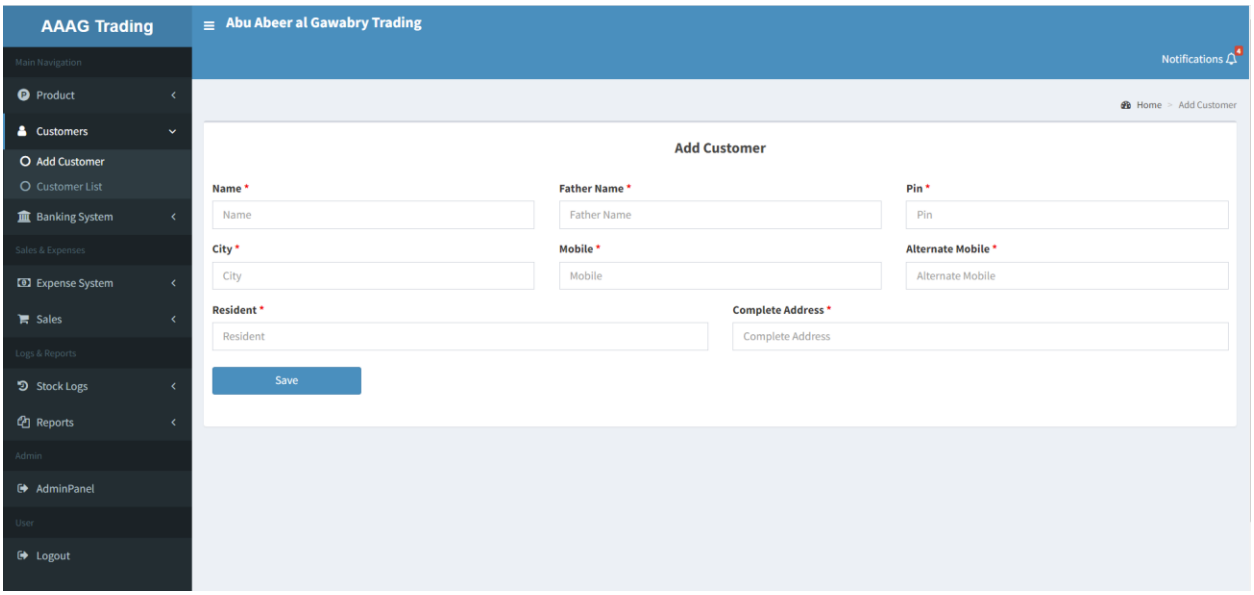


Figure 60: Add customer page

In this page, the users can add customer information such as name, father name, Pin, city, Mobile, alternate mobile, resident and the complete address

5.2.9 List customer

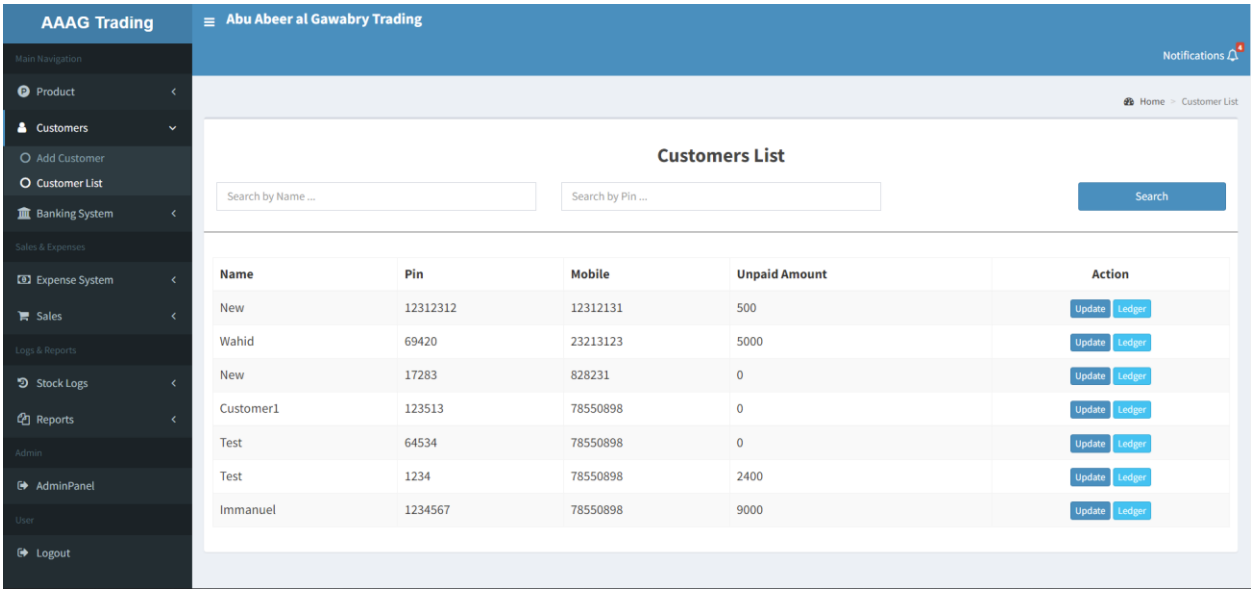


Figure 61: Customer list page

This page is used to show a list of customers that exist along with some actions

5.2.10 Customer ledger

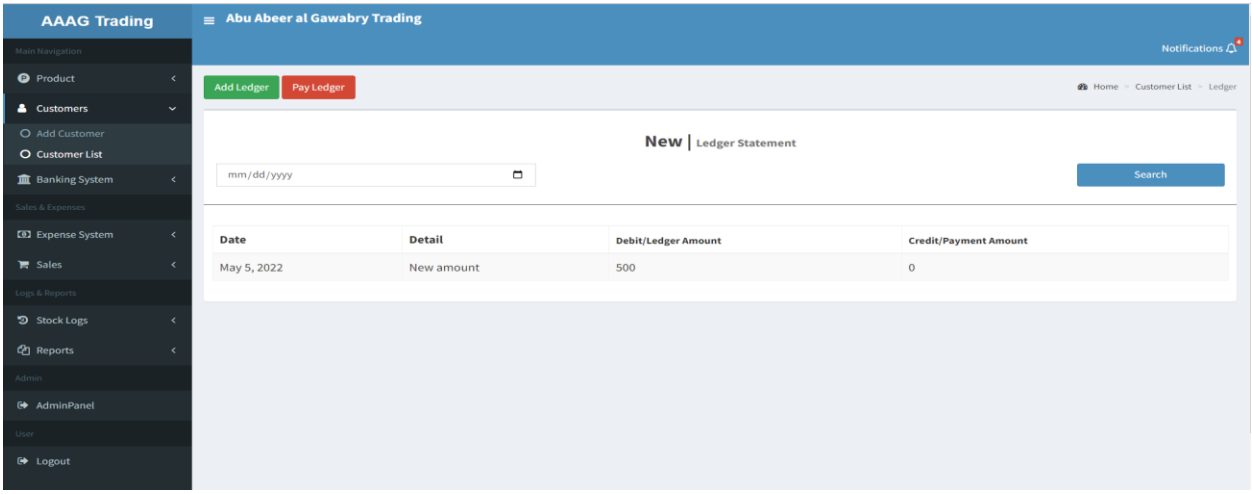


Figure 62:Customer ledger page

This page shows the customer ledger. The employees can add a new ledger or use it to pay ledgers

5.2.11 Add ledger

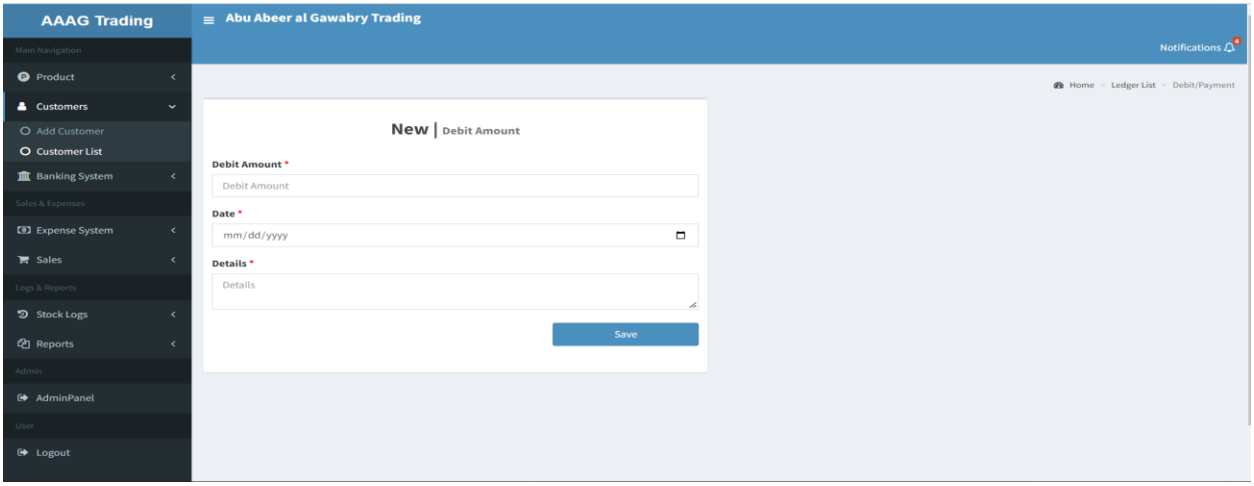


Figure 63: Add ledger page

In the add ledger page, the user can add the debit amount, date of the ledger as well as the details

5.2.12 Pay ledger

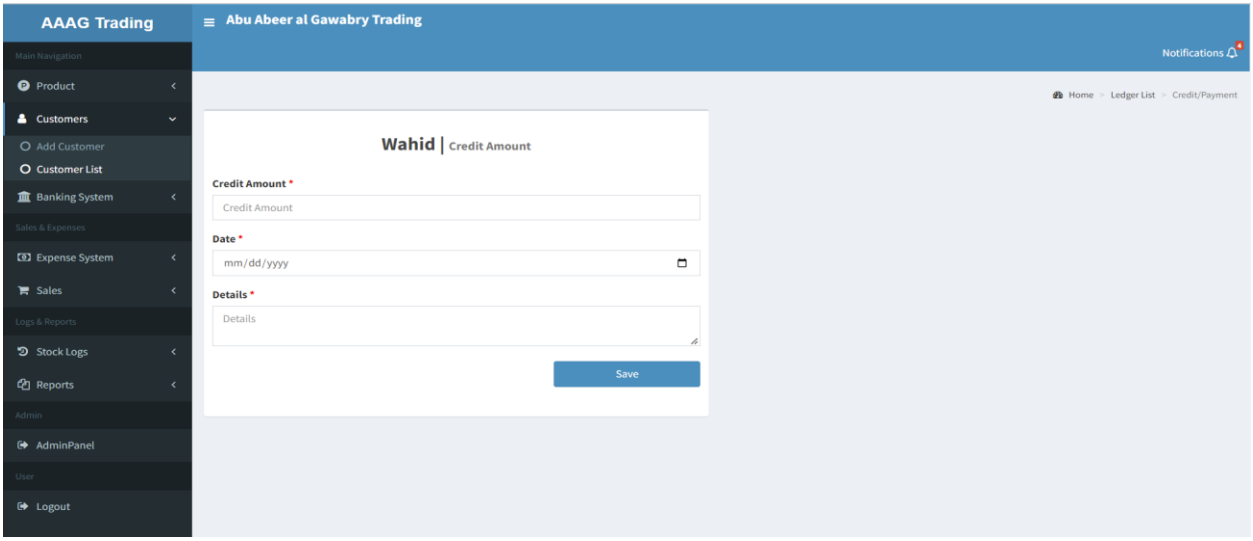


Figure 64: Pay ledger

In the pay ledger screen, the user can add the credit details of the customers.

5.2.13 Add bank

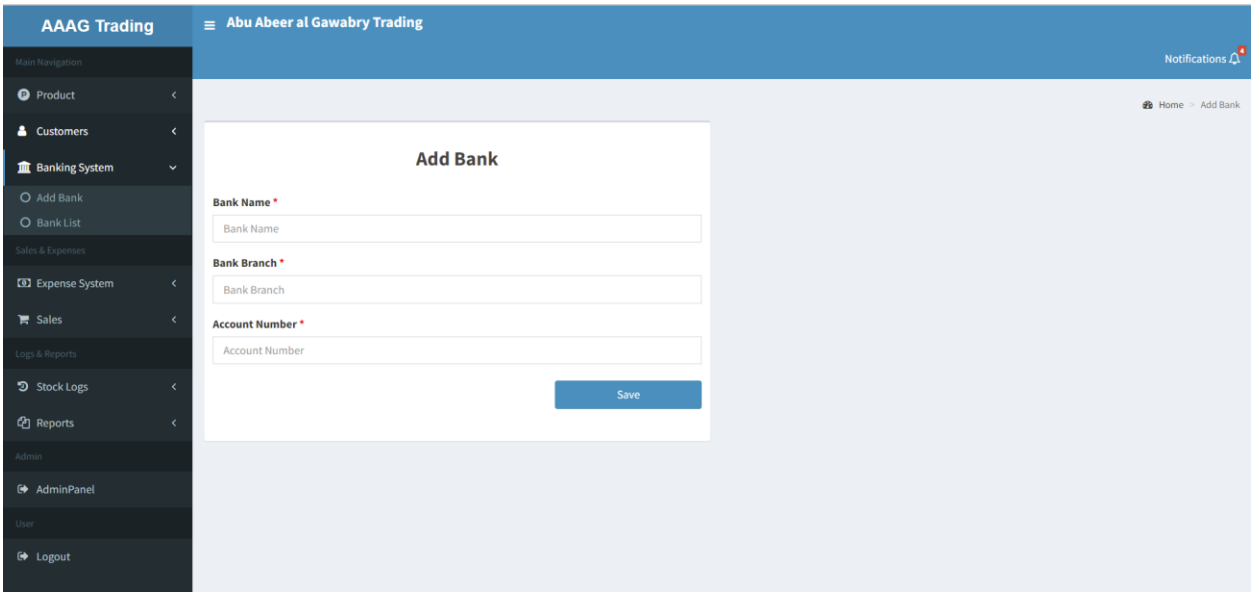


Figure 65: add bank

In this page, the users can add bank information

5.2.14 List bank

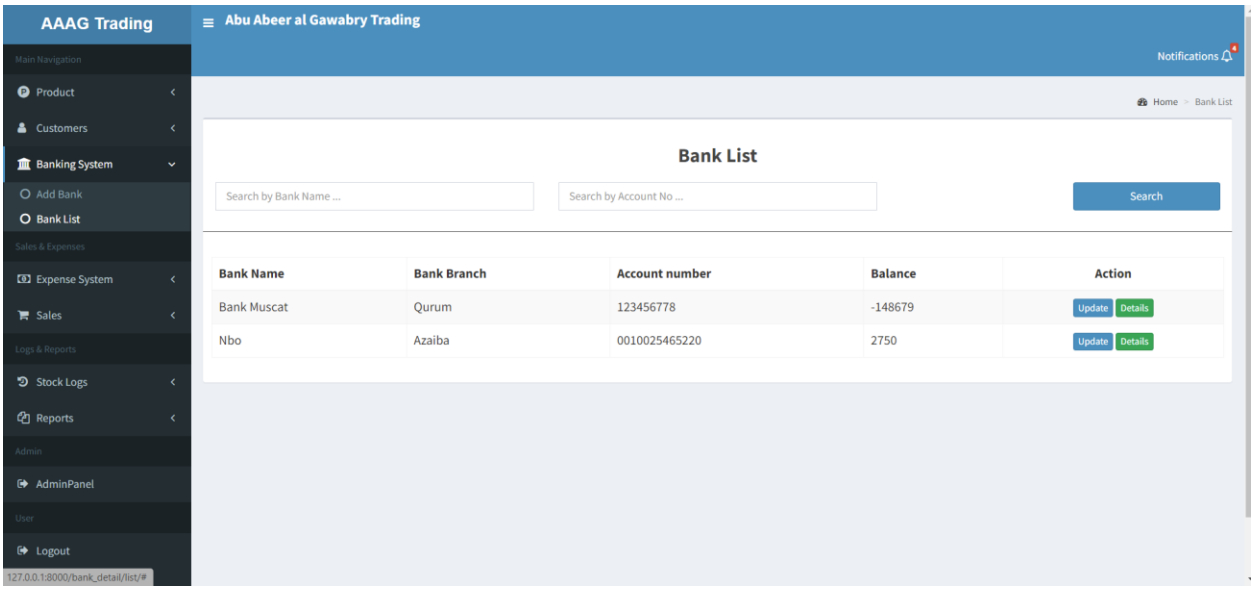


Figure 66: list bank

List of the banks can be views on this page

5.2.15 Update bank

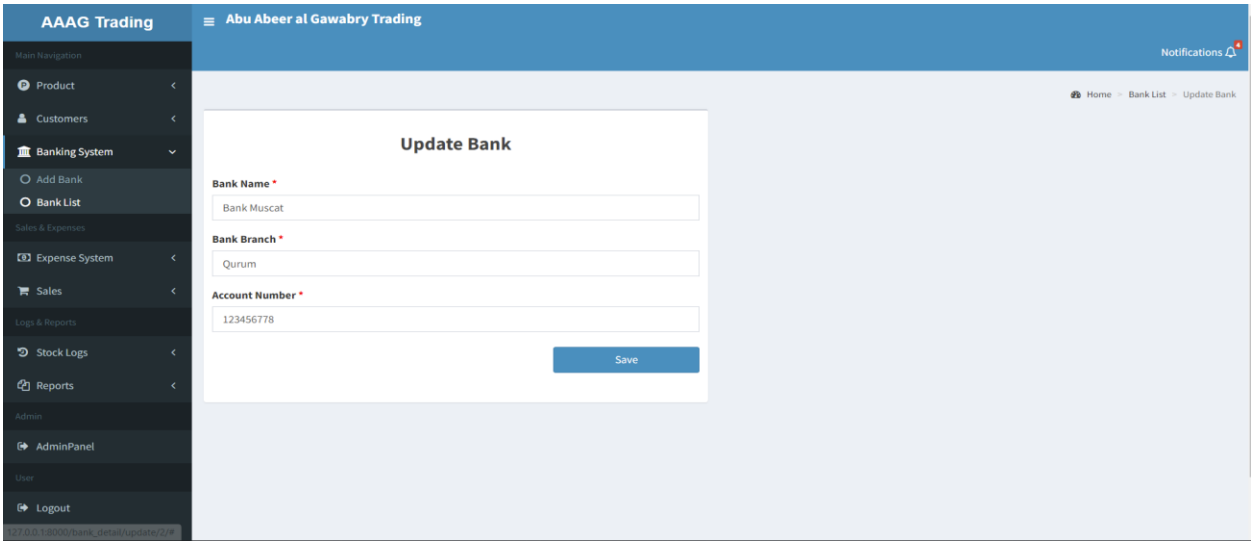


Figure 67: Update bank

Bank details can be viewed on this page

5.2.16 Add debit

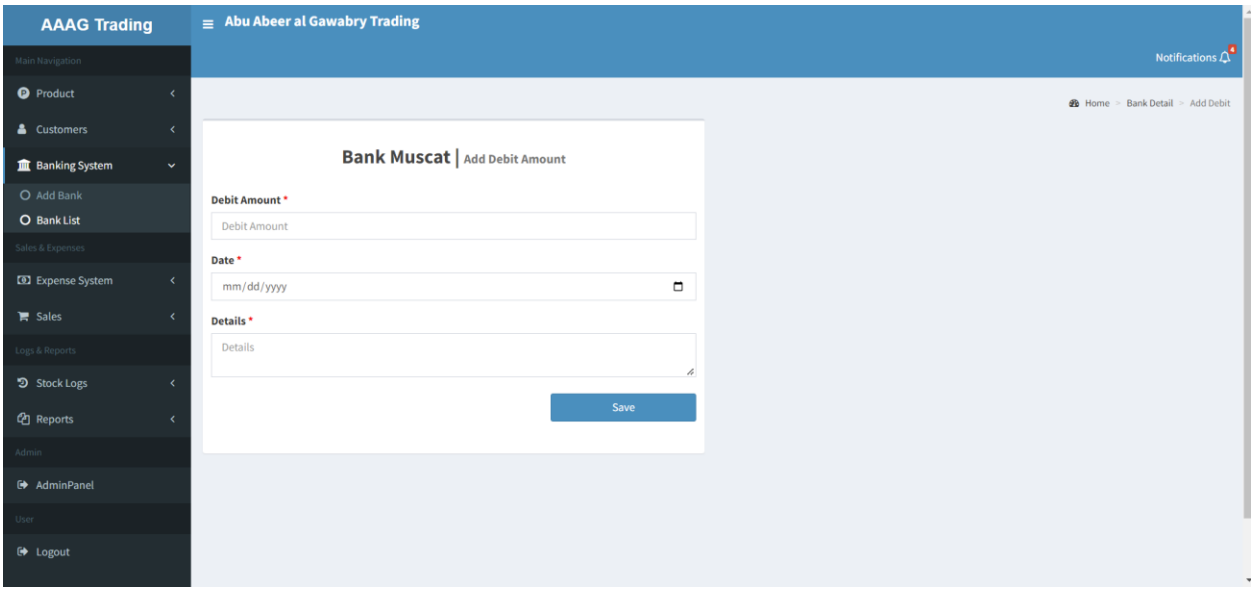


Figure 68: Add debit

Employees and admin/manager can add debit amount in this page including date, and the details.

5.2.17 Add credit

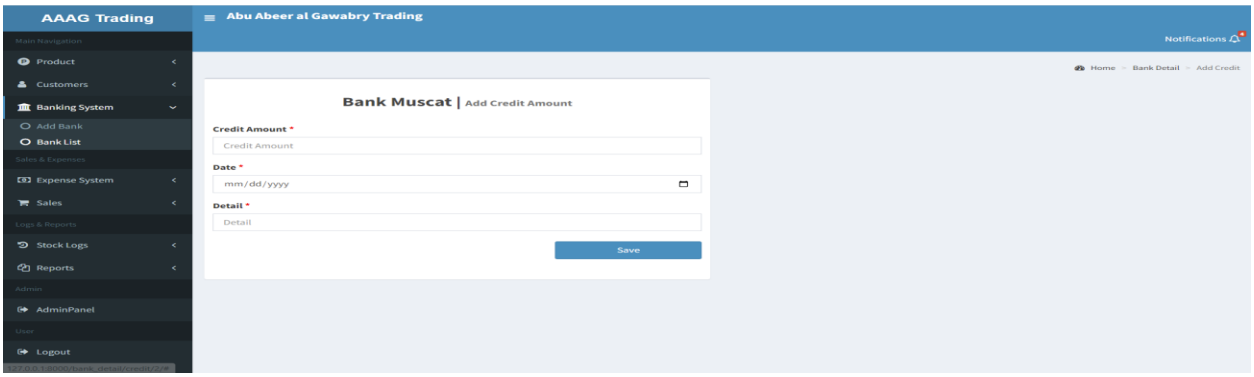


Figure 69: Add credit

Employees and admin/manager can add credit amount in this page including date, and the details.

