Point Of Sales System SRS(Software Requirement Specifications)

1.1 Introduction:

1.1.1 Purpose of the system:

POS is a crucial tool for retail businesses, serving as the central hub for managing sales, inventory, payment processing, and customer interactions. Here are the key purposes and benefits of the system:

1. Streamlined Operations: It provides checkout efficiency which speeds up the checkout process by centralizing the database and provides real time tracking of inventory levels, reducing the risk of stock-outs and misplaced inventory.
2. Enhanced customer service: It helps to effectively personalize the promotion of products for each customer and supports various payment mediums.
3. Cost savings: By automating the sales tax and discount, it reduces risk of errors and staff time to focus on customer service.
4. Integration and flexibility: It is cloud based and is able to integrate with various tools.

It is better than existing systems in the following ways:

1. Security: All the transactions and transcripts are secured and are at end-to-end device;
2. Cross Platform: Any user can access the application on any device as the technologies used are platform independent.

1.1.2 Scope of the system:

- Design and development of a POS software system.
- Integration with inventory and accounting systems.
- Multi-user access with role-based permissions.
- Deployment on hardware (cash registers, barcode scanners, receipt printers).

1.2 Project feasibility study

1.2.1 **Technological feasibility**

- The required technologies are modern, well-documented, and widely used, ensuring availability of support and resources.
- Team familiarity with these tools and frameworks makes implementation feasible.

1.2.2 **Operational feasibility**

A business needs streamlined transaction processing, real-time inventory tracking and enhanced customer experience.

- The POS system aligns with the organization's goal of improving efficiency and customer satisfaction.
- Easy-to-use interfaces and automated workflows minimize training needs.

1.2.3 **Financial feasibility**

- **Initial cost**
- Software development tools and licensing (Node.js, MongoDB, etc.) are open-source, reducing costs.

- Hardware costs are moderate, with barcode scanners and receipt printers as optional expenses.
- **Potential savings**
- Reduced labor costs due to automation.
- Minimized errors in transaction processing and inventory management.
- **Return on Investment**
- Improved customer experience and operational efficiency will likely increase sales and profitability.
- Quick adoption and scalability ensure long-term financial benefits.

1.2.4 Legal and Compliance Authority

- Compliance with data protection laws (e.g., GDPR) ensures secure handling of customer and transaction data.
- Adherence to industry standards for digital payments enhances trust and reliability.
- Legal and compliance risks are minimal with proper measures.

1.2.5 Scheduling Feasibility

- The project can be completed in 3–4 months, including development, testing, and deployment phases.
- A detailed timeline with clearly defined milestones ensures smooth project execution.

1.3 Functional Requirements:

1.3.1. Sales Management

The system will handle all sales-related activities from item scanning to payment processing and receipt generation.

1.3.1.1. Item Scanning

Scanning the items using a barcode scanner to add them to the sales list is done here by the cashier.

1.3.1.2. Price Calculation

Automatic price calculator calculates the total price including taxes and discounts is done here and at time of payment to cashier window.

1.3.1.3. Payment Processing

It processes payments via cash, credit/debit cards, or digital wallets in cashier window.

1.3.1.4. Receipt Generation

It generates and print receipts for customers.

1.3.2. Inventory Management

The system will manage the stock levels, updates inventory after each sale, and alerts for low stock items.

1.3.2.1. Stock Entry

It inputs new stock into the system.

1.3.2.2. Stock Adjustment

It updates stock levels after sales or returns.

1.3.2.3. Inventory Tracking

It tracks inventory levels in real-time.

1.3.2.4. Low Stock Alerts

System will