5.2.18 Add expense

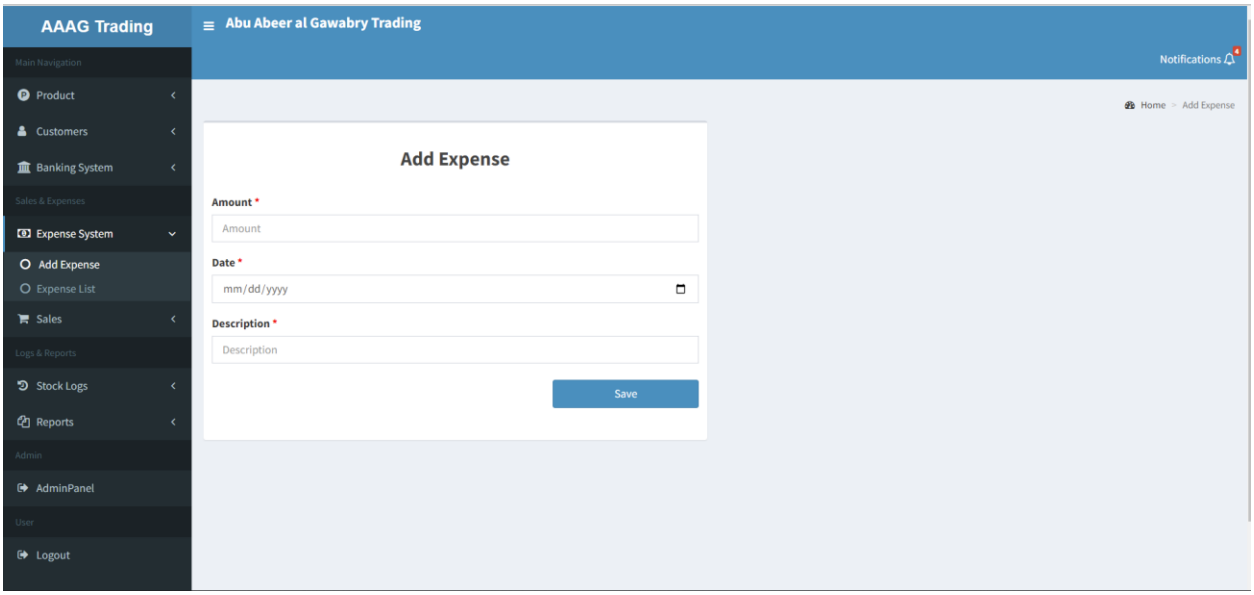


Figure 70: Add expense

In this page, any outside expenses can be added.

5.2.19 Expense list

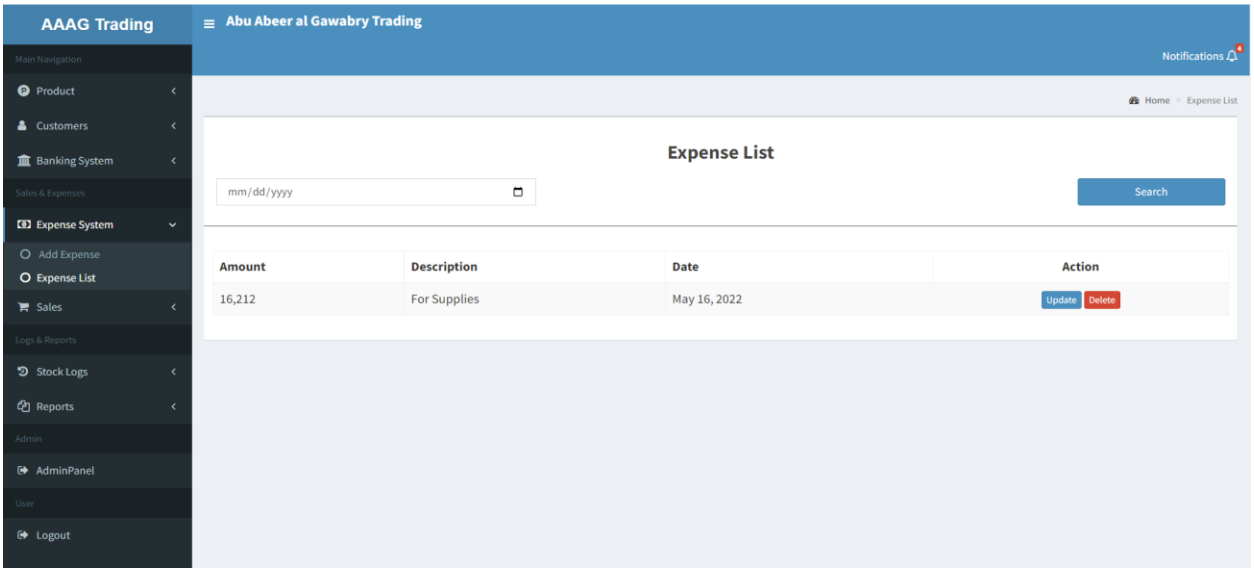


Figure 71: Expense list

This page shows the list of the added expenses

5.2.20 Update expense

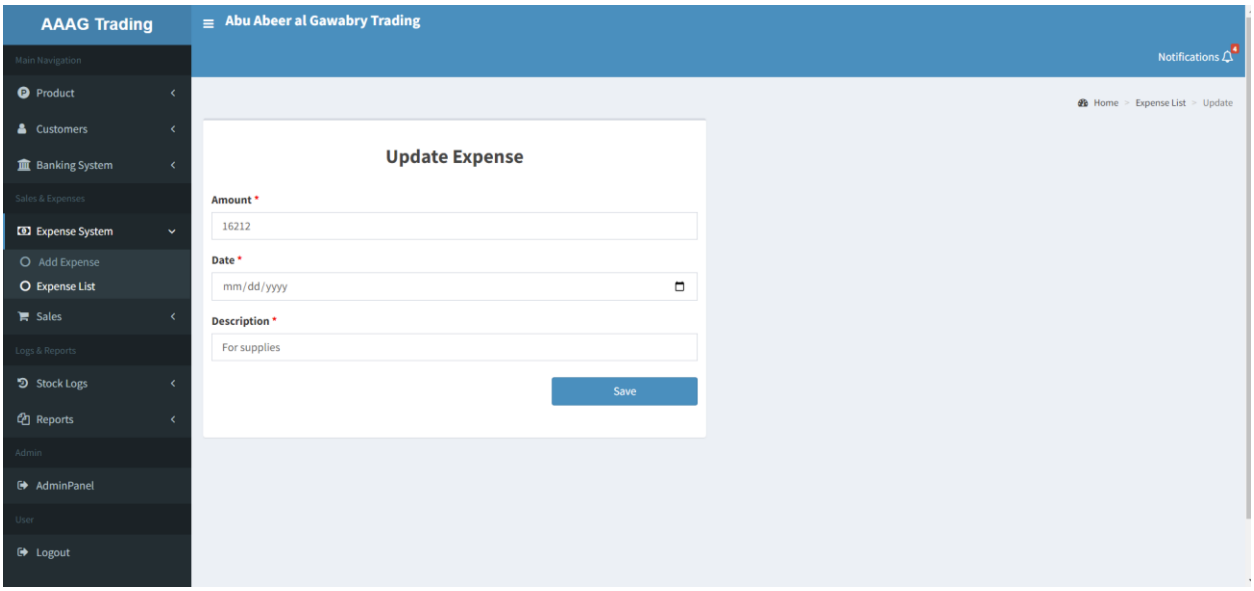


Figure 72: Update expense

Expenses can be updated using this page

5.2.21 Invoice Page

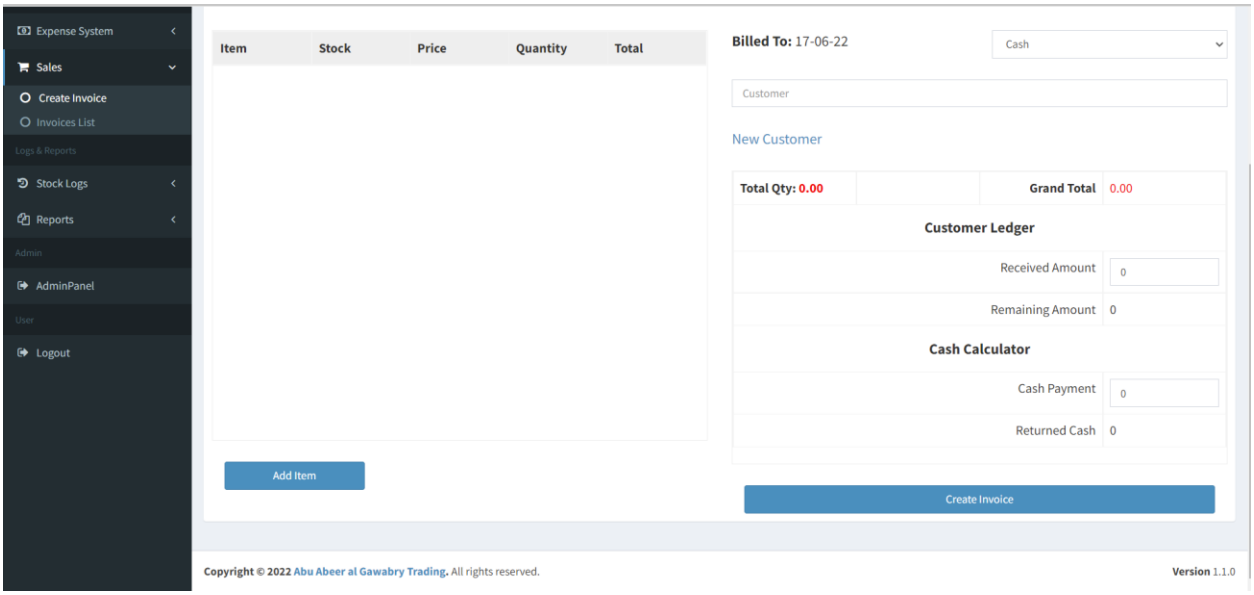


Figure 73: Invoice page

The purchase invoice can be created on this page. A new customer can also be created here. There is an option to choose between installments and cash

5.2.22 Invoice list

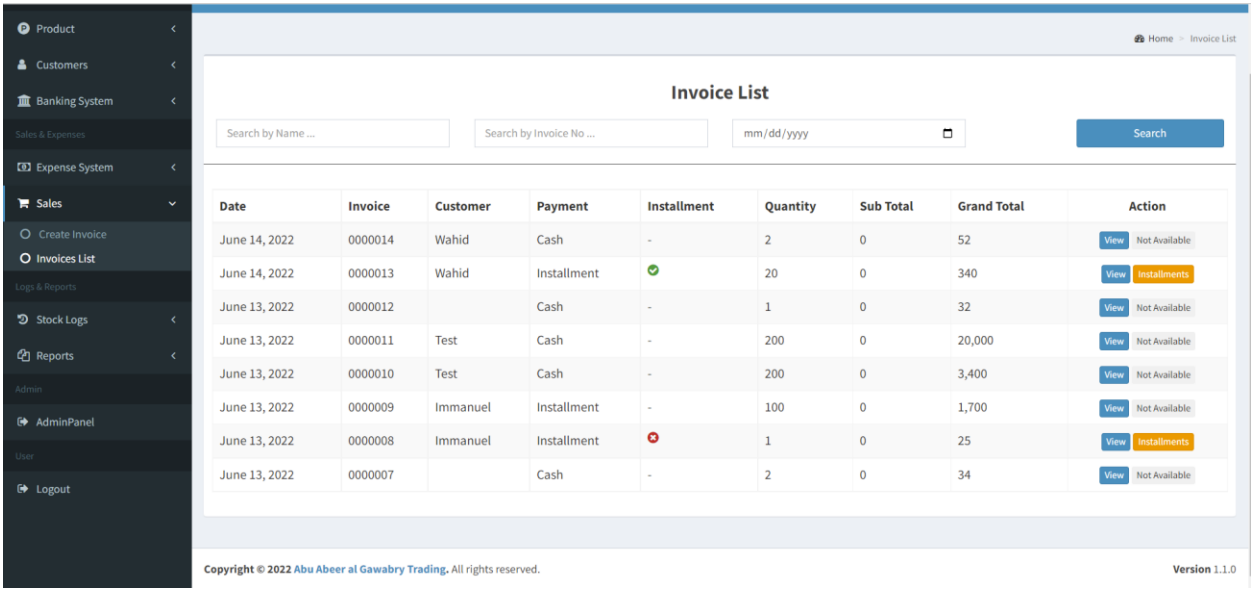


Figure 74: Invoice list

The list of invoices made can be viewed here.

5.2.23 view invoice

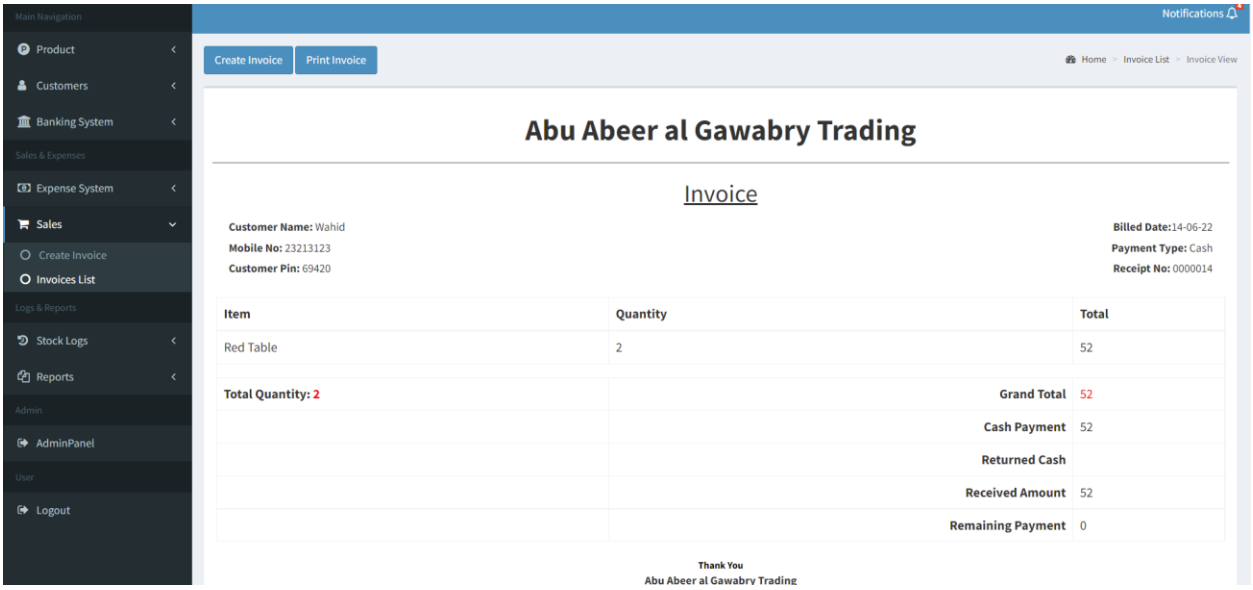


Figure 75: View invoice

The invoices made can be viewed here

5.2.24 Print voice

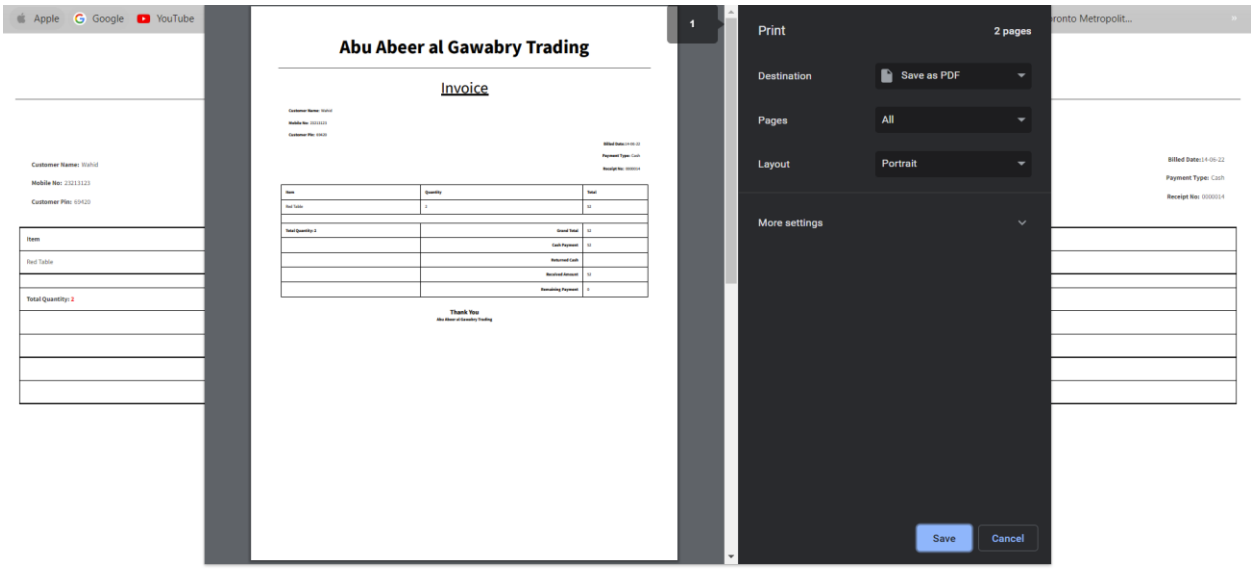


Figure 76: Print invoice

Existing invoice can be printed on this page in a pdf format

5.2.25 View installment

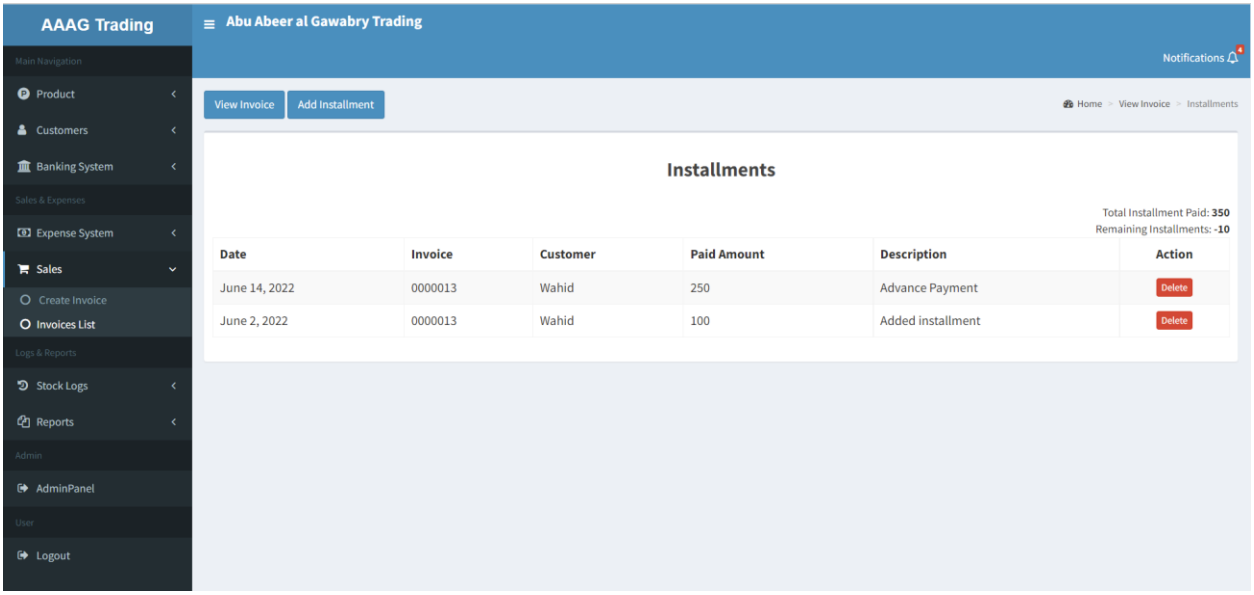


Figure 77: view installment

Customers who have installments can be viewed here.

5.2.26 Add installment

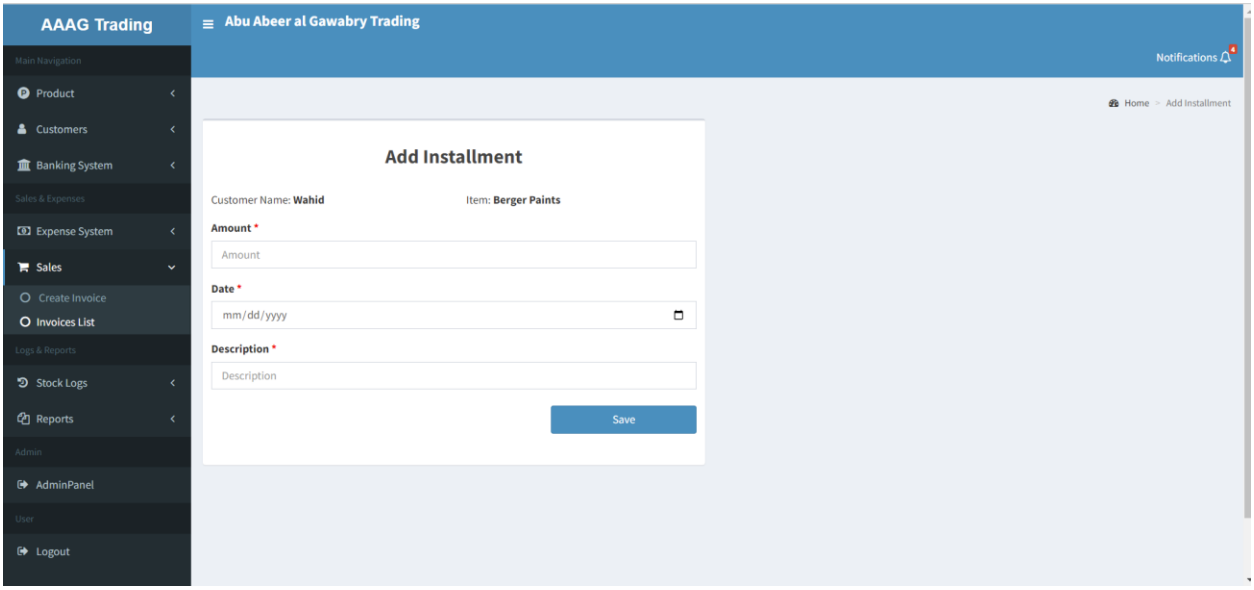


Figure 78: Add installment

New installments can be added here.

5.2.27 Daily logs

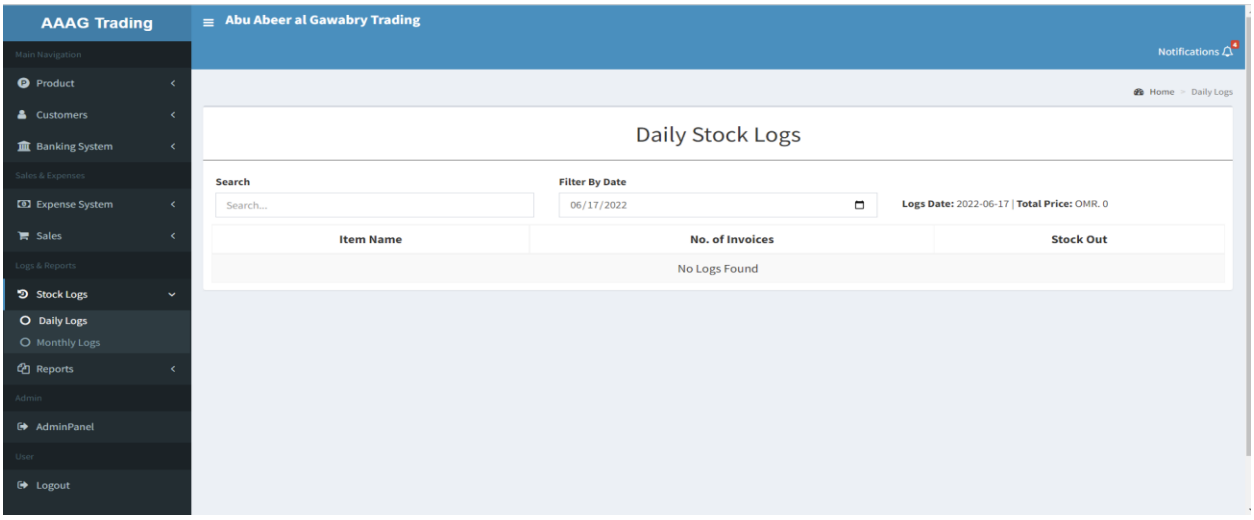


Figure 79: Daily logs

The list of daily logs of sales is available in this page and can be filtered by date and name.

5.2.28 Monthly stocks

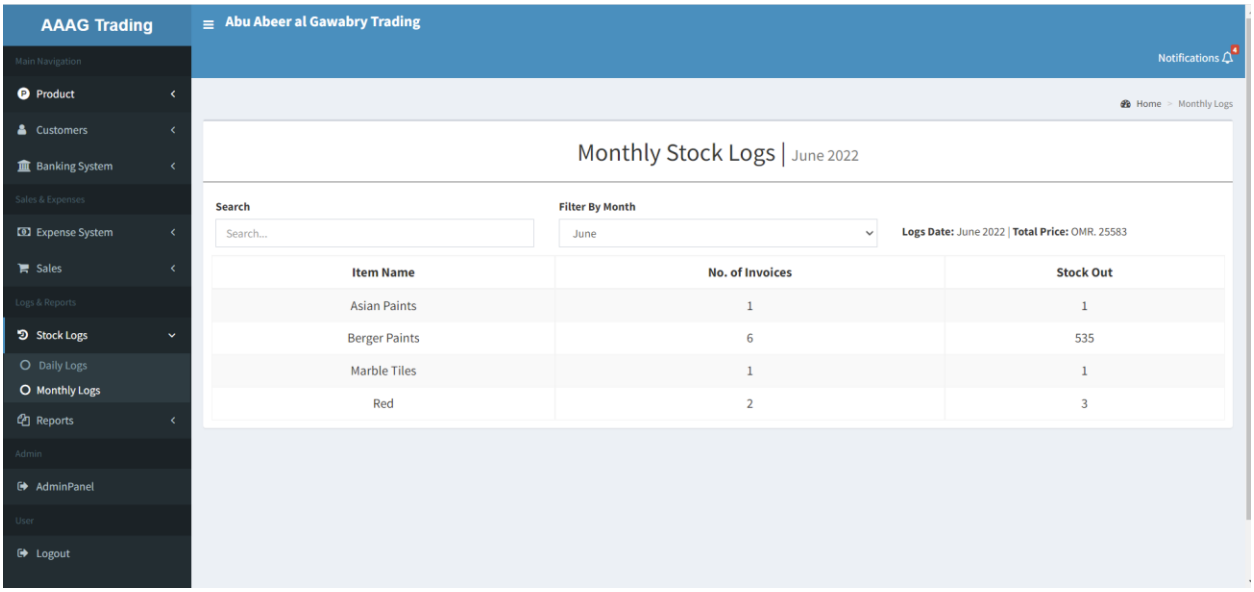


Figure 80: Monthly logs

Sales made in the month can be seen here.

5.2.29 Reports

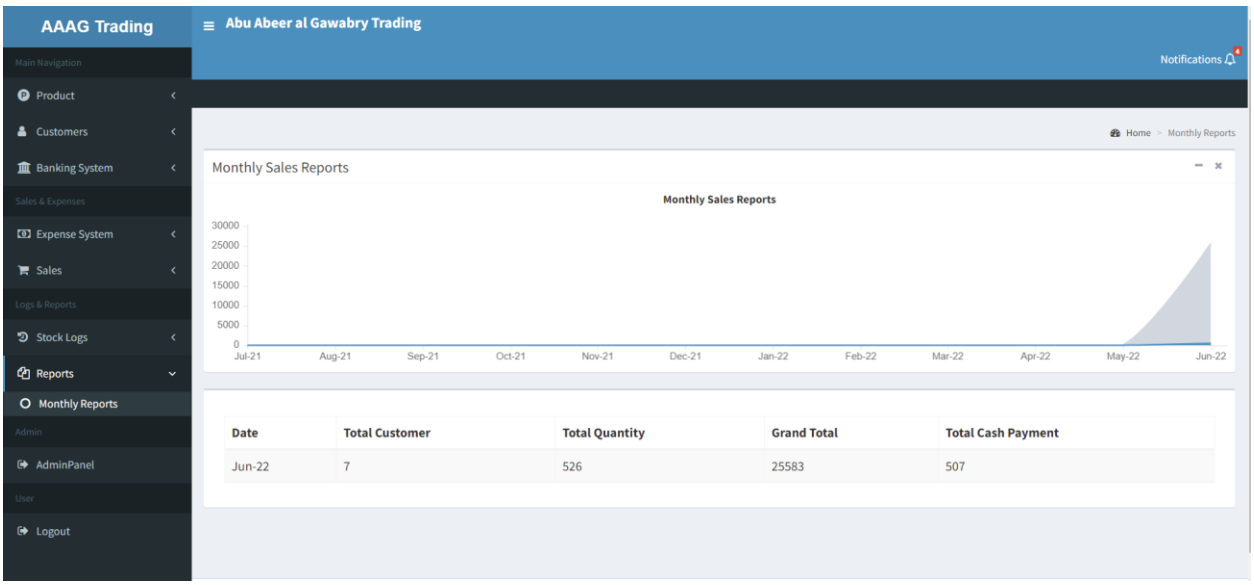


Figure 81: Month reports

This page shows the report of monthly sales along with a sales graph.

5.2.30 Admin panel

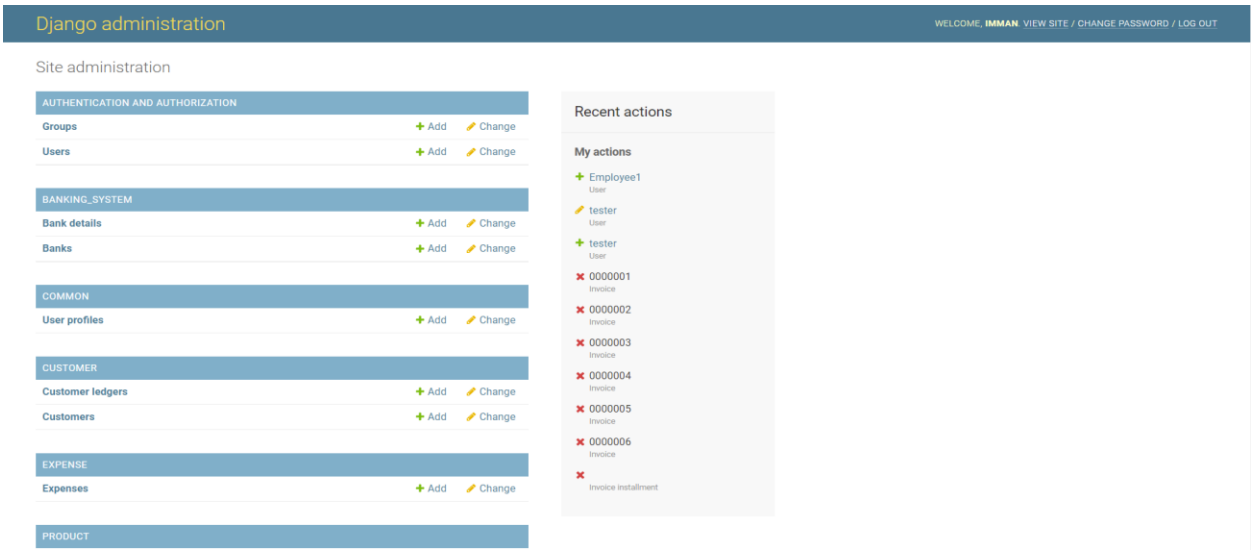


Figure 82: Admin panel

The admin has overall view of the system and can control from the backend. This page allows the admin to add employees, delete employees and give employees permissions

5.3 Testing and evaluation

5.3.1 Importance of testing

Software testing is the process of assessing whether a software system satisfies the anticipated criteria and ensuring that it is defect-free. It consists of putting apps elements through their paces with manual or computerized technologies in order to evaluate one or more characteristics of interest. The objective of software testing is to identify errors, deficiencies, or incomplete specifications when compared to the actual specifications. When working on a software development project, one should be aware that mistakes can occur at any stage of the life cycle (indiamsoftware,2022)

5.3.2 Test strategy

For testing, data for running tests was gathered from system determinations and then utilized to test every feasible part or entry as input. Following each test trial, the true outcome was contrasted with the typical outcome.

This online framework is large and highly regulated, which is why the testing framework methodology was chosen as the testing method to shorten the testing cycle and produce all-around reported results. The system has multiple internet pages, and each page contains multiple connections and verifications that ensure the system's success and productivity.

Test plan

A test plan is a document that specifies the scope, strategy, and timeline for testing operations. The test plan may also include a list of the resources required for the software tester to function efficiently (wearedevelopers,2021)

A test case was developed to make sure the functionality of the system is working wells since the web application has multiple pages and multiple connections that need to be made for it to perform according to the expectations

5.3.3 Test cases

The following are the test cases for the system

Test Case: Employee Functions		
Number	**Date Created**	**Created By**
Tc 001	**14/6/2022**	Immanuel

Login page: Testing the functionalities of login page						
S.NO	**Description**	**Expected Output**	**Expectation**	**Remarks**	**Date**	**By**
1	Enter the correct username and password	Successful login	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta
2	Enter the incorrect username and password	Incorrect Login	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta

Product Page					
Description	**Expected Output**	**Expectation**	**Remarks**	**Date**	**By**
Add product category	Successfully Added	☒As per expected result ☐Not As per expected result	☒Pass ☐ Fail	14/6/2022	Putta

Add Product	Added successfully	☒As per expected result □ Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta
List of products	Shows list of products	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta

Description	**Expected Output**	**Expectation**	**Remarks**	**Date**	**By**
Add Product	Successfully Added	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta
Update Product	Product update	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta

Customer Page					
Description	**Expected Output**	**Expectation**	**Remarks**	**Date**	**By**
Add Customer Details	Successfully Added	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta
View Customer list	Can view list	☒As per expected result ☐Not As per expected result	☒Pass ☐ Fail	14/6/2022	Putta
Bank Page					
Expected Output	**Expectation**	**Remarks**	**Date**	**By**	
Add Bank information	☒As per expected result ☐Not As per expected result	☒ Pass ☐ Fail	14/6/2022	Putta	
View Bank List	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta	
Notification					
Expected Output	**Expectation**	**Remarks**	**Date**	**By**	
Notification if an item has low stock	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta	

Expense System				
Expected Output	**Expectation**	**Remarks**	**Date**	**By**
Add expense	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta
Expense List	☒As per expected result ☐Not As per expected result	☒ Pass ☐ Fail	14/6/2022	Putta
Sales Page				
Expected Output	**Expectation**	**Remarks**	**Date**	**By**
Create invoice	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta
List of invoices	☒As per expected result ☐Not As per expected result	☒ Pass ☐ Fail	14/6/2022	Putta
Stock Logs				
Expected Output	**Expectation**	**Remarks**	**Date**	**By**
Daily logs	☒As per expected result ☐Not As per expected result	☒ Pass ☐Fail	14/6/2022	Putta

Monthly logs	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta
Reports				
Expected Output	**Expectation**	**Remarks**	**Date**	**By**
View reports	☒As per expected result ☐Not As per expected result	☒Pass ☐ Fail	14/6/2022	Putta
Notifications				
Expected Output	**Expectation**	**Remarks**	**Date**	**By**
Show low stock notifications	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta

Test Case: Admin Functions		
Number	**Date Created**	**Created By**
Tc 002	**28/5/2022**	Immanuel

Testing functionalities of system Admin

S.NO	Description	Expected Output	Expectation	Remarks	Date	By
1	Change user Password	Changed user password	☒As per expected result ☐ Not As per expected	☒Pass ☐Fail	14/6/2022	Putta
2	Edit user information	User information changed	☒As per expected result ☐Not As per expected	☒Pass ☐Fail	14/6/2022	Putta
3	Change user permissions	Changed user permissions	☒ As per expected result ☐Not As per expected result	☒ Pass ☐Fail	14/6/2022	Putta
4	Change user status	Changed user status	☒As per expected result ☐Not expected	☒Pass ☐ Fail	14/6/2022	Putta

Product Page					
Description	**Expected Output**	**Expectation**	**Remarks**	**Date**	**By**
Add product category	Successfully Added	☒As per expected result ☐Not As per expected result	☒Pass ☐ Fail	14/6/2022	Putta
Delete Product	Product Deleted	☒ As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta

Description	**Expected Output**	**Expectation**	**Remarks**	**Date**	**By**
Add Product	Successfully Added	☒As per expected result ☒Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta
Update Product	Order deleted	☒ As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta
Customer Page					
Description	**Expected Output**	**Expectation**	**Remarks**	**Date**	**By**

Add Customer Details	Successfully Added	☒As per expected result ☐Not As per expected result	☒ Pass ☐Fail	14/6/2022	Putta
View Customer list	Can view list	☒As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta

Bank Page				
Expected Output	**Expectation**	**Remarks**	**Date**	**By**
Add Bank information	☒ As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta
View Bank List	☒ As per expected result ☐Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta

Expense System				
Expected Output	**Expectation**	**Remarks**	**Date**	**By**
Add expense	☒As per expected result ☐ Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta
Expense List	☒ As per expected result		14/6/2022	Putta

	☐Not As per expected result	☒Pass ☐Fail		
Sales Page				
Expected Output	**Expectation**	**Remarks**	**Date**	**By**
Create invoice	☒As per expected result ☐ Not As per expected result	☒Pass ☐ Fail	14/6/2022	Putta
List of invoices	☒ As per expected result ☐ Not As per expected result	☒ Pass ☐Fail	14/6/2022	Putta
Stock Logs				
Expected Output	**Expectation**	**Remarks**	**Date**	**By**
Daily logs	☒As per expected result ☐Not As per expected result	☒Pass ☐ Fail	14/6/2022	Putta
Monthly logs	☒ As per expected result ☐ Not As per expected result	☒Pass ☐Fail	14/6/2022	Putta

5.3.4 Analyzing the results

From the above test cases, it is evident that the web application works as its intended to. This makes the system ready for deployment. All the proposed functionality of the system tends to be working perfectly as expected.

5.4 Deployment

5.4.1 Requirement analysis

Client requirements: The minimum requirement for this web application is an internet browser that can support complex HTML such as Google Chrome, Safari, Edge.

The web application is not software heavy but an operating system of Windows 7 professional or above is preferred.

Training requirements for users: This is a simple inventory system that does not require many skills to use. However, an average skill in using computer devices and navigation skill is preferred.

5.5 Critical evaluation of the system

The benefit of computerized sales and inventory is that it makes the process of documenting sales and inventory more efficient and precise. This results in an effective and efficient sales and inventory system capable of correctly monitoring the movement of items from sales and inventories. Such administration may be done manually for a small firm with few items; but, as the size of the business expands and the list of products grows, such management becomes more difficult. It becomes difficult and time-consuming to keep track of sales and inventories. That is why researchers are working to create a computerized point of sale and inventory system solution that can efficiently manage sales and inventory system duty.

The price of a sales and inventory system is sometimes prohibitively high, and pricing is typically a concern for any system, but this system is not too expensive. Sales and inventories facilitate company operations. Understanding the fundamentals might be tough and time-consuming, especially if you don't have a firm grasp on a computer.

Chapter 6: Conclusions and recommendations

6.1 Conclusion

The primary aim of the project was to design and implement an inventory material management system for, and organization called Abu Abeer al Gawabry trading.

The aim of the project was to give the company a new system that will be used to add item category, add product, view list of products, edit the list of products with more functions such as stock in, stock out, add customer information, update customer information, banking system, expense system, sales, stocks, and reports.

All the objectives and the requirements have been met successfully and it is ready for deployment

6.1.2 Evaluating the contribution of the project

The project has addressed the problem of the need of an industry grade inventory management system for a company in Oman that is based on the trade of construction materials. The system consists of various tool and packages that help put it together.

The system is not only limited to construction materials but can also be used for medical records, Human resource and anything that requires the data to be stored.

6.2 Recommendations

This project can be developed further in so many ways since it is developed using python and Django.

- **Prediction of sales:** Prediction of sales based on the date can be a very good way for knowing whether an item will sell good or not
- **Enhanced security:** two factor authorization can be added to make the system even more secure
- **Barcode scanning:** This will make it easier to search for products when the inventory system gets a lot if items and makes it easier instead of the simple search function
- **Multiplatform compatibility:** The web application is limited to web browsers only. There will be future plans to develop it to mobile application and tablets.

6.3 Limitations

- Tracking product is inconsistent: in the construction business, there are different building materials that have different sized and code. That is not applied in this system. There are future plans to add this.
- Currently the web application only works on PC. It does not work on mobiles or any other devices. There are future plans to make this work

6.4 Impact of the project

The Technology Impact attributable to the Project, as estimated upon completion of the Project, is referred to as the Project Impact. Both positive and negative influences within the Project Assessment Boundary must be considered when determining the Project Impact (Lawinsider, 2020)

6.4.1 Social impact

This project will be helpful to any company or person trying to store data regarding sales or analysis. The system is not only limited to one business. It can be used in hospitals and other businesses too.

6.4.2 Ethical impact

The project is an inventory management whose sole purpose is to manage inventory.

The sensitive details about the customer and the users will be stored. This should not be manipulated. Manipulation of inventory figures and levels when a client questions his inventory levels or inquiries about inventory management circumstances.

All the information about the users stored in the system is safe and protected by Django's safety protocols. Only the admin of the system has access to user passwords and has the capability to change the personal information upon demand.

6.4.3 Legal impact

The resources for making the project have been taken from online sources and research papers are given credits accordingly. Most of the templates required to make the project have been taken from online sources.

APPENDICES

1. Codes:

1. Back end

Main Admin

```
from django.contrib import admin

from banking_system.models import BankDetail, Bank

class BankAdmin(admin.ModelAdmin):

    list_display = (

        'name', 'branch', 'account_number'

    )

class BankDetailAdmin(admin.ModelAdmin):

    list_display = (

        '__str__', 'debit', 'credit', 'invoice', 'description', 'date'

    )

admin.site.register(Bank, BankAdmin)

admin.site.register(BankDetail, BankDetailAdmin)
```

Main Apps code

```
from django.apps import AppConfig

class BankingSystemConfig(AppConfig):

    name = 'banking_system'
```

Main Models code

```
from django.db import models

from django.db.models import Sum

from django.utils import timezone

class Bank(models.Model):

    name = models.CharField(max_length=200)

    branch = models.CharField(max_length=200, null=True, blank=True)

    account_number = models.CharField(max_length=200)

    def __str__(self):

        return self.name

    def bank_balance(self):

        bank_details = self.bank_detail.all()

        if bank_details:

            debit = bank_details.aggregate(Sum('debit'))

            credit = bank_details.aggregate(Sum('credit'))

            debit_amount = debit.get('debit__sum')

            credit_amount = credit.get('credit__sum')

        else:

            debit_amount = 0

            credit_amount = 0

        balance = credit_amount - debit_amount

        return balance
```

```
class BankDetail(models.Model):
    bank = models.ForeignKey(Bank, on_delete=models.CASCADE, related_name='bank_detail',
                             null=True, blank=True)
    invoice = models.ForeignKey(
        'sales.Invoice', related_name='bank_invoice'
        , null=True, blank=True, on_delete=models.SET_NULL
    )
    debit = models.DecimalField(max_digits=65, decimal_places=2, default=0,
                                null=True, blank=True)
    credit = models.DecimalField(max_digits=65, decimal_places=2, default=0,
                                 null=True, blank=True)
    description = models.TextField(max_length=500, null=True, blank=True)
    date = models.DateField(default=timezone.now, null=True, blank=True)
    def __str__(self):
        return self.bank.name if self.bank else ''
```

Main URLs Code

```
from django.urls import path
from banking_system.views import (
    AddBankFormView, BankListView, BankDetailListView, AddBankUpdateView,
    CreditBankFormView, CreditBankUpdateView, DebitBankFormView, DebitBankUpdateView
)
```

```
urlpatterns = [
    path('add/', AddBankFormView.as_view(), name='add'),
    path('list/', BankListView.as_view(), name='list'),
    path('detail_list/<int:pk>/', BankDetailListView.as_view(), name='detail_list'),
    path('update/<int:pk>/', AddBankUpdateView.as_view(), name='update'),
    path('credit/<int:pk>/', CreditBankFormView.as_view(), name='credit'),
    path('credit/update/<int:pk>/', CreditBankUpdateView.as_view(), name='credit_update'),
    path('debit/<int:pk>/', DebitBankFormView.as_view(), name='debit'),
    path('debit/update/<int:pk>/', DebitBankUpdateView.as_view(), name='debit_update'),
]
```

Main views code

```
from django.shortcuts import render
from banking_system.forms import BankForm, BankDetailForm
from banking_system.models import Bank, BankDetail
from django.views.generic import ListView, FormView, UpdateView
from django.http import HttpResponseRedirect
from django.urls import reverse
from django.core.exceptions import ObjectDoesNotExist
from django.http import Http404
class AddBankFormView(FormView):
    form_class = BankForm
    template_name = 'banking/add_bank.html'
```

```
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            AddBankFormView, self).dispatch(request, *args, **kwargs)
    def form_valid(self, form):
        form.save()
        return HttpResponseRedirect(reverse('bank:list'))
    def form_invalid(self, form):
        return super(AddBankFormView, self).form_invalid(form)
class BankListView(ListView):
    model = Bank
    template_name = 'banking/bank_list.html'
    paginate_by = 100
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            BankListView, self).dispatch(request, *args, **kwargs)
    def get_queryset(self):
        queryset = self.queryset
        if not queryset:
            queryset = Bank.objects.all().order_by('-id')
```

```
        if self.request.GET.get('bank_name'):
            queryset = queryset.filter(
                name__contains=self.request.GET.get('bank_name'))
        if self.request.GET.get('account_no'):
            queryset = queryset.filter(
                account_number=self.request.GET.get('account_no').lstrip('0')
            )
        return queryset.order_by('-id')
class AddBankUpdateView(UpdateView):
    model = Bank
    form_class = BankForm
    template_name = 'banking/update_add_bank_list.html'
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            AddBankUpdateView, self).dispatch(request, *args, **kwargs)
    def form_valid(self, form):
        form.save()
        return HttpResponseRedirect(reverse('bank:list'))
    def form_invalid(self, form):
        return super(AddBankUpdateView, self).form_invalid(form)
```

```
    def get_context_data(self, **kwargs):
        context = super(AddBankUpdateView, self).get_context_data(**kwargs)
        bank = Bank.objects.all()
        context.update({
            'bank': bank
        })
        return context
class BankDetailListView(ListView):
    model = BankDetail
    template_name = 'banking/bank_detail_list.html'
    paginate_by = 100
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            BankDetailListView, self).dispatch(request, *args, **kwargs)
    def get_queryset(self, **kwargs):
        queryset = self.queryset
        if not queryset:
            queryset = self.model.objects.filter(
                bank__id=self.kwargs.get('pk')).order_by('-date')
        if self.request.GET.get('date'):
            queryset = queryset.filter(
```

```
                date__icontains=self.request.GET.get('date')
            )
        return queryset
    def get_context_data(self, **kwargs):
        context = super(BankDetailListView, self).get_context_data(**kwargs)
        try:
            bank = Bank.objects.get(id=self.kwargs.get('pk'))
        except Bank.DoesNotExist:
            raise Http404('Bank does not exits!')
        context.update({
            'bank': bank
        })
        return context


class DebitBankFormView(FormView):
    template_name = 'banking/debit.html'
    form_class = BankDetailForm
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            DebitBankFormView, self).dispatch(request, *args, **kwargs)
```

```
    def form_valid(self, form):
        obj = form.save()
        return HttpResponseRedirect(reverse('bank:detail_list',
                                    kwargs={'pk': obj.bank.id}))
    def form_invalid(self, form):
        print(form.errors)
        print("hiiiiiii")
        return super(DebitBankFormView, self).form_invalid(form)
    def get_context_data(self, **kwargs):
        context = super(
            DebitBankFormView, self).get_context_data(**kwargs)
        try:
            bank = Bank.objects.get(id=self.kwargs.get('pk'))
        except Bank.DoesNotExist:
            raise Http404('Bank does not exits!')
        context.update({
            'bank': bank
        })
        return context
class DebitBankUpdateView(UpdateView):
    model = BankDetail
    form_class = BankDetailForm
    template_name = 'banking/update_debit.html'
```

```
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            DebitBankUpdateView, self).dispatch(request, *args, **kwargs)
    def form_valid(self, form):
        obj = form.save()
        return HttpResponseRedirect(reverse('bank:detail_list',
                                            kwargs={'pk': obj.bank.id}))
    def form_invalid(self, form):
        print(form.errors)
        print('hi')
        return super(DebitBankUpdateView, self).form_invalid(form)
class CreditBankFormView(DebitBankFormView):
    template_name = 'banking/credit.html'
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))


        return super(
            CreditBankFormView, self).dispatch(request, *args, **kwargs)
class CreditBankUpdateView(DebitBankUpdateView):
    template_name = 'banking/update_credit.html'
```

```
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            CreditBankUpdateView, self).dispatch(request, *args, **kwargs)
```

Customer views code

```
from django.shortcuts import render
from django.views.generic import ListView, FormView, UpdateView
from django.http import HttpResponseRedirect
from django.urls import reverse
from django.core.exceptions import ObjectDoesNotExist
from django.http import Http404
from customer.models import Customer, CustomerLedger
from customer.forms import CustomerForm, CustomerLedgerForm



class AddCustomer(FormView):
    form_class = CustomerForm
    template_name = 'customer/add_customer.html'
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
```

```
        return super(
            AddCustomer, self).dispatch(request, *args, **kwargs)
    def form_valid(self, form):
        form.save()
        return HttpResponseRedirect(reverse('customer:list'))
    def form_invalid(self, form):
        return super(AddCustomer, self).form_invalid(form)
class CustomerList(ListView):
    model = Customer
    template_name = 'customer/customer_list.html'
    paginate_by = 100
    ordering = '-id'

    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))

        return super(
            CustomerList, self).dispatch(request, *args, **kwargs)

    def get_queryset(self):
        queryset = self.queryset
        if not queryset:
```

```
            queryset = Customer.objects.all().order_by('-id')
        if self.request.GET.get('customer_name'):
            queryset = queryset.filter(
                name__icontains=self.request.GET.get('customer_name'))
        if self.request.GET.get('customer_id'):
            queryset = queryset.filter(
                cnic=self.request.GET.get('customer_id').lstrip('0')
            )
        return queryset.order_by('-id')
class UpdateCustomer(UpdateView):
    model = Customer
    form_class = CustomerForm
    template_name = 'customer/update_customer.html'
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))


        return super(
            UpdateCustomer, self).dispatch(request, *args, **kwargs)
    def form_valid(self, form):
        form.save()
        return HttpResponseRedirect(reverse('customer:list'))
```

```
    def form_invalid(self, form):
        return super(UpdateCustomer, self).form_invalid(form)
    def get_context_data(self, **kwargs):
        context = super(UpdateCustomer, self).get_context_data(**kwargs)
        customer = Customer.objects.all()
        context.update({
            'customer': customer
        })
        return context
class CustomerLedgerListView(ListView):
    model = CustomerLedger
    template_name = 'customer_ledger/ledger_list.html'
    paginate_by = 100

    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))

        return super(
            CustomerLedgerListView, self).dispatch(request, *args, **kwargs)

    def get_queryset(self, **kwargs):
```

```
    queryset = self.queryset
    if not queryset:
        queryset = self.model.objects.filter(
            customer__id=self.kwargs.get('pk')).order_by('-date')

    if self.request.GET.get('date'):
        queryset = queryset.filter(
            date__icontains=self.request.GET.get('date')
        )

    return queryset
def get_context_data(self, *, object_list=None, **kwargs):
    context = super(
        CustomerLedgerListView, self).get_context_data(**kwargs)

    try:
        customer = Customer.objects.get(id=self.kwargs.get('pk'))
    except Customer.DoesNotExist:
        raise Http404('Customer does not exits!')

    context.update({
        'customer': customer
    })
```

```
        return context
class DebitCustomerLedgerFormView(FormView):
    template_name = 'customer_ledger/debit.html'
    form_class = CustomerLedgerForm

    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))

        return super(
            DebitCustomerLedgerFormView, self).dispatch(request, *args, **kwargs)

    def form_valid(self, form):
        obj = form.save()
        return HttpResponseRedirect(
            reverse('customer:ledger_list',
                    kwargs={'pk': obj.customer.id}
            )
        )

    def get_context_data(self, **kwargs):
        context = super(
            DebitCustomerLedgerFormView, self).get_context_data(**kwargs)
```

```
        try:
            customer = Customer.objects.get(id=self.kwargs.get('pk'))
        except Customer.DoesNotExist:
            raise Http404('Customer does not exits!')
        context.update({
            'customer': customer
        })
        return context
class CreditCustomerLedgerFormView(DebitCustomerLedgerFormView):
    template_name = 'customer_ledger/credit.html'
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            CreditCustomerLedgerFormView, self).dispatch(request, *args, **kwargs)
    def get_context_data(self, **kwargs):
        context = super(
            CreditCustomerLedgerFormView, self).get_context_data(**kwargs)
        try:
            customer = Customer.objects.get(id=self.kwargs.get('pk'))
        except Customer.DoesNotExist:
            raise Http404('Customer does not exits!')
        context.update({
```

```
            'customer': customer
        })
        return context
```

Product Views code

```
from product.forms import (
    ProductCategoryForm, ProductForm, StockInForm, StockOutForm
)
from product.models import (
    ProductCategory, Product, StockIn, StockOut
)
from django.views.generic import ListView, FormView, UpdateView
from django.http import HttpResponseRedirect
from django.urls import reverse
from django.core.exceptions import ObjectDoesNotExist
from django.http import Http404
class AddProductCategory(FormView):
    form_class = ProductCategoryForm
    template_name = 'product/add_category.html'
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            AddProductCategory, self).dispatch(request, *args, **kwargs)
```

```
    def form_valid(self, form):
        form.save()
        return HttpResponseRedirect(reverse('product:add'))
    def form_invalid(self, form):
        return super(AddProductCategory, self).form_invalid(form)
class AddProduct(FormView):
    form_class = ProductForm
    template_name = 'product/add_product.html'
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            AddProduct, self).dispatch(request, *args, **kwargs)
    def form_valid(self, form):
        form.save()
        return HttpResponseRedirect(reverse('product:list'))
    def form_invalid(self, form):
        return super(AddProduct, self).form_invalid(form)
    def get_context_data(self, **kwargs):
        context = super(AddProduct, self).get_context_data(**kwargs)
        category = ProductCategory.objects.all()
        context.update({
            'category': category
```

```
        })
        return context
class UpdateProduct(UpdateView):
    model = Product
    form_class = ProductForm
    template_name = 'product/update_product.html'
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            UpdateProduct, self).dispatch(request, *args, **kwargs)
    def form_valid(self, form):
        form.save()
        return HttpResponseRedirect(reverse('product:list'))
    def form_invalid(self, form):
        return super(UpdateProduct, self).form_invalid(form)
    def get_context_data(self, **kwargs):
        context = super(UpdateProduct, self).get_context_data(**kwargs)
        categories = ProductCategory.objects.all()
        context.update({
            'categories': categories
        })
        return context
```

```
class ProductList(ListView):
    model = Product
    template_name = 'product/product_list.html'
    paginate_by = 100
    ordering = '-id'
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            ProductList, self).dispatch(request, *args, **kwargs)
    def get_queryset(self):
        queryset = self.queryset
        if not queryset:
            queryset = Product.objects.all().order_by('-id')
        if self.request.GET.get('product_name'):
            queryset = queryset.filter(
                name__contains=self.request.GET.get('product_name')
            )
        if self.request.GET.get('product_category'):
            queryset = queryset.filter(
                category__category__contains=self.request.GET.get('product_category')
```

```
            )


        return queryset.order_by('-id')
class StockInProduct(FormView):
    form_class = StockInForm
    template_name = 'product/add_stock_item.html'
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            StockInProduct, self).dispatch(request, *args, **kwargs)
    def form_valid(self, form):
        obj = form.save()
        return HttpResponseRedirect(reverse('product:stockin_detail',
                                    kwargs={'pk': obj.product.id}))
    def form_invalid(self, form):
        return super(StockInProduct, self).form_invalid(form)


    def get_context_data(self, **kwargs):
        context = super(StockInProduct, self).get_context_data(**kwargs)
        try:
            product = (
                Product.objects.get(id=self.kwargs.get('pk'))
```

```
            )
        except ObjectDoesNotExist:
            raise Http404('Product not found')
        context.update({
            'product': product
        })
        return context
class StockOutProduct(FormView):
    form_class = StockOutForm
    template_name = 'product/stock_out_item.html'
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            StockOutProduct, self).dispatch(request, *args, **kwargs)
    def form_valid(self, form):
        obj = form.save()
        return HttpResponseRedirect(reverse('product:stockout_detail',
                                            kwargs={'pk': obj.product.id}))
    def form_invalid(self, form):
        return super(StockOutProduct, self).form_invalid(form)
    def get_context_data(self, **kwargs):
        context = super(StockOutProduct, self).get_context_data(**kwargs)
```

```
        try:
            product = (
                Product.objects.get(id=self.kwargs.get('pk'))
            )
        except ObjectDoesNotExist:
            raise Http404('Product not found')
        context.update({
            'product': product
        })
        return context
class StockInDetail(ListView):
    template_name = 'product/stockin_detail.html'
    paginate_by = 100
    model = StockIn
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
            return HttpResponseRedirect(reverse('common:login'))
        return super(
            StockInDetail, self).dispatch(request, *args, **kwargs)
    def get_queryset(self, **kwargs):
        queryset = self.queryset
        if not queryset:
            queryset = self.model.objects.filter(
```

```
            product__id=self.kwargs.get('pk')).order_by('-date')
        if self.request.GET.get('date'):
            queryset = queryset.filter(
                dated_order__icontains=self.request.GET.get('date')
            )
        return queryset.order_by('-id')
    def get_context_data(self, *, object_list=None, **kwargs):
        context = super(
            StockInDetail, self).get_context_data(**kwargs)
        try:
            product = Product.objects.get(id=self.kwargs.get('pk'))
        except Product.DoesNotExist:
            raise Http404('Product does not exits!')
        context.update({
            'product': product
        })
        return context
class StockOutDetail(ListView):
    template_name = 'product/stockout_detail.html'
    paginate_by = 100
    model = StockOut
    def dispatch(self, request, *args, **kwargs):
        if not self.request.user.is_authenticated:
```

```
            return HttpResponseRedirect(reverse('common:login'))


        return super(
            StockOutDetail, self).dispatch(request, *args, **kwargs)
    def get_queryset(self, **kwargs):
        queryset = self.queryset
        if not queryset:
            queryset = self.model.objects.filter(
                product__id=self.kwargs.get('pk')).order_by('-date')
        if self.request.GET.get('date'):
            queryset = queryset.filter(
                date__icontains=self.request.GET.get('date')
            )
        return queryset.order_by('-id')
    def get_context_data(self, *, object_list=None, **kwargs):
        context = super(
            StockOutDetail, self).get_context_data(**kwargs)
        try:
            product = Product.objects.get(id=self.kwargs.get('pk'))
        except Product.DoesNotExist:
            raise Http404('Product does not exits!')
        context.update({
            'product': product
```

```
    })

    return context
```

2. Front-End

Banking functions

- **Add bank**

```
{% extends 'base.html' %}

{% block bank %}active{% endblock %}

{% block add_bank %}avtive{% endblock %}

{% block content %}

{% load staticfiles %}


<div class="content-wrapper">
  <section class="content-header">
    <ol class="breadcrumb">
      <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
      <li class="active">Add Bank</li>
    </ol>
  </section>
  <br>
  <section class="content">
    <div class="row">
```

```
<div class="col-md-6 pull-left">
  <div class="box">
    <div class="box-header">
      <h3 class="text-center"><b>Add Bank</b></h3>
    </div>
    <div class="box-body">
      <form action="{% url 'bank:add' %}" method="post" autocomplete="off">
        {% csrf_token %}
        <div class="row">
          <div class="col-md-12 pull-left form-group">
            <label class="form-control-label">Bank Name <span style="color:red">*</span></label>
            <input type="text" name="name" class="form-control form-control-alternative" placeholder="Bank Name" required>
          </div>
          <div class="col-md-12 pull-left form-group">
            <label class="form-control-label">Bank Branch <span style="color:red">*</span></label>
            <input type="text" name="branch" class="form-control form-control-alternative" placeholder="Bank Branch" required>
          </div>
        </div>
        <div class="row">
          <div class="col-md-12 pull-left form-group">
```

```
                        <label class="form-control-label">Account Number <span style="color:red">*</span></label>
                        <input type="text" name="account_number" class="form-control form-control-alternative" placeholder="Account Number" required>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-4 pull-right form-group">
                        <label class="form-control-label"></label>
                        <button type="submit" value="Create" class="btn btn-primary form-control">Save</button>
                    </div>
                </div>
            </form>
        </div>
    </div>
  </div>
 </section>
</div>

{% endblock %}
```

```
{% extends 'base.html' %}

{% block bank %}active{% endblock %}

{% block bank_list %}active{% endblock %}

{% block content %}

{% load staticfiles %}


<div class="content-wrapper">
   <section class="content-header">
      <ol class="breadcrumb">
         <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
         <li><a href="{% url 'bank:detail_list' pk=bank.id %}">Bank Detail</a></li>
         <li class="active">Add Credit</li>
      </ol>
   </section>
   <br>
   <section class="content">
      <div class="row">
         <div class="col-md-6 pull-left">
            <div class="box">
               <div class="box-header">
```

<h3 class="text-center">{{ bank.name|title }} | <small>Add Credit Amount</small></h3>

<form action="{% url 'bank:debit' pk=bank.id %}" method="post" autocomplete="off">

{% csrf_token %}

<input type="hidden" name="bank" value="{{bank.id}}">

<label class="form-control-label">Credit Amount *</label>

<input type="text" name="credit" class="form-control form-control-alternative" placeholder="Credit Amount" required>

<label class="form-control-label">Date *</label>

<input type="date" name="date" class="form-control" required>

```
                        <label class="form-control-label">Detail <span style="color:red">*</span></label>
                        <input type="text" name="description" class="form-control form-control-alternative" placeholder="Detail" required>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-4 pull-right form-group">
                        <label class="form-control-label"></label>
                        <button type="submit" value="Create" class="btn btn-primary form-control">Save</button>
                    </div>
                </div>
            </form>
        </div>
    </div>
  </div>
 </section>
</div>

{% endblock %}
```

Customer Functions

- **Add customer**

```
{% extends 'base.html' %}

{% block customer %}active{% endblock %}

{% block customer_create %}active{% endblock %}

{% block content %}

{% load staticfiles %}


<div class="content-wrapper">
   <section class="content-header">
      <ol class="breadcrumb">
         <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
         <li class="active">Add Customer</li>
      </ol>
   </section>
   <br>
   <section class="content">
<div class="row">
```

```
<div class="col-md-12">
  <div class="box">
    <div class="box-header">
      <h4 class="text-center"><b>Add Customer</b></h4>
    </div>
    <div class="box-body">
      <form action="{% url 'customer:add' %}" method="post" autocomplete="off">
        {% csrf_token %}
        <div class="row">
          <div class="col-md-4 pull-left form-group">
            <label class="form-control-label">Name <span style="color:red">*</span></label>
            <input type="text" name="name" class="form-control form-control-alternative" placeholder="Name" required>
          </div>
          <div class="col-md-4 form-group">
            <label class="form-control-label">Father Name <span style="color:red">*</span></label>
            <input type="text" name="father_name" class="form-control form-control-alternative" placeholder="Father Name" required>
          </div>
          <div class="col-md-4 pull-right form-group">
```

<label class="form-control-label">Pin *</label>

<input type="text" name="cnic" class="form-control form-control-alternative" placeholder="Pin" required>

<label class="form-control-label">City *</label>

<input type="text" name="city" class="form-control form-control-alternative" placeholder="City" required>

<label class="form-control-label">Mobile *</label>

<input type="text" name="mobile" class="form-control form-control-alternative" placeholder="Mobile" required>

<label class="form-control-label">Alternate Mobile *</label>

<input type="text" name="alternate_mobile" class="form-control form-control-alternative" placeholder="Alternate Mobile">

```
                </div>
                <div class="row">
                  <div class="col-md-6 pull-left form-group">
                    <label class="form-control-label">Resident <span style="color:red">*</span></label>
                    <input type="text" name="resident" class="form-control form-control-alternative" placeholder="Resident" required>
                  </div>
                  <div class="col-md-6 pull-right form-group">
                    <label class="form-control-label">Complete Address <span style="color:red">*</span></label>
                    <input type="text" name="address" class="form-control form-control-alternative" placeholder="Complete Address" required>
                  </div>
                </div>
                <div class="row">
                  <div class="col-md-2 pull-left form-group">
                    <label class="form-control-label"></label>
                    <button type="submit" value="Create" class="btn btn-primary form-control">Save</button>
                  </div>
                </div>
              </form>
            </div>
```

```
            </div>
          </div>
        </div>
    </section>
</div>


{% endblock %}
```

- **Update customer**

```
{% extends 'base.html' %}
{% block customer %}active{% endblock %}
{% block customer_list %}active{% endblock %}
{% block content %}
{% load staticfiles %}

<div class="content-wrapper">
    <section class="content-header">

        <ol class="breadcrumb">
            <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
            <li><a href="{% url 'customer:list' %}">Customer List</a></li>
```

```
        <li class="active">Update</li>
      </ol>
    </section>
    <br>
    <section class="content">
      <div class="row">
        <div class="col-md-12">
          <div class="box">
            <div class="box-header">
              <h3 class="text-center"><b>Update Customer Detail</b></h3>
            </div>
            <div class="box-body">
              <form action="{% url 'customer:update' pk=object.id %}" method="post" autocomplete="off">
                {% csrf_token %}
                <div class="row">
                  <div class="col-md-4 pull-left form-group">
                    <label class="form-control-label">Name <span style="color:red">*</span></label>
                    <input type="text" name="name" class="form-control form-control-alternative" placeholder="Name" value="{{ object.name }}" required>
                  </div>
                  <div class="col-md-4 form-group">
```

<label class="form-control-label">Father name *</label>

<input type="text" name="father_name" class="form-control form-control-alternative" placeholder="father_name" value="{{ object.father_name }}" required>

<label class="form-control-label">Pin *</label>

<input type="text" name="cnic" class="form-control form-control-alternative" placeholder="Pin" value="{{ object.cnic }}" required>

<label class="form-control-label">City *</label>

<input type="text" name="city" class="form-control form-control-alternative" placeholder="City" value="{{ object.city }}" required>

<label class="form-control-label">Mobile *</label>

<input type="text" name="mobile" class="form-control form-control-alternative" placeholder="Mobile" value="{{ object.mobile }}" required>

```
                    </div>
                    <div class="col-md-4 pull-right form-group">
                      <label class="form-control-label">Alternate Mobile <span style="color:red">*</span></label>
                      <input type="text" name="alternate_mobile" class="form-control form-control-alternative" placeholder="Alternate Mobile" value="{{ object.alternate_mobile }}">
                    </div>
                  </div>
                  <div class="row">
                    <div class="col-md-6 pull-left form-group">
                      <label class="form-control-label">Resident <span style="color:red">*</span></label>
                      <input type="text" name="resident" class="form-control form-control-alternative" placeholder="Resident" value="{{ object.resident }}" required>
                    </div>
                    <div class="col-md-6 pull-right form-group">
                      <label class="form-control-label">Complete Address <span style="color:red">*</span></label>
                      <input type="text" name="address" class="form-control form-control-alternative" placeholder="Complete Address" value="{{ object.address }}" required>
                    </div>
                  </div>
                  <div class="row">
```

```
                        <div class="col-md-2 pull-left form-group">
                            <label class="form-control-label"></label>
                            <button type="submit" value="Create" class="btn btn-primary form-control">Save</button>
                        </div>
                    </div>
                </form>
            </div>
        </div>
    </div>
  </section>
</div>

{% endblock %}
```

- **Expense (Add Expense)**

```
{% extends 'base.html' %}
{% block expense %}active{% endblock %}
{% block add_expense %}active{% endblock %}
{% block content %}
{% load staticfiles %}
```

```
<div class="content-wrapper">
  <section class="content-header">
    <ol class="breadcrumb">
      <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
      <li class="active">Add Expense</li>
    </ol>
  </section>
  <br>
  <section class="content">
    <div class="row">
      <div class="col-md-6 pull-left">
        <div class="box">
          <div class="box-header">
            <h3 class="text-center"><b>Add Expense</b></h3>
          </div>
          <div class="box-body">
            <form action="{% url 'expense:add' %}" method="post" autocomplete="off">
              {% csrf_token %}
              <div class="row">
                <div class="col-md-12 pull-left form-group">
```

<label class="form-control-label">Amount *</label>

<input type="text" name="amount" class="form-control form-control-alternative" placeholder="Amount" required>

<label class="form-control-label">Date *</label>

<input type="date" name="date" class="form-control form-control-alternative" placeholder="Date" onfocus="(this.type='date')" onblur="(this.type='text')" required>

<label class="form-control-label">Description *</label>

<input type="text" name="description" class="form-control form-control-alternative" placeholder="Description" required>

```
                        <label class="form-control-label"></label>
                        <button type="submit" value="Create" class="btn btn-primary form-control">Save</button>
                    </div>
                </div>
            </form>
        </div>
    </div>
  </div>
 </section>
</div>

{% endblock %}
```

- **Update expense**

```
{% extends 'base.html' %}
{% block expense %}active{% endblock %}
{% block expense_list %}active{% endblock %}
{% block content %}
{% load staticfiles %}
```

```
<div class="content-wrapper">
  <section class="content-header">

    <ol class="breadcrumb">
      <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
      <li><a href="{% url 'expense:list' %}">Expense List</a></li>
      <li class="active">Update</li>
    </ol>
  </section>
  <br>
  <section class="content">
    <div class="row">
      <div class="col-md-6 pull-left">
        <div class="box">
          <div class="box-header">
            <h3 class="text-center"><b>Update Expense</b></h3>
          </div>
          <div class="box-body">
            <form action="{% url 'expense:update' pk=object.id %}" method="post" autocomplete="off">
              {% csrf_token %}
              <div class="row">
```

```
                    <div class="col-md-12 pull-left form-group">
                        <label class="form-control-label">Amount <span style="color:red">*</span></label>
                        <input type="text" name="amount" class="form-control form-control-alternative" placeholder="Amount" value="{{ object.amount|floatformat:-2 }}" required>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-12 pull-left form-group">
                        <label class="form-control-label">Date <span style="color:red">*</span></label>
                        <input type="date" name="date" class="form-control form-control-alternative" placeholder="Date" onfocus="(this.type='date')" onblur="(this.type='text')" required>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-12 pull-left form-group">
                        <label class="form-control-label">Description <span style="color:red">*</span></label>
                        <input type="text" name="description" class="form-control form-control-alternative" placeholder="Description" value="{{ object.description }}" required>
                    </div>
```

```
                    </div>
                    <div class="row">
                        <div class="col-md-4 pull-right form-group">
                            <label class="form-control-label"></label>
                            <button type="submit" value="Create" class="btn btn-primary form-control">Save</button>
                        </div>
                    </div>
                </form>
            </div>
        </div>
    </div>
</section>
</div>


{% endblock %}
```

- **Logs (monthly)**

```
{% extends 'base.html' %}
{% block stock_logs %}active{% endblock %}
{% block monthly_stock_logs %}active{% endblock %}
{% block content %}
{% load staticfiles %}
  <div class="content-wrapper">
    <section class="content-header">
      <ol class="breadcrumb">
        <li><a href="#"><i class="fa fa-dashboard"></i> Home</a></li>
        <li class="active">Monthly Logs</li>
      </ol>
    </section>
    <br>
    <section class="content">
      <div class="box">
        <h2 class="text-center">Monthly Stock Logs | <small>{{ month }} {{ year }}</small></h2>
        <hr>
        <div class="row">
          <div class="col-lg-12">
            <div class="custom-search-form col-md-4">
```

```
                    <label>Search</label>
                    <input type="text" class="form-control search-logs" placeholder="Search...">
                </div>
                <div class="col-md-4">
                    <label>Filter By Month</label>
                    <select class="form-control month" id="month">
                        <option value="January">January</option>
                        <option value="February">February</option>
                        <option value="March">March</option>
                        <option value="April">April</option>
                        <option value="May">May</option>
                        <option value="June">June</option>
                        <option value="July">July</option>
                        <option value="August">August</option>
                        <option value="September">September</option>
                        <option value="October">October</option>
                        <option value="November">November</option>
                        <option value="December">December</option>
                    </select>
                </div>
                <div class="col-md-4" style="padding-top: 30px">
```

Logs Date: {{ month }} {{ year }} | Total Price: OMR. {{ total|floatformat:'-2' }}

<table class="table table-striped table-advance table-hover item-table table-responsive table-bordered text-center">

<thead>

<tr>

<th class="text-center">Item Name</th>

<th class="text-center">No. of Invoices</th>

<th class="text-center">Stock Out</th>

</tr>

</thead>

<tbody class="logs-table-body">

{% if not object_list %}

<tr><td colspan="3" class="text-center">No Logs Found</td></tr>

{% endif %}

{% for stock in object_list %}

<tr>

<td>{{ stock.product__name|title}}</td>

<td>{{ stock.receipt_item|floatformat:-2}}</td>

```
                    <td>{{ stock.total_qty|floatformat:-2 }} </td>
                </tr>
                {% endfor %}
                </tbody>
            </table>
        </div>
        {% if paginator.page_range|length > 1 %}
            <div class="text-center">
                <ul class="pagination">
                    <li><a style="background-color: #3070A8; color: white" class="page_previous" href="{% if page_obj.has_previous %}{% url 'common:monthly_logs' %}?page={{ page_obj.previous_page_number }}{% else %}#{% endif %}" title="Previous"><<<</a></li>
                    <li><a style="background-color: #3070A8; color: white" class="page_next" href="{% if page_obj.has_next %}{% url 'common:monthly_logs' %}?page={{ page_obj.next_page_number }}{% else %}#{% endif %}" title="Next">>>></a></li>
                </ul>
            </div>
        {% endif %}
    </div>
</section>
</div>
{% endblock %}
```

```
{% block script %}
   {{ block.super }}
   <script>
      $(document).ready(function(){

         $('#month option:contains("{{ month }}")').prop('selected',true);

         $(".search-logs").on("keyup", function() {
            var value = $(this).val().toLowerCase();
            $(".logs-table-body tr").filter(function() {
               $(this).toggle($(this).text().toLowerCase().indexOf(value) > -1)
            });
         });

         $('.month').on('change', function () {
            if ($(this).val() == '') {
               window.location.href = '{% url "common:monthly_logs" %}'
            } else {
               window.location.href = '{% url "common:monthly_logs" %}' + '?month=' +
$(this).val();
            }
         })
```

```
        });
    </script>
    <script>
        $('#stock-log').on('click', function() {
            var $this = $(this);
            $this.button('loading');
            setTimeout(function() {
                $this.button('reset');
            }, 5000);
        });
    </script>
{% endblock %}
```

- **Daily Logs**

```
{% extends 'base.html' %}
{% block stock_logs %}active{% endblock %}
{% block daily_stock_logs %}active{% endblock %}
{% block content %}
{% load staticfiles %}

<div class="content-wrapper">
    <section class="content-header">
```

```
        <ol class="breadcrumb">
            <li><a href="#"><i class="fa fa-dashboard"></i> Home</a></li>
            <li class="active">Daily Logs</li>
        </ol>
    </section>
    <br>
    <section class="content">
        <div class="box">
            <h2 class="text-center">Daily Stock Logs</h2>
            <hr>
            <div class="row">
                <div class="col-lg-12">
                    <div class="custom-search-form col-md-4">
                        <label>Search</label>
                        <input type="text" class="form-control search-logs" placeholder="Search...">
                    </div>
                    <div class="col-md-4">
                        <label>Filter By Date</label>
                        <input type="date" class="form-control date" placeholder="Filter by Date" value="{% if today_date %}{{ today_date }}{% else %}{{ logs_date }}{% endif %}">
                    </div>
```

Logs Date: {% if today_date %}{{ today_date }}{% endif %}{% if logs_date %}{{ logs_date }}{% endif %} | Total Price: OMR. {{ total|floatformat:'-2' }}

<table class="table table-striped table-advance table-hover item-table table-responsive table-bordered text-center">

<thead>

<tr>

<th class="text-center">Item Name</th>

<th class="text-center">No. of Invoices</th>

<th class="text-center">Stock Out</th>

</tr>

</thead>

<tbody class="logs-table-body">

{% if not object_list %}

<tr><td colspan="3" class="text-center">No Logs Found</td></tr>

{% endif %}

{% for stock in object_list %}

<tr>

```
                <td>{{ stock.product__name|title}}</td>
                <td>{{ stock.receipt_item|floatformat:-2}}</td>
                <td>{{ stock.total_qty|floatformat:-2 }} </td>
            </tr>
            {% endfor %}
            </tbody>
        </table>
        </div>
        {% if paginator.page_range|length > 1 %}
        <div class="text-center">
            <ul class="pagination">
                <li><a style="background-color: #3070A8; color: white" class="page_previous" href="{% if page_obj.has_previous %}{% url 'common:daily_logs' %}?page={{ page_obj.previous_page_number }}{% else %}#{% endif %}" title="Previous"><<<</a></li>
                <li><a style="background-color: #3070A8; color: white" class="page_next" href="{% if page_obj.has_next %}{% url 'common:daily_logs' %}?page={{ page_obj.next_page_number }}{% else %}#{% endif %}" title="Next">>>></a></li>
            </ul>
        </div>
        {% endif %}
    </div>
</section>
</div>
```

```
{% endblock %}


{% block script %}
   {{ block.super }}
<script>
       $(document).ready(function(){
          $(".search-logs").on("keyup", function() {
             var value = $(this).val().toLowerCase();
             $(".logs-table-body tr").filter(function() {
                $(this).toggle($(this).text().toLowerCase().indexOf(value) > -1)
             });
          });


          $('.date').on('change', function () {
             if ($(this).val() == '') {
                window.location.href = '{% url "common:daily_logs" %}'
             } else {
                window.location.href = '{% url "common:daily_logs" %}' + '?date=' +
$(this).val();
             }
          })


       });
```

```
    </script>
<script>
$('#stock-log').on('click', function() {
    var $this = $(this);
  $this.button('loading');
    setTimeout(function() {
       $this.button('reset');
   }, 5000);
});
    </script>

{% endblock %}
```

- **Products (Add category)**

```
{% extends 'base.html' %}
{% block product %}active{% endblock %}
{% block add_category %}active{% endblock %}
{% block content %}
{% load staticfiles %}

<div class="content-wrapper">
    <section class="content-header">
```

```
      <ol class="breadcrumb">
        <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
        <li class="active">Add Category</li>
      </ol>
    </section>
    <br>
    <section class="content">
      <div class="row">
        <div class="col-md-6 pull-left">
          <div class="box">
            <div class="box-header">
              <h4 class="text-center"><b>Add Category</b></h4>
            </div>
            <div class="box-body">
              <form action="{% url 'product:add_category' %}" method="post" autocomplete="off">
                {% csrf_token %}
                <div class="row">
                  <div class="col-md-12 pull-left form-group">
                    <label class="form-control-label">Category <span style="color:red">*</span></label>
                    <input type="text" name="category" class="form-control form-control-alternative" placeholder="Add Category" required>
```

```
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-4 pull-right form-group">
                        <label class="form-control-label"></label>
                        <button type="submit" value="Create" class="btn btn-primary form-control">Save</button>
                    </div>
                </div>
            </form>
        </div>
    </div>
</div>
</section>
</div>

{% endblock %}
```

- **Add product**

```
{% extends 'base.html' %}
```

```
{% block product %}active{% endblock %}

{% block add_product %}active{% endblock %}

{% block content %}

{% load staticfiles %}


<div class="content-wrapper" xmlns:>
   <section class="content-header">
      <ol class="breadcrumb">
         <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
         <li><a href="{% url 'product:add_category' %}">Product Category</a></li>
         <li class="active">Add Product</li>
      </ol>
   </section>
   <br>
   <section class="content">
      <!-- Info boxes -->
      <div class="row">
         <div class="col-md-6 pull-left">
            <div class="box">
               <div class="box-header">
                  <h4 class="text-center"><b>Add Product</b></h4>
               </div>
```

<form action="{% url 'product:add' %}" method="post" autocomplete="off">

{% csrf_token %}

<label class="form-control-label">Name *</label>

<input type="text" name="name" class="form-control form-control-alternative" placeholder="Name" required>

<label class="form-control-label" name="category">Category*</label>

<select class="form-control form-control-alternative" name="category">

{% for category in category %}

<option value="{{category.id}}">{{category.category}}</option>

{% endfor %}

</select>

<label class="form-control-label">Notify Quantity *</label>

<input type="number" name="notify_qty" class="form-control form-control-alternative" placeholder="Notify Quantity" required>

<label class="form-control-label">Date *</label>

<input type="date" name="date" class="form-control form-control-alternative" placeholder="Date" onfocus="(this.type='date')" onblur="(this.type='text')" required>

<label class="form-control-label"></label>

<button type="submit" value="Create" class="btn btn-primary form-control">Save</button>

</form>

</section>

```
</div>


{% endblock %}
```

- **Add stock item**

```
{% extends 'base.html' %}

{% block product %}active{% endblock %}

{% block product_list %}active{% endblock %}

{% block content %}

{% load staticfiles %}


<div class="content-wrapper">

  <section class="content-header">

    <ol class="breadcrumb">

      <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>

      <li><a href="{% url 'product:list' %}">Product List</a></li>

      <li><a href="{% url 'product:stockin_detail' pk=product.id %}">Stock In List</a></li>

      <li class="active">Add</li>

    </ol>

  </section>
```

```
<br>
<section class="content">
  <div class="row">
    <div class="col-md-12">
      <div class="box">
        <div class="box-header">
          <h3 class="text-center"><b>Add Stock Item</b></h3>
        </div>
        <div class="box-body">
          <form class="form-horizontal product_item_form" action="{% url 'product:add_stock' pk=product.id %}" method="post" autocomplete="off">
            {% csrf_token %}
            <input type="hidden" name="purchased_item" value="0">
            <input type="hidden" name="product" value="{{ product.id }}">
            <input type="hidden" name="retail_price" value="0">
            <div class="row">
              <div class="col-md-6 pull-left">
                <label class="form-control-label" >Name & Category<span style="color:red">*</span></label>
                <input type="text" class="form-control form-control-alternative" value="{{product.name}} | {{product.category.category}}" readonly>
              </div>
              <div class="col-md-6 pull-right">
```

<input type="hidden" name="product" class="form-control form-control-alternative" value="{{product.id}}" >

<label class="form-control-label">Stock Quantity*</label>

<input type="text" class="form-control quantity" name="stock_quantity" id="quantity" placeholder="Stock Quantity" required>

<label class="form-control-label">Buying Price / Item*</label>

<input type="text" class="form-control buying_price_item" name="buying_price_item" id="buying_price_item" placeholder="Buying Price Per Item" required>

<label class="form-control-label">Selling Price / Item*</label>

<input type="text" class="form-control price_per_item" name="price_per_item" id="price_per_item" onkeyup="sum()" placeholder="Price Per Item" required>

<label class="form-control-label">Buying Percent*</label>

<input type="text" class="form-control buying_percent" name="buying_percent" id="buying_percent" value="0" required>

<label class="form-control-label">Selling Price / Item*</label>

<input type="text" class="form-control selling_percent" name="selling_percent" id="selling_percent" value="0" required>

<label>Total Buying Amount</label>

<input type="text" class="form-control total_buying_amount" name="total_buying_amount" id="total_buying_amount" placeholder="Total Buying Amount" readonly>

<label>Total Selling Amount</label>

<input type="text" class="form-control total_amount" name="total_amount" id="total_amount" placeholder="Total Selling Amount" readonly>

```
                    </div>
                    <div class="row">
                        <div class="col-md-6 pull-left">
                            <label><strong>Date Stock In</strong></label>
                            <input type="text" class="form-control dated_order" name="dated_order" onkeyup="checkDec(this);" placeholder="Date Stock In" onfocus="(this.type='date')" onblur="(this.type='text')"><br>
                        </div>
                    </div>
                    <div class="row">
                        <div class="col-md-2 pull-left">
                            <button type="button"  class="btn btn-primary product-sub-btn" style="width: 100%">Save Stock</button>
                        </div>
                    </div>
                </form>
            </div>
        </div>
    </div>
</section>
</div>
```

```
{% endblock %}


{% block script %}
   {{ block.super }}
 <script>
       $(document).ready(function(){


          $('.buying_price_item').on('change', function(){
             $('.total_buying_amount').val(Number($(this).val())                    *
Number($('.quantity').val()));
          });


          $('.quantity').on('change', function(){
             if ($('.buying_price_item').val()) {
                $('.total_buying_amount').val(Number($('.buying_price_item').val())          *
Number($(this).val()));
             }
             if ($('.price_per_item').val()) {
                $('.total_amount').val(Number($('.price_per_item').val())                    *
Number($(this).val()));
             }
          });
```

```
$(".product-sub-btn").on ('click', function(){
    $(this).addClass('disabled');
    $(this).html('<i class="fa fa-circle-o-notch fa-spin"></i> Loading...');
    var error= false;

    if ($('.product-error').val() == ""){
        $('.product-error').show();
        error = true;
    }
    else {
        $('.product-error').hide();
    }
    if ($('.quantity').val () == ""){
        $('.item-error').show();
        error = true;
    }
    else {
        $('.item-error').hide();
    }
    if (error == true) {
        $(this).removeClass('disabled');
        $(this).html('Save Customer');
        return;
```

```
        }
        else
        {
            $('.product_item_form').submit();

        }

    });



});

$('.product-error').keyup (function(){
    $('.product-error').hide();
});

function checkDec(el){
    var ex = /^[0-9]+\.?[0-9]*$/;
    if(ex.test(el.value)==false){
        el.value = el.value.substring(0,el.value.length - 1);
    }
}
```

```
$('.quantity').keyup (function(){
    $('.item-error').hide();
});

function checkDec(el){
    var ex = /^[0-9]+\.?[0-9]*$/;
    if(ex.test(el.value)==false){
        el.value = el.value.substring(0,el.value.length - 1);
    }
}

$('.form-control').click (function () {
    $('.list-error').hide();
    checkDec(this);

});
function sum() {
    var txtFirstNumberValue = document.getElementById('quantity').value;
    var txtSecondNumberValue = document.getElementById('price_per_item').value;
    var result = parseInt(txtFirstNumberValue) * parseInt(txtSecondNumberValue);
    if (!isNaN(result)) {
```

```
            document.getElementById('total_amount').value = result;
        }
    }
</script>

{% endblock %}
```

- **Stock in detail**

```
{% extends 'base.html' %}
{% block product %}active{% endblock %}
{% block product_list %}active{% endblock %}
{% block content %}
{% load staticfiles %}

<div class="content-wrapper">
    <section class="content-header">
        <a class="col-md-2 pull-left" href="{%url 'product:add_stock' pk=product.id %}"><button type="button" class="btn btn-primary btn-sm">Add Stock In</button></a>
        <ol class="breadcrumb">
            <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
            <li><a href="{% url 'product:list' %}">Product List</a></li>
```

```
        <li class="active">Stock In</li>
      </ol>
    </section>
    <br>
    <section class="content">
      <div class="row">
        <div class="col-md-12">
          <div class="box">
            <div class="box-header">
              <h4 class="text-center"><b>{{ product|title }} |</b> <small><b>{{ product.category.category }}</b></small> Stock In</h4>
            </div>
            <br>
            <div class="row">
              <div class="col-md-12">
                <div class="col-md-4 pull-left">
                  <input type="date" class="form-control date_search" placeholder="Search by Date ..." >
                </div>
                <div class="col-md-2 pull-right">
                  <button class="form-control btn btn-primary search">Search</button>
                </div>
```

```
                <br><br>
                <hr>
              </div>
            </div>
            <div class="box-body">
              <div class="table-responsive">
                <table class="table table-striped table-advance table-hover item-table table-bordered">
                  <thead class="thead-dark">
                  <tr>
                    <th class="text-center">Product</th>
                    <th class="text-center"><small>Stock Quantity</small></th>
                    <th class="text-center"><small>Price Per Item</small></th>
                    <th class="text-center"><small>Selling %</small></th>
                    <th class="text-center"><small>Total Amount</small></th>
                    <th class="text-center"><small>Buying Price Item</small></th>
                    <th class="text-center"><small>Buying %</small></th>
                    <th class="text-center"><small>Total Buying Amount</small></th>
                    <th class="text-center"><small>Order Dated</small></th>
                  </tr>
                  </thead>
                  <tbody class="product-table-body text-center">
```

```
                    {% for stock_in in object_list %}
                    <tr>
                        {% load humanize %}
                        <td>{{ stock_in.product }}</td>
                        <td>{{ stock_in.stock_quantity|floatformat:-2|intcomma}}</td>
                        <td>{{ stock_in.price_per_item|floatformat:-2|intcomma }}</td>
                        <td>{{ stock_in.selling_percent|floatformat:-2|intcomma }}</td>
                        <td>{{ stock_in.total_amount|floatformat:-2|intcomma }}</td>
                        <td>{{ stock_in.buying_price_item|floatformat:-2|intcomma }}</td>
                        <td>{{ stock_in.buying_percent|floatformat:-2|intcomma }}</td>
                        <td>{{ stock_in.total_buying_amount|floatformat:-2|intcomma }}</td>
                        <td>{{ stock_in.dated_order }}</td>
                    </tr>
                    {% endfor %}
                </tbody>
            </table>
            {% if paginator.page_range|length > 1 %}
            <div class="text-center col-lg-12">
                <ul class="pagination justify-content-center">
```

<<<

>>>

{% endif %}

</section>

{% endblock %}

{% block script %}

{{ block.super }}

<script>

$(function () {

```
    $('.search').on('click', function () {
        var url = '{% url "product:stockin_detail" pk=product.id %}';
        window.location.href  = url + '?' + 'date=' + $('.date_search').val();
    });
})
</script>
{% endblock %}
```

- **Stock out details**

```
{% extends 'base.html' %}
{% block product %}active{% endblock %}
{% block product_list %}active{% endblock %}
{% block content %}
{% load staticfiles %}

<div class="content-wrapper">
```

```
<section class="content-header">
    <a class="col-md-2 pull-left" href="{%url 'product:stock_out' pk=product.id %}"><button type="button" id="add-stock" class="btn btn-primary btn-sm">Stock Out </button></a>
    <ol class="breadcrumb">
      <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
      <li><a href="{% url 'product:list' %}">Product List</a></li>
      <li class="active">Stock Out</li>
    </ol>
  </section>
  <br>
  <section class="content">
    <div class="row">
      <div class="col-md-12">
        <div class="box">
          <div class="box-header">
            <h4 class="text-center"><b>{{ product|title }} |</b> <small><b>{{ product.category.category }}</b></small> Stock Out</h4>
          </div>
          <br>
          <div class="row">
            <div class="col-md-12">
              <div class="col-md-4 pull-left">
```

<input type="date" class="form-control date_search" placeholder="Search by Date ..." >

<button class="form-control btn btn-primary search">Search</button>

<hr>

<table class="table table-striped table-advance table-hover item-table table-bordered">

<thead class="thead-dark">

<tr>

<th class="text-center">Product</th>

<th class="text-center">Stock Out Quantity</th>

<th class="text-center">Dated</th>

</tr>

</thead>

<tbody class="product-table-body text-center">

```
                        {% for stock_out in object_list %}
                        <tr>
                            {% load humanize %}
                            <td>{{ stock_out.product }} </td>
                            <td>{{ stock_out.stock_out_quantity|floatformat:-2|intcomma}}</td>
                            <td>{{ stock_out.date }}</td>
                        </tr>
                        {% endfor %}
                    </tbody>
                </table>
                {% if paginator.page_range|length > 1 %}
                <div class="text-center col-lg-12">
                    <ul class="pagination justify-content-center">
                        <li><a style="background-color: #3070A8; color: white" class="page_previous" href="{% if page_obj.has_previous %}{% url 'product:stockout_detail' pk=product.id %}?page={{ page_obj.previous_page_number }}{% else %}#{% endif %}" title="Previous"><<<</a></li>
                        <li><a style="background-color: #3070A8; color: white" class="page_next" href="{% if page_obj.has_next %}{% url 'product:stockout_detail' pk=product.id %}?page={{ page_obj.next_page_number }}{% else %}#{% endif %}" title="Next">>>></a></li>
                    </ul>
                </div>
                {% endif %}
```

```
                    </div>
                </div>
            </div>
        </div>
    </section>
</div>
{% endblock %}


{% block script %}
    {{ block.super }}
<script>
$(function () {
    $('.search').on('click', function () {
        var url = '{% url "product:stockout_detail" pk=product.id %}';
        window.location.href  = url + '?' + 'date=' + $('.date_search').val();
    });
})
</script>
{% endblock %}
```

- **Update products**

```
{% extends 'base.html' %}

{% block product %}active{% endblock %}

{% block product_list %}active{% endblock %}

{% block content %}

{% load staticfiles %}


<div class="content-wrapper">
   <section class="content-header">
      <ol class="breadcrumb">
         <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
         <li><a href="{% url 'product:list' %}">Product List</a></li>
         <li class="active">Update</li>
      </ol>
   </section>
   <br>
   <section class="content">
      <div class="row">
         <div class="col-md-6 pull-left">
            <div class="box">
               <div class="box-header">
                  <h4 class="text-center"><b>Product Update</b></h4>
               </div>
```

<form action="{% url 'product:update' pk=object.id %}" method="post" autocomplete="off">

{% csrf_token %}

<label class="form-control-label">Name *</label>

<input type="text" name="name" class="form-control form-control-alternative" placeholder="Name" value="{{ object.name }}" required>

<label class="form-control-label" name="category">Category*</label>

<select class="form-control form-control-alternative" name="category">

<option value="{{object.category.id}}">{{object.category.category}}</option>

{% for category in categories %}

<option value="{{category.id}}">{{category.category}}</option>

{% endfor %}

</select>

<label class="form-control-label">Notify Quantity *</label>

<input type="number" name="notify_qty" class="form-control form-control-alternative" placeholder="Notify Quantity" value="{{ object.notify_qty|floatformat:-2 }}" required>

<label class="form-control-label">Date *</label>

<input type="date" name="date" class="form-control form-control-alternative date" placeholder="Date" onfocus="(this.type='date')" onblur="(this.type='text')" value="{{ object.date|date:'Y-m-d' }}" required>

<label class="form-control-label"></label>

<button type="submit" value="Create" class="btn btn-primary form-control">Save</button>

</form>

```
                </div>
              </div>
            </div>
          </div>
    </section>
</div>

{% endblock %}
```

- **Reports**

```
{% extends 'base.html' %}
{% block reports %}active{% endblock %}
{% block monthly_reports %}active{% endblock %}

{% load staticfiles %}

{% block content %}
    <button type="button" class="btn btn-primary print-invoice-btn" onclick="PrintInvoice('print-invoice')" style="margin-left:10px">Print </button>
    <div class="content-wrapper">
        <section class="content-header">
            <ol class="breadcrumb">
```

```
            <li><a href="#"><i class="fa fa-dashboard"></i> Home</a></li>
            <li class="active">Monthly Reports</li>
          </ol>
        </section>
        <br>
        <section class="content">
          <div class="row">
            <div class="col-md-12">
              <div class="box">
                <div class="box-header with-border">
                  <h2 class="box-title">Monthly Sales Reports</h2>

                  <div class="box-tools pull-right">
                    <button type="button" class="btn btn-box-tool" data-widget="collapse"><i class="fa fa-minus"></i>
                    </button>
                    <button type="button" class="btn btn-box-tool" data-widget="remove"><i class="fa fa-times"></i></button>
                  </div>
                </div>

<div class="box-body">
                  <div class="row">
```

```
                        <div class="col-lg-12">
                            <p class="text-center">
                                <strong>Monthly Sales Reports</strong>
                            </p>

                            <div class="chart">
                                <canvas id="salesChart" style="height: 180px;"></canvas>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        <div class="box">
            <div class="box-body">
                <div class=" print-invoice" id="print-invoice">
                    <div class="col-md-12">
                        <br>
                        <div class="content-panel table-responsive">
                            <table class="table table-striped table-advance table-hover table-bordered item-table">
                                <thead>
                                <tr>
```

```
                <th>Date</th>
                <th>Total Customer</th>
                <th>Total Quantity</th>
                <th>Grand Total</th>
                <th>Total Cash Payment</th>
            </tr>
            </thead>
            <tbody class="product-table-body">
            {% for result in results %}
                {% if not result.grand_total == 0 %}
                    <tr>
                        <td>{{ result.date }}</td>
                        <td>{{ result.total_customer }}</td>
                        <td>{{ result.total_quantity|floatformat:-2 }}</td>
                        <td>{{ result.grand_total|floatformat:'-2' }}</td>
                        <td>{{ result.total_cash_payment|floatformat:'-2' }}</td>
                    </tr>
                {% endif %}
            {% endfor %}
            </tbody>
        </table>
    </div>
</div>
```

```
                </div>
            </div>
        </div>
    </section>
</div>
{% endblock %}


{% block script %}
    {{ block.super }}
    <script>
        var labels = [];
        var data_grand_total = [];
        var data_cash_payment = [];
        {% for result in results|slice:":12" %}
            labels.push("{{ result.date }}");
            data_grand_total.push({{ result.grand_total|floatformat:-2 }});
            data_cash_payment.push({{ result.total_cash_payment|floatformat:-2 }});
        {% endfor %}
    </script>
    <script src="{% static 'dist/js/pages/dashboard2.js' %}"></script>
{% endblock %}
```

- **Sales (Create invoice)**

```
{{% extends 'base.html' %}
{% block sales %}active{% endblock %}
{% block create_invoice %}active{% endblock %}
{% load staticfiles %}
{% block content %}
  <style>
    .item-row th {
      background: #eee;
    }

    .delete-btn {
      position: relative;
    }

    .delete {
      display: block;
      color: #000;
      text-decoration: none;
      position: absolute;
      background: #EEEEEE;
      font-weight: bold;
```

```
            padding: 0px 3px;
            border: 1px solid;
            top: -6px;
            left: -6px;
            font-family: Verdana;
            font-size: 12px;
        }

        table {
          table-layout: fixed ;
          width: 100% ;
        }
        td {
          width: 25% ;
        }

    </style>
    <div class="content-wrapper">
        <section class="content-header">
            <ol class="breadcrumb">
                <li><a href="{% url 'common:index' %}"><i class="fa fa-dashboard"></i> Home</a></li>
                <li class="active">Create Invoice</li>
```

```
    </ol>
</section>
<br>
<section class="content">
    <div class="row" style="font-size: large">
        <div class="col-lg-12">
            <div class="box">
                <div class="box-header">
                    <h2 class="text-center">Create Invoice</h2>
                </div>
                <hr>
                <div class="box-body">
                    <div class="row" style="font-size: large">
                        <div class="col-xs-12 col-md-12">
                            <div class="row">
                                <div class="col-xs-6 col-md-6">
                                    <div id="form-control   item-data-sc" class="table-responsive item-data-sc" style="height: 500px; overflow: auto; border: 1px solid #eee">
                                        <table class="table table-bordered">
                                            <thead>
                                            <tr class="item-row">
                                                <th>Item</th>
                                                <th>Stock</th>
```

<th>Price</th>

<th>Quantity</th>

<th>Total</th>

</tr>

</thead>

<tbody>

</tbody>

</table>

Add Item please add Item

Billed To: {{today_date|date:'d-m-y'}}

```
                                <div class="col-xs-6 col-md-6">
                                    <select class="form-control payment_type" name="payment_type" id="payment_type">
                                        <option value="Cash">Cash</option>
                                        <option value="Installment">Installment</option>
                                    </select>
                                </div>
                                <div class="bank-wrapper" style="display: none">
                                    <div class="col-xs-6 col-md-6">
                                        <br>
                                        Select a Bank for Deposit Check:
                                    </div>
                                    <div class="col-xs-6 col-md-6">
                                        <br>
                                        <select class="form-control bank" name="bank" id="bank">
                                            {% for bank in banks %}
                                                <option value="{{ bank.id }}">{{ bank.name|title }}</option>
                                            {% endfor %}
                                        </select>
                                    </div>
                                </div>
```

```
                </div>
            </div>
            <br>
            <div class="existing-customer">
                <span class="customererror" style="color: red; display: none">Customer Doesn't exist please add Customer</span>
                <input class="form-control customer" id="customer-id" placeholder="Customer" list="customer-list" >
                <datalist id="customer-list">
                    {% for customer in customers %}
                        <option data-id="{{ customer.id }}" value="{{ customer.name }}"></option>
                    {% endfor %}
                </datalist><br>
                <a class="new-customer" style="cursor: pointer;">New Customer</a><br>
                <br>
            </div>
            <div id="new-customer-form" class="new-customer-form" style="display: none">
                <span class="already-error" style="color: red; display: none">This Customer Already exist</span>
                <input type="text" class="form-control customer_name" name="customer_name" placeholder="Customer Name"><br>
```

<input type="text" class="form-control customer_cnic" name="customer_cnic" placeholder="Customer CNIC">

<input type="number" class="form-control customer_phone" name="customer_phone" placeholder="Customer Phone">

Existing Customer

<table class="table table-bordered">

<thead>

<tbody>

<tr hidden>

<td></td>

<td></td>

<td class="text-right">Sub Total</td>

<td>0.00</td>

</tr>

<tr hidden>

<td></td>

<td></td>

<td class="text-right">Discount</td>

<td><input class="form-control" id="discount" value="0" type="text"></td>

</tr>

<tr hidden>

<td></td>

<td></td>

<td class="text-right">Shipping</td>

<td><input class="form-control" id="shipping" value="0" type="text"></td>

</tr>

<tr>

<td>Total Qty: 0</td>

<td></td>

<td class="text-right">Grand Total</td>

<td>0</td>

</tr>

<tr>

<td colspan="4" class="text-center"><h4>Customer Ledger</h4></td>

</tr>

<tr>

<td colspan="3" class="text-right">Received Amount</td>

<td><input class="form-control" id="paidAmount" value="0" type="text"></td>

</tr>

<tr>

<td colspan="3" class="text-right">Remaining Amount</td>

<td>0</td>

</tr>

<tr>

<td colspan="4" class="text-center"><h4>Cash Calculator</h4></td>

</tr>

<tr>

<td class="text-right" colspan="3">Cash Payment</td>

<td><input class="form-control" id="cash_payment" value="0" type="text"></td>

</tr>

<tr>

<td class="text-right" colspan="3">Returned Cash</td>

<td>0</td>

</tr>

```
                                    </tbody>
                                </table>
                            </div>
                            <br>
                            <div class="col-md-12">
                                <a id="create-invoice" href="javascript:;" title="Create Invoice" class="form-control btn btn-primary">Create Invoice</a>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </section>
</div>
{% endblock %}

{% block script %}
    {{ block.super }}
```

```
<script src="{% static 'dist/js/jquery.invoice.js' %}"></script>
<script>
    jQuery(document).ready(function(){
        jQuery().invoice({
            addRow : "#addRow",
            delete : ".delete",
            parentClass : ".item-row",

            price : ".price",
            qty : ".qty",
            total : ".total",
            totalQty: "#totalQty",

            subtotal : "#subtotal",
            discount: "#discount",
            shipping : "#shipping",
            grandTotal : "#grandTotal",

            remainingAmount: '#remainingAmount',
            paidAmount: '#paidAmount',

            cashPayment: '#cash_payment',
            returnedCash: '#returned_cash'
```

```
        });


    });
    $(function () {
        $('body').on('focusout', '.invoice-item', function(){


            var item_price = $(this).parent().parent().find("option[value='" + $(this).val() + "']").data('price');
            $(this).parent().parent().parent().find('.price').val(item_price);


            var stock = $(this).parent().parent().find("option[value='" + $(this).val() + "']").data('stock')
            $(this).parent().parent().parent().find('.stock').html(stock);
        });
        $('body').on('click', '#create-invoice', function () {


            if ($('#item-name').hasClass('item-name') == false ) {
                $('.item-error').show();
                return;
            }
            var error = false;
            if ($('.customer').is(":visible") == true) {
```

```
 var existing_customers_id = [];
                {% for customer in customers %}
                   existing_customers_id.push({{ customer.id }});
                {% endfor %}

                var cus_val = $('.customer').val();
                var cus_id = $('#customer-list [value="' + cus_val + '"]').data('id');

}
            if ($('.customer_name').is(":visible") == true) {
                if ($('.customer_name').val() == "") {
                   $('.customer_name').css('border-color', 'red');
                   error = true;
                }
                else {
                   $('.customer_name').css('border', 'none');
                }
                if ($('.customer_cnic').val() == "") {
                   $('.customer_cnic').css('border-color', 'red');
                   error = true;
                }
                else {
                   $('.customer_cnic').css('border', 'none');
```

```
        }
        if ($('.customer_phone').val() == "") {
            $('.customer_phone').css('border-color', 'red');
        }
        else {
            $('.customer_phone').css('border', 'none');
        }
        {% for customer in customers %}
            if ($('.customer_name').val() == '{{ customer.name }}' && $('.customer_cnic').val()== '{{ customer.cnic }}') {
                $('.already-error').show();
                return;
            }
        {% endfor %}
    }
    if ($('.invoice-item').val() == ""){
        $(".item-error").show();
        error=true;
        $('.invoice-item').css('border-color', 'red');
    }
    else {
        $('.invoice-item').css('border' ,'none');
```

```
}
if ($('.price').val()=="") {

    $('.item-error').show();
    $('.price').css('border-color', 'red');
    error=true;
}
else {
    $('.item-error').hide();
    $('.price').css('border', 'none');
}
if (error == true){
    return;
}
var customer_name = $('.customer_name').val();
var customer_phone = $('.customer_phone').val();
var customer_cnic = $('.customer_cnic').val();
var payment_type = $('.payment_type').val();
var bank = $('.bank').val();
var items = [];
var total_quantity = 0;
for (var i=1; i<$('.item-row').length; i+=1) {
    var products = {};
```

```
            products['item_name'] = $('.item-row').eq(i).find('#invoice-item').val();


            var ll = $('.item-row').eq(i).find('#invoice-item').val();
            products['item_id'] = $('.item-row').eq(i).find('#all-items [value="' + ll + '"]').data('id');


            products['price'] = $('.item-row').eq(i).find('.price').val();
            products['qty'] = $('.item-row').eq(i).find('.qty').val();
            products['perdiscount'] = $('.item-row').eq(i).find('.perdiscount').val();
            products['total'] = $('.item-row').eq(i).find('.total').text();
            if ($('.item-row').eq(i).find('#invoice-item').val() != "") {
                items.push(products);
                total_quantity += Number(products['qty']);
            }
        }
        var sub_total = $('#subtotal').text();
        var discount = $('#discount').val();
        var shipping = $('#shipping').val();
        var grand_total = $('#grandTotal').text();
        var remaining_amount = $('#remainingAmount').text();
        var paid_amount = $('#paidAmount').val();
        var cash_payment = $('#cash_payment').val();
        var returned_cash = $('#returned_cash').text();
```

```
var totalQty = total_quantity;


var customer_value = $('.customer').val();
var customer_id = $('#customer-list [value="' + customer_value + '"]').data('id');


$.post("{% url 'sales:generate_invoice_api' %}", {
        customer_id: customer_id,
        customer_name: customer_name,
        customer_phone: customer_phone,
        customer_cnic: customer_cnic,
        sub_total: sub_total,
        discount: discount,
        shipping: shipping,
        grand_total: grand_total,
        totalQty: totalQty,
        remaining_amount: remaining_amount,
        paid_amount: paid_amount,
        cash_payment: cash_payment,
        returned_cash: returned_cash,
        payment_type: payment_type,
        bank: bank,
        items: JSON.stringify(items)
    }, function (result, status) {
```

```
            {#window.location.href = '/sales/invoice/'+ result.invoice_id +/detail/;#}
            window.location.href = '/sales/invoices';
        }
    ).fail(function (xhr, status, errors) {
        alert('Something is wrong! please check all fields and try again');

    });
});

$('.new-customer').on('click', function () {
    $('.existing-customer').hide();
    $('.new-customer-form').show();
    $('.customer').val('');
});

$('.added-customer').on('click', function () {
    $('.existing-customer').show();
    $('.new-customer-form').hide();
    $('.customer_name').val('');
    $('.customer_phone').val('');
});

$('.payment_type').on('change', function () {
```

```
            if ($(this).val() == 'Cash') {
                $('.bank-wrapper').hide();
            } else {
                $('.bank-wrapper').show();
            }
        });


    });
    $('.btn-primary').on('click', function () {
        $('.item-error').hide();
    });
</script>
{% endblock %}
```

Main connector Folders

- **Base folder**

```
{% load static %}

{% load template_tags %}

<html>

<head>

   <meta charset="utf-8">

   <meta http-equiv="X-UA-Compatible" content="IE=edge">

   <title>Abu Abeer al Gawabry Trading </title>

   <meta content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no" name="viewport">

   <link rel="stylesheet" href="{% static 'bower_components/bootstrap/dist/css/bootstrap.min.css' %}">

   <link rel="stylesheet" href="{% static 'bower_components/font-awesome/css/font-awesome.min.css' %}">

   <link rel="stylesheet" href="{% static 'bower_components/Ionicons/css/ionicons.min.css' %}">

   <link rel="stylesheet" href="{% static 'bower_components/jvectormap/jquery-jvectormap.css' %}">

   <link rel="stylesheet" href="{% static 'dist/css/AdminLTE.min.css' %}">

   <link rel="stylesheet" href="{% static 'dist/css/skins/_all-skins.min.css' %}">

<link rel="stylesheet"
```

```
href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,600,700,300italic,400italic,600italic">

</head>

<body class="hold-transition skin-blue sidebar-mini">

<div class="wrapper">


   <header class="main-header">

<a href="#" class="logo">

<span class="logo-mini"><b>Trading</b></span>

<span class="logo-lg"><b>AAAG Trading</b></span>

      </a>


<nav class="navbar navbar-static-top">

<a href="#" class="sidebar-toggle" data-toggle="push-menu" role="button">

            <span class="sr-only">Toggle navigation</span>

         </a>

         <div  class="header">

            <span style="color:white;"><h4><b> Abu Abeer al Gawabry Trading</b></h4></span>

         </div>

<div class="navbar-custom-menu">

            <ul class="nav navbar-nav">
```

```
<li class="dropdown notifications-menu">
  <a href="#" class="dropdown-toggle" data-toggle="dropdown">
    Notifications <i class="fa fa-bell-o"></i>
    <span class="label label-danger notification_count">10</span>
  </a>
  {% get_products as p_n %}
  <ul class="dropdown-menu">
    <li class="header">Lower Quantity Notifications</li>

    <li class="header">
      <table class="table table-bordered table-hover">
        {% for p in p_n %}
          {% if p.product_available_items <= p.notify_qty %}
            <tr>
              <th><strong>{{ p.name|title }}</strong></th>
              <td style="color:red">{{ p.product_available_items|floatformat:-2 }}</td>
            </tr>
          {% endif %}
        {% endfor %}
      </table>
    </li>
    <li class="footer"><a href="{% url 'product:list' %}">View all</a></li>
```

```
                    </ul>
                </li>
            </ul>
        </div>


    </nav>
  </header>
<aside class="main-sidebar">
<section class="sidebar">
<ul class="sidebar-menu" data-widget="tree">
            <li class="header">Main Navigation</li>
            <li class="treeview {% block product %}{% endblock %}">
                <a href="#">
                    <i class="fa fa-product-hunt"></i> <span>Product</span>
                    <span class="pull-right-container">
                        <i class="fa fa-angle-left pull-right"></i>
                    </span>
                </a>
                <ul class="treeview-menu">
                    <li class="{% block add_category %}{% endblock %}"><a href="{% url 'product:add_category' %}"><i class="fa fa-circle-o"></i>Add Category</a></li>
                    <li class="{% block add_product %}{% endblock %}"><a href="{% url 'product:add' %}"><i class="fa fa-circle-o"></i>Add Product</a></li>
```

```
                <li class="{% block product_list %}{% endblock %}"><a href="{% url 'product:list' %}"><i class="fa fa-circle-o"></i>List Products</a></li>
            </ul>
        </li>
        <li class="treeview {% block customer %}{% endblock %}">
            <a href="#">
                <i class="fa fa-user"></i> <span>Customers</span>
                <span class="pull-right-container">
                    <i class="fa fa-angle-left pull-right"></i>
                </span>
            </a>
            <ul class="treeview-menu">
                <li class="{% block customer_create %}{% endblock %}"><a href="{% url 'customer:add' %}"><i class="fa fa-circle-o"></i>Add Customer</a></li>
                <li class="{% block customer_list %}{% endblock %}"><a href="{% url 'customer:list' %}"><i class="fa fa-circle-o"></i>Customer List</a></li>
            </ul>
        </li>
        <li class="treeview {% block bank %}{% endblock %}">
            <a href="#">
                <i class="fa fa-university"></i> <span>Banking System</span>
                <span class="pull-right-container">
                    <i class="fa fa-angle-left pull-right"></i>
```

```
            </span>
          </a>
          <ul class="treeview-menu">
            <li class="{% block add_bank %}{% endblock %}"><a href="{% url 'bank:add' %}"><i class="fa fa-circle-o"></i>Add Bank</a></li>
            <li class="{% block bank_list %}{% endblock %}"><a href="{% url 'bank:list' %}"><i class="fa fa-circle-o"></i>Bank List</a></li>
          </ul>
        </li>
        <li class="header">Sales & Expenses</li>
        <li class="treeview {% block expense %}{% endblock %}">
          <a href="#">
            <i class="fa fa-money"></i> <span>Expense System</span>
            <span class="pull-right-container">
              <i class="fa fa-angle-left pull-right"></i>
            </span>
          </a>
          <ul class="treeview-menu">
            <li class="{% block add_expense %}{% endblock %}"><a href="{% url 'expense:add' %}"><i class="fa fa-circle-o"></i> <span>Add Expense</span></a></li>
            <li class="{% block expense_list %}{% endblock %}"><a href="{% url 'expense:list' %}"><i class="fa fa-circle-o"></i>Expense List</a></li>
          </ul>
        </li>
```

```
<li class="treeview {% block sales %}{% endblock %}">
  <a href="#">
    <i class="fa fa-shopping-cart"></i> <span>Sales</span>
    <span class="pull-right-container">
      <i class="fa fa-angle-left pull-right"></i>
    </span>
  </a>
  <ul class="treeview-menu">
    <li class="{% block create_invoice %}{% endblock %}"><a href="{% url 'sales:invoice_create' %}"><i class="fa fa-circle-o"></i> <span>Create Invoice</span></a></li>
    <li class="{% block invoice_list %}{% endblock %}"><a href="{% url 'sales:invoice_list' %}"><i class="fa fa-circle-o"></i>Invoices List</a></li>
  </ul>
</li>
<li class="header">Logs & Reports</li>
<li class="treeview {% block stock_logs %}{% endblock %}">
  <a href="#">
    <i class="fa fa-history"></i> <span>Stock Logs</span>
    <span class="pull-right-container">
      <i class="fa fa-angle-left pull-right"></i>
    </span>
  </a>
```

```
                <ul class="treeview-menu">
                    <li class="{% block daily_stock_logs %}{% endblock %}"><a href="{% url 'common:daily_logs' %}"><i class="fa fa-circle-o"></i> <span>Daily Logs</span></a></li>
                    <li class="{% block monthly_stock_logs %}{% endblock %}"><a href="{% url 'common:monthly_logs' %}"><i class="fa fa-circle-o"></i>Monthly Logs</a></li>
                </ul>
            </li>
            <li class="treeview {% block reports %}{% endblock %}">
                <a href="#">
                    <i class="fa fa-copy"></i> <span>Reports</span>
                    <span class="pull-right-container">
                        <i class="fa fa-angle-left pull-right"></i>
                    </span>
                </a>
                <ul class="treeview-menu">
                    <li class="{% block monthly_reports %}{% endblock %}"><a href="{% url 'common:reports' %}"><i class="fa fa-circle-o"></i> <span>Monthly Reports</span></a></li>
                </ul>
            </li>
            <li class="header">Admin</li>
            <li>
```

```
                <a target="_blank" href="/admin/"><i class="fa fa-sign-out"></i> <span>AdminPanel</span></a>
            </li>
            <li class="header">User</li>
             <li>
               <a href="{% url 'common:logout' %}"><i class="fa fa-sign-out"></i> <span>Logout</span></a>
            </li>
          </ul>
       </section>
    </aside>

{% block content %}{% endblock %}
footer class="main-footer">
       <div class="pull-right hidden-xs">
          <b>Version</b> 1.1.0
       </div>
       <strong>Copyright &copy; 2022 <a href="#">Abu Abeer al Gawabry Trading</a>.</strong> All rights
       reserved.
    </footer>
</div>
```

```
{% block script %}
   <script src="{% static 'js/jquery.js' %}"></script>
   <script src="{% static 'js/jquery-1.8.3.min.js' %}"></script>
   <script src="{% static 'bower_components/jquery/dist/jquery.min.js' %}"></script>
   <script src="{% static 'bower_components/bootstrap/dist/js/bootstrap.min.js' %}"></script>
   <script src="{% static 'bower_components/fastclick/lib/fastclick.js' %}"></script>
   <script src="{% static 'dist/js/adminlte.min.js' %}"></script>
   <script src="{% static 'bower_components/jquery-sparkline/dist/jquery.sparkline.min.js' %}"></script>
   <script src="{% static 'plugins/jvectormap/jquery-jvectormap-1.2.2.min.js' %}"></script>
   <script src="{% static 'plugins/jvectormap/jquery-jvectormap-world-mill-en.js' %}"></script>
   <script src="{% static 'bower_components/jquery-slimscroll/jquery.slimscroll.min.js' %}"></script>
   <script src="{% static 'bower_components/chart.js/Chart.js' %}"></script>
   <script src="{% static 'dist/js/demo.js' %}"></script>

   <script>
      var notification_count = 0;
      {% for p in p_n %}
         {% if p.product_available_items <= p.notify_qty %}
               notification_count += 1;
```

```
            {% endif %}
        {% endfor %}


        $('.notification_count').text(notification_count);
    </script>


{% endblock %}
</body>
</html>
{% extends "base.html"%}
{% block content %}
{% load staticfiles %}
<div class="content-wrapper">
    <!-- Content Header (Page header) -->
    <section class="content-header">
      <h1>
        Dashboard
        <small>Version 2.0</small>
      </h1>
      <ol class="breadcrumb">
        <li><a href="#"><i class="fa fa-dashboard"></i> Home</a></li>
        <li class="active">Dashboard</li>
      </ol>
```

```
</section>

<!-- Main content -->
<section class="content">
  <!-- Info boxes -->
  <div class="row">
    <div class="col-md-3 col-sm-6 col-xs-12">
      <div class="info-box">
        <span class="info-box-icon bg-aqua"><i class="ion ion-ios-gear-outline"></i></span>

        <div class="info-box-content">
          <span class="info-box-text">CPU Traffic</span>
          <span class="info-box-number">90<small>%</small></span>
        </div>
        <!-- /.info-box-content -->
      </div>
      <!-- /.info-box -->
    </div>
    <!-- /.col -->
    <div class="col-md-3 col-sm-6 col-xs-12">
      <div class="info-box">
        <span class="info-box-icon bg-red"><i class="fa fa-google-plus"></i></span>
```

```
          <div class="info-box-content">
            <span class="info-box-text">Likes</span>
            <span class="info-box-number">41,410</span>
          </div>
          <!-- /.info-box-content -->
        </div>
        <!-- /.info-box -->
      </div>
      <!-- /.col -->
```

- **Login Page**

```
{% load static %}
<html>
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <title>Log in</title>
  <!-- Tell the browser to be responsive to screen width -->
  <meta content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no" name="viewport">
  <!-- Bootstrap 3.3.7 -->
```

```
    <link rel="stylesheet" href="{% static 'bower_components/bootstrap/dist/css/bootstrap.min.css' %}">
    <!-- Font Awesome -->
    <link rel="stylesheet" href="{% static 'bower_components/font-awesome/css/font-awesome.min.css' %}">
    <!-- Ionicons -->
    <link rel="stylesheet" href="{% static 'bower_components/Ionicons/css/ionicons.min.css' %}">
    <!-- Theme style -->
    <link rel="stylesheet" href="{% static 'dist/css/AdminLTE.min.css' %}">
    <!-- iCheck -->
    <link rel="stylesheet" href="{% static 'plugins/iCheck/square/blue.css' %}">
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,600,700,300italic,400italic,600italic">
</head>
<body class="hold-transition login-page">
<div class="login-box">
    <div class="login-logo">
        <a href="#"><b>Abu Abeer al Gawabry Trading</b></a>
    </div>
    <!-- /.login-logo -->
    <div class="login-box-body">
        <p class="login-box-msg">Sign in to start your session</p>
```

```
<form action="{% url 'common:login' %}" method="post">
    {% csrf_token %}
    <div class="form-group has-feedback">
        <input type="text" class="form-control" name="username" placeholder="Username">
        <span class="glyphicon glyphicon-envelope form-control-feedback"></span>
    </div>
    <div class="form-group has-feedback">
        <input type="password" class="form-control" placeholder="Password" name="password">
        <span class="glyphicon glyphicon-lock form-control-feedback"></span>
    </div>
    <div class="row">
        <div class="col-xs-8">
            <div class="checkbox icheck">
                <label>
                    <input type="checkbox"> Remember Me
                </label>
            </div>
        </div>
<div class="col-xs-4">
```

```
                <button type="submit" class="btn btn-primary btn-block btn-flat">Sign In</button>
            </div>
<script src="{% static 'plugins/iCheck/icheck.min.js' %}"></script>
<script>
   $(function () {
      $('input').iCheck({
         checkboxClass: 'icheckbox_square-blue',
         radioClass: 'iradio_square-blue',
         increaseArea: '20%' /* optional */
      });
   });
</script>
</body> </html>
```

1. Survey Questions

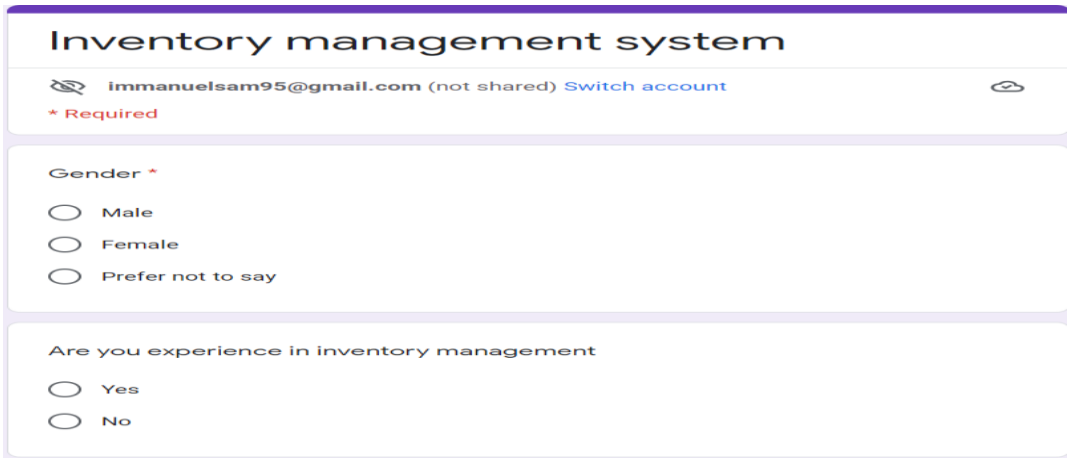


Figure 83:Survey questions 1&2

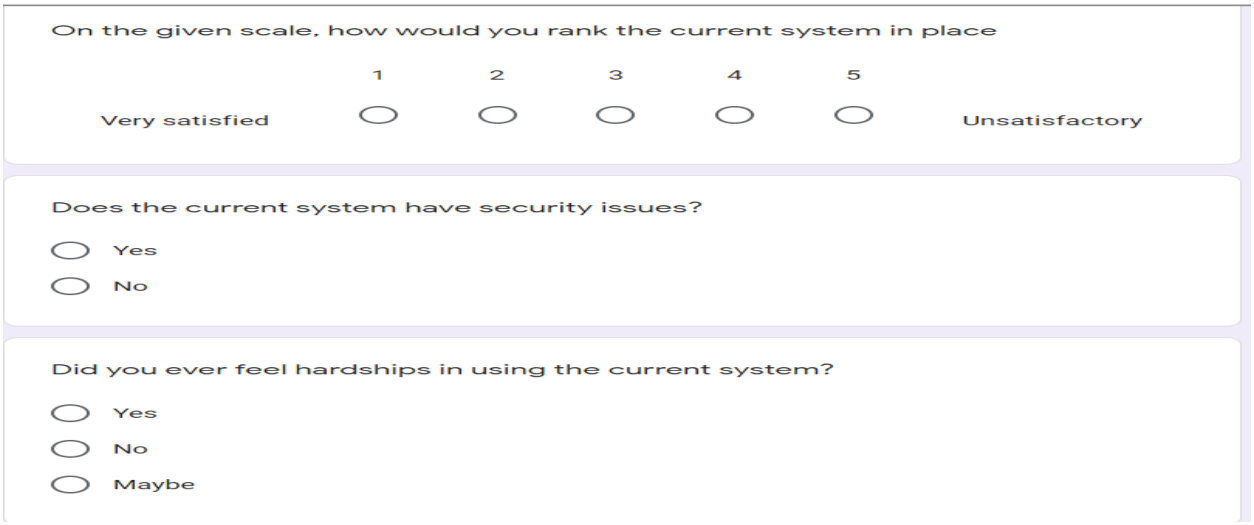


Figure 84: Survey question 3,4&5

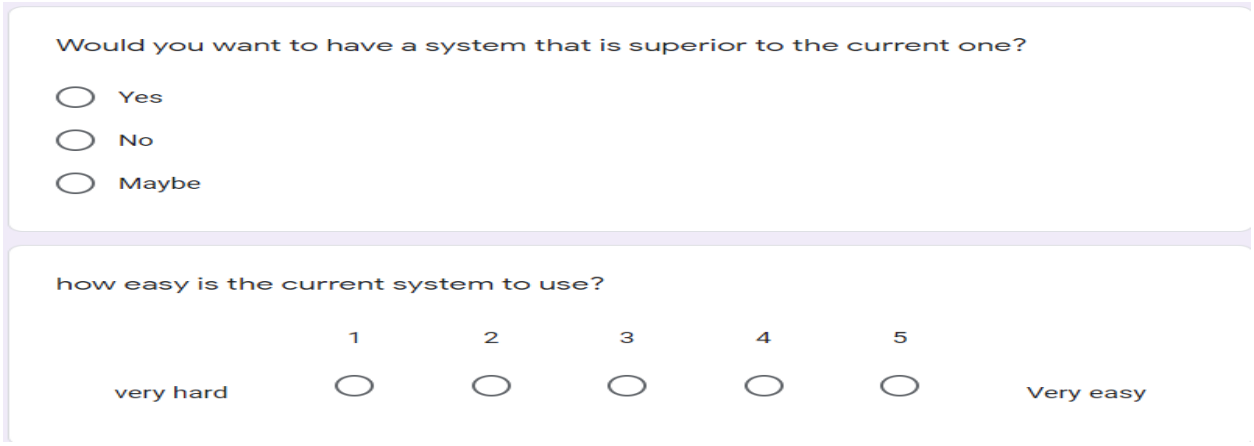
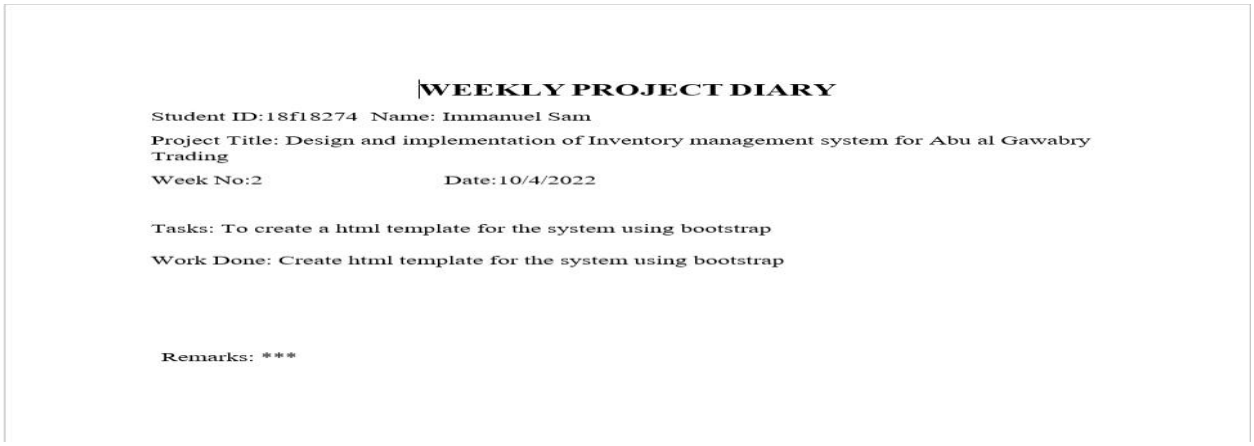


Figure 85: Survey question 6&7

2. Weekly diaries

Weekly diary 2



WEEKLY PROJECT DIARY

Student ID:18f18274 Name: Immanuel Sam

Project Title: Design and implementation of Inventory management system for Abu al Gawabry Trading

Week No:2 Date:10/4/2022

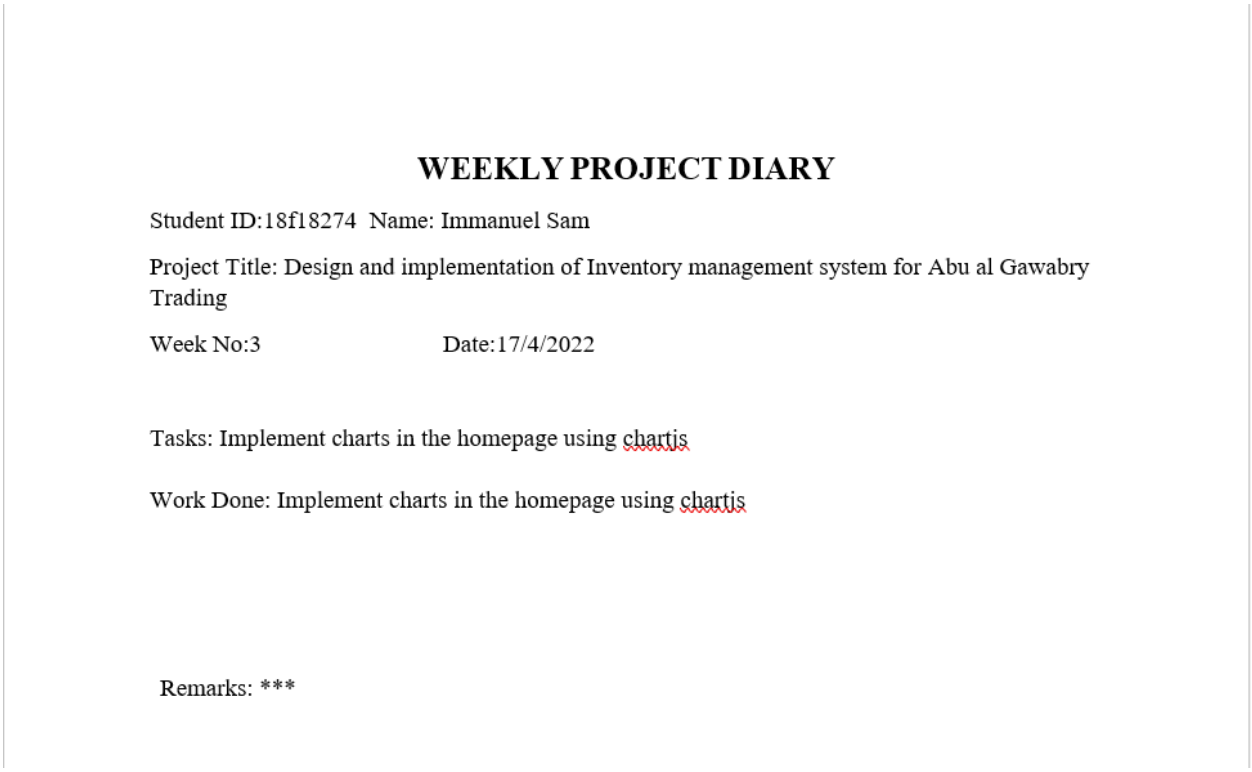
Tasks: To create a html template for the system using bootstrap

Work Done: Create html template for the system using bootstrap

Remarks: ***

Figure 86:Weekly diary 2

Weekly diary 3



WEEKLY PROJECT DIARY

Student ID:18f18274 Name: Immanuel Sam

Project Title: Design and implementation of Inventory management system for Abu al Gawabry Trading

Week No:3 Date:17/4/2022

Tasks: Implement charts in the homepage using chartis

Work Done: Implement charts in the homepage using chartis

Remarks: ***

Figure 87: Weekly diary 3

Weekly diary 4

WEEKLY PROJECT DIARY

Student ID:18f18274 Name: Immanuel Sam

Project Title: Design and implementation of Inventory management system for Abu al Gawabry Trading

Week No:4 Date:24/4/2022

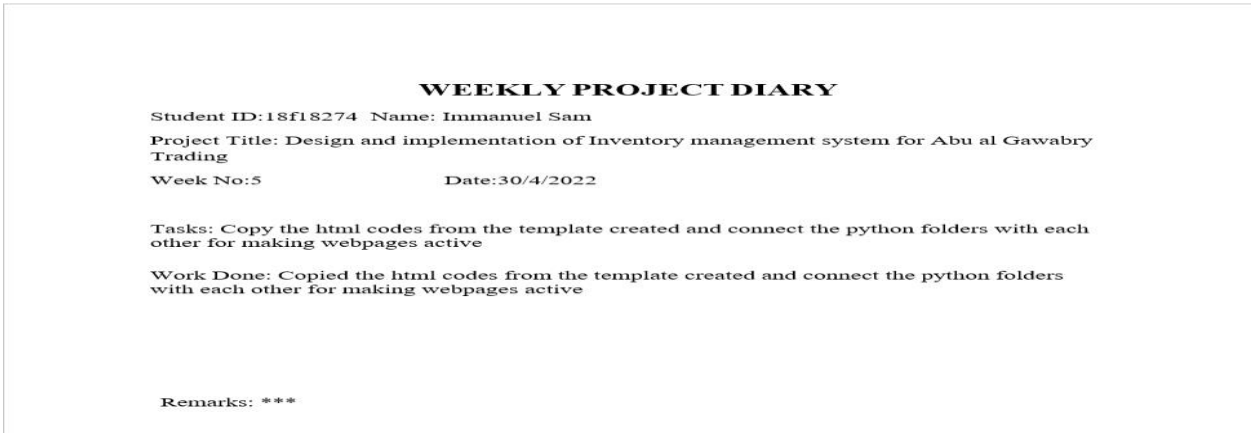
Tasks: connect visual studio code to python and Django and create manage.py file

Work Done: manage.py file was created to start the web app

Remarks: ***

Figure 88: Weekly diary 4

Weekly diary 5



WEEKLY PROJECT DIARY

Student ID:18f18274 Name: Immanuel Sam

Project Title: Design and implementation of Inventory management system for Abu al Gawabry Trading

Week No:5 Date:30/4/2022

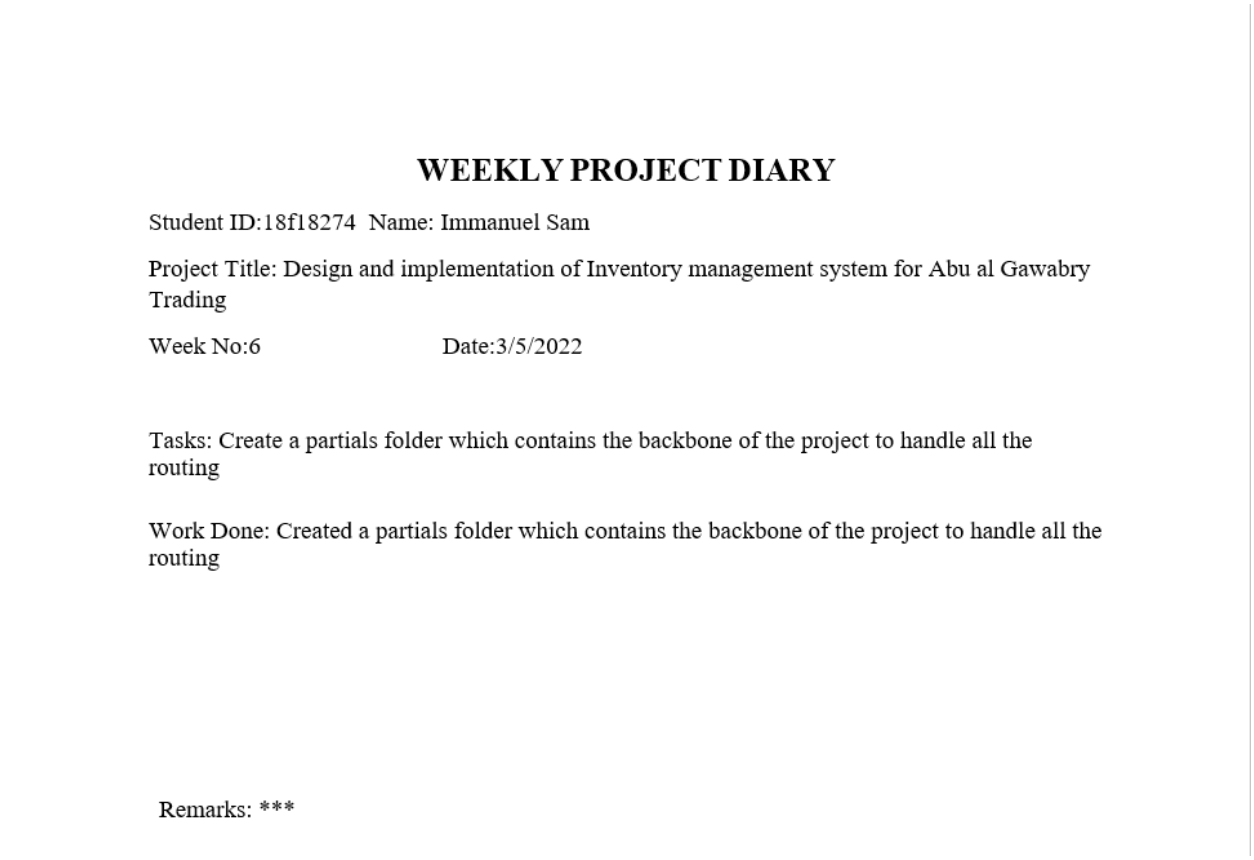
Tasks: Copy the html codes from the template created and connect the python folders with each other for making webpages active

Work Done: Copied the html codes from the template created and connect the python folders with each other for making webpages active

Remarks: ***

Figure 89: Weekly diary 5

Weekly diary 6



WEEKLY PROJECT DIARY

Student ID:18f18274 Name: Immanuel Sam

Project Title: Design and implementation of Inventory management system for Abu al Gawabry Trading

Week No:6 Date:3/5/2022

Tasks: Create a partials folder which contains the backbone of the project to handle all the routing

Work Done: Created a partials folder which contains the backbone of the project to handle all the routing

Remarks: ***

Figure 90:Weekly diary 6

Weekly diary 7

WEEKLY PROJECT DIARY

Student ID:18f18274 Name: Immanuel Sam

Project Title: Design and implementation of Inventory management system for Abu al Gawabry Trading

Week No:7 Date:7/5/2022

Tasks: Create the Django admin framework which opens the link to the main framework

Work Done: Created the Django admin framework which opens the link to the main framework

Remarks: ***

Figure 91: Weekly diary 7

Weekly diary 8

WEEKLY PROJECT DIARY

Student ID:18f18274 Name: Immanuel Sam

Project Title: Design and implementation of Inventory management system for Abu al Gawabry Trading

Week No:8 Date:12/5/2022

Task: Test the Add product function and make the code suitable to understand the template input by the user for this function

Work Done: Tested the Add product function and make the code suitable to understand the template input by the user for this function

Remarks: ***

Figure 92: Weekly diary 8

3. No Objection Certificate

Abu Abeer al Gawabry Trading

Date: January 5,2022

To: Immanuel Sam

Student of Data analytics in Middle East College

Subject: Implementation of a Web based inventory management system

Dear Immanuel,

This letter certifies that the company gas provided you the permission to design and implement a web-based inventory management system for our company.

This approval form was provided to you at your request

Best regards,

Hamad Barka

Figure 93: NOC

4. Project initiation report

Project Initiation Report

1. Proposed Project Title:

Construction material management system for Abu Abeer Al gawabry trading using Django.

For whom is the system being developed for?

The system is being developed for Abu Abeer Al gawabry trading. It is a building material company located in hamriya.

Abu Abeer Al gawabry trading is a company that focuses on selling building material and contracting building material. They work with clients such as Petroleum development Oman(PDO). The company was established in 2010.

2. Please describe the type of the project being planned.

The project being planned is a live project for Abu Abeer Al gawabry trading.

The project will give the company a new system that will be used to store, add, delete, update building materials, employer details, customer details from the company database.

The existing system does not have enough functionalities for a good inventory system. It is an outdated and basic system.

The proposed system will change all that and give them a better system that can separate the users based on privileges like manager, customer, and employee with each of them having their own privilege.

The project will be built using python and Django module for python. The database will be within python

Django is a Python-based free and open-source web framework that follows the model–template–views architectural pattern

Django will give the project a clean design.

The project will be a fully live project with the ability to host in a live server by sharing a key.

The project will be coded on Microsoft VS code using python language with the Django plugin for hosting the web application.

3. State the reason(s) for selecting this topic (pls. mention the nature of the problem that is currently being faced and the key benefits that can be achieved by this project)

Abu Abeer Al gawabry trading currently has a system that is outdated and can only do basic functions like adding, updating, deleting items along with basic calculation functions.

The proposed system will make everything better and will give it UI, login function for staff, customers, a username, and password system, retrieving username and password if the staff or customers forget it and a better overall template. The proposed system will also have a better inventory management system.

The project will have a separate login for Manager, employees, and customers. Each of them will have different privileges. The manager will be able to overlook all the employees and their information. The manager can see how many items have been ordered, by who its orders and the date its ordered.

Employees will be able to add, insert, update delete the items in the store and customers will be able to choose what they want to order and add them to a cart.

There will be a create account page which will allow the customer to create an account. If the customer, employee or manager forget their password, they will get an email to change it.They can even add a profile image.

The system will also a visualized chart showing all the items that are available in the inventory.

4. What will be the main challenge(s) for you in doing this project?

The main challenges faced will be the coding using python and Django. Learning the modules and applying it in a live project will be a little challenge.

Name of the Project Supervisor: Mr. Puttaswamy Malali

Name & ID of the Student: Immanuel Sam(18f18274)

Signature of the Student:

Contact No(s): 78550898

Figure 94: Project initiation report

5. Research ethics approval form

Certificate of Ethical Approval

RollNumber 18F18274

Student Name IMMANUEL SAM SUHIRTHARAJ

Semester 2021 Fall

Project Title

Construction Material Management System For Abu Abeer Al Gawabry Trading Using Django Framework

This is to certify that the above named student has completed the Middle East College Ethical Approval process and their project has been confirmed and approved as Low Risk.

Supervisor Puttaswamy Malali

Date of Approval Jan 11, 2022

Figure 95: Research ethics approval

6. Digital Certificates

1. KPMG Virtual internship

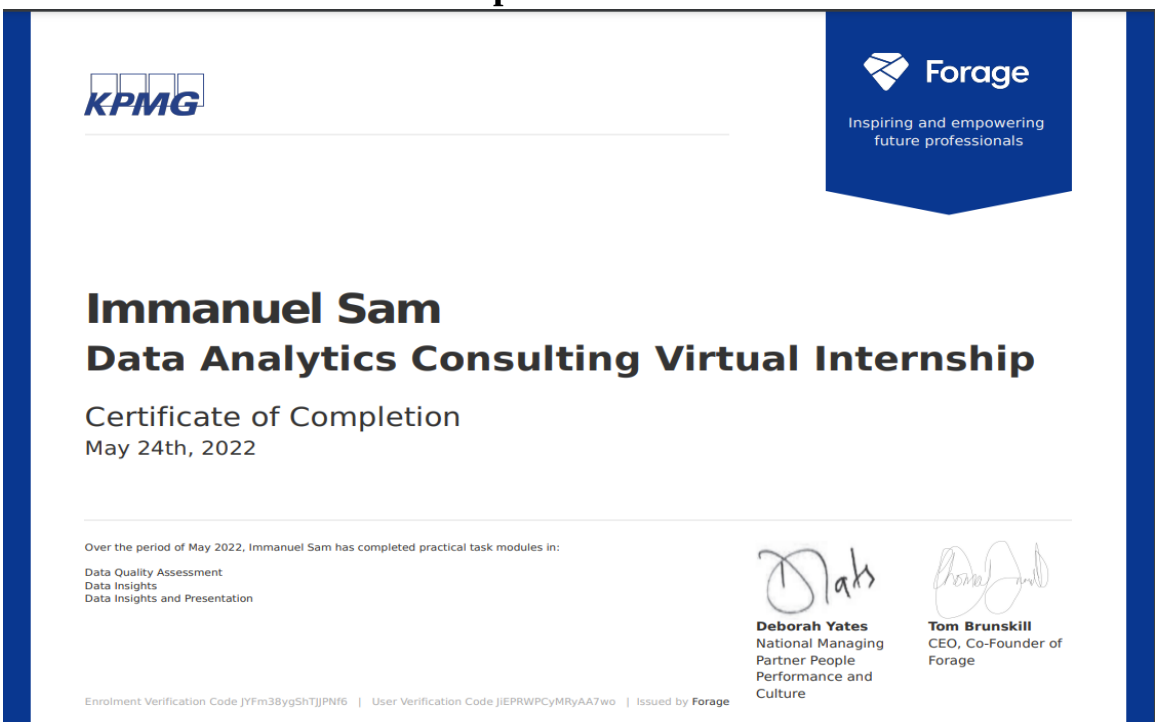


Figure 96: Certificate1

2. Accenture virtual internship certificate

Figure 97: Certificate 2

3. Goldman Sachs engineering program

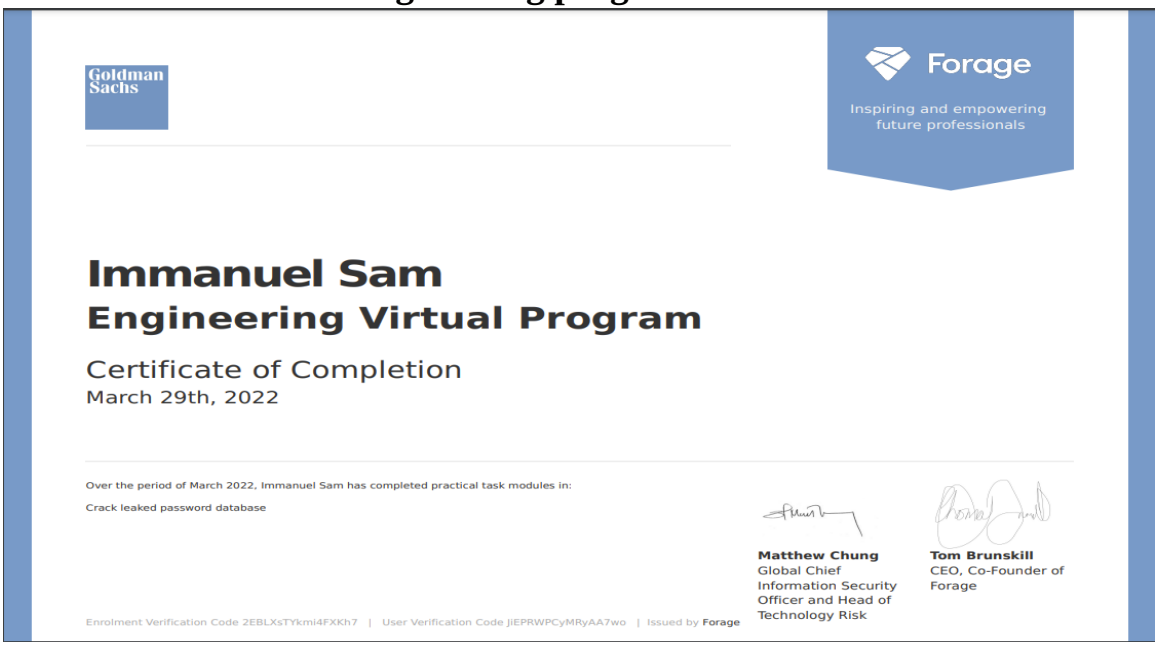


Figure 98: Certificate 3

4. Data analysis python Badge IBM

Figure 99: Badge 1

5. ANZ Virtual internship

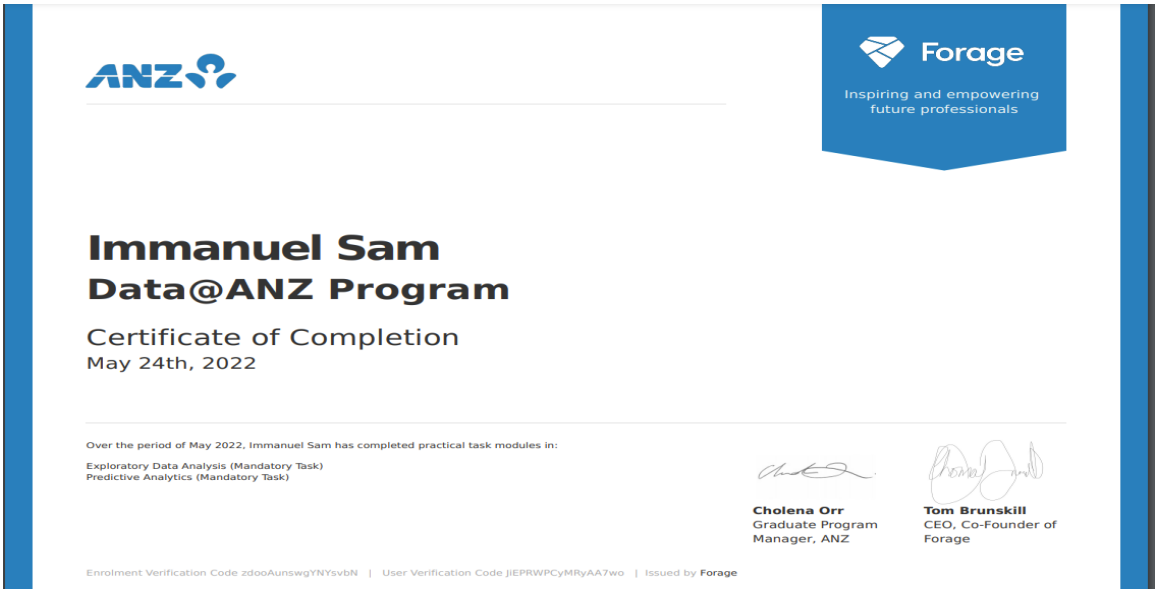


Figure 100: certificate 4

6. General Electronics virtual internship

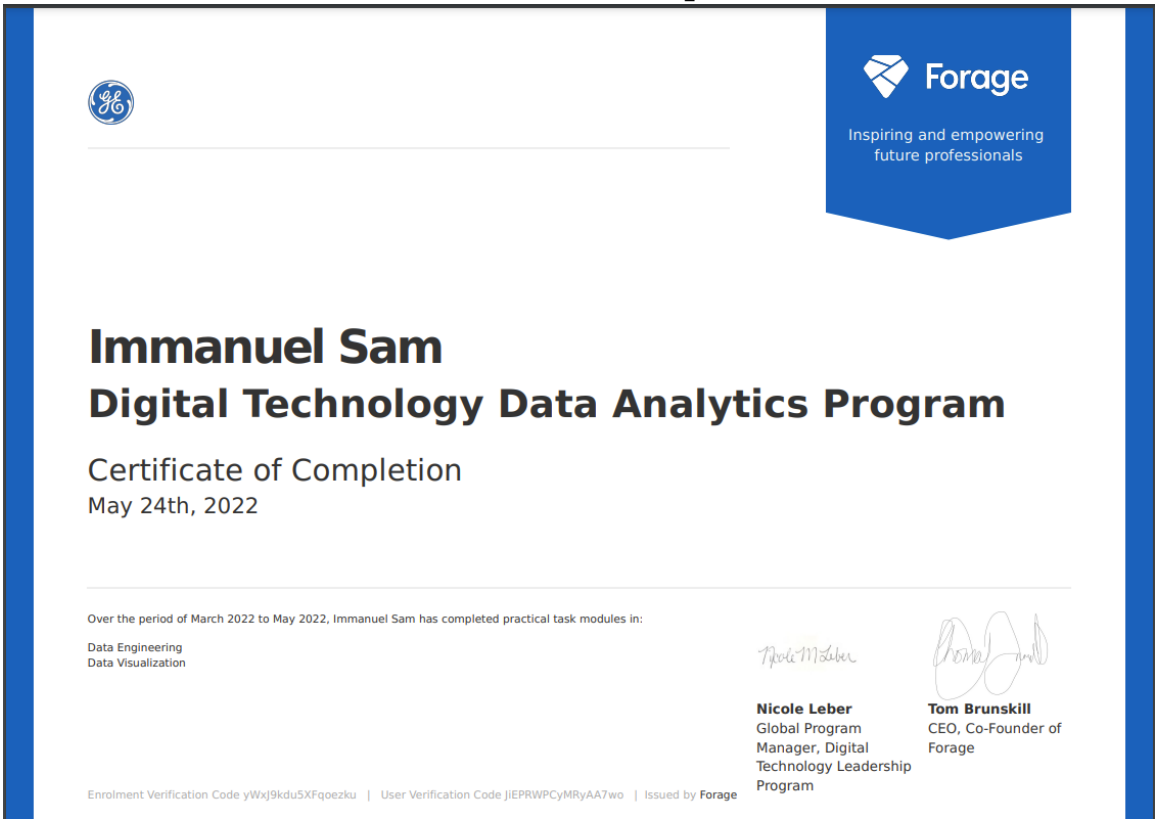


Figure 101: certificate

References

1) Learn What Stakeholders Are And The Roles That They Play (2022) available from <https://www.investopedia.com/terms/s/stakeholder.asp>
2) Inventory Software: What Is Silver Inventory System? (2022) available from <http://www.inventory-system.com/silver1.html>
3) Online Inventory Management Software | Zoho Inventory (2022) available from <https://www.zoho.com/inventory/>
4) Sortly: Simple Inventory Management Software for Small Businesses (2022) available from <https://www.sortly.com/>
5) Cin7 Inventory Management: Our 2021 Review | Business.Org (2022) available from <https://www.business.org/finance/inventory-management/cin7-review/>
6) BrightpearlBigcommerce (2022) available from <https://www.bigcommerce.com/apps/brightpearl/>
7) Sheakh, T. (2018) "A Study Of Inventory Management System Case Study". Journal Of Dynamical And Control Systems 10 (10), 1176
8) Jonsson, P. and Mattsson, S. (2008) "Inventory Management Practices And Their Implications On Perceived Planning Performance". International Journal Of Production Research 46 (7), 1787-1812
9) Study of Visual Studio Code (2022) [online] available from <https://www21.in.tum.de/teaching/osp/WS20/assets/pr-plainer-vscode.pdf>
10) SDLC V-Model (2019) available from <https://www.tutorialspoint.com/sdlc/sdlc_v_model.htm#:~:text=The%20V%2Dmodel%20is%20an,for%20each%20corresponding%20development%20stage.>
11) What Is Research Methodology? Definition + Examples - Grad Coach (2020) available from <https://gradcoach.com/what-is-research-methodology/> [18 November 2021]
12) Wisestep. 2022. What is an Interview: Definition, Objectives, Types & Guidelines - Wisestep. [online] Available at: <https://content.wisestep.com/what-is-an-interview/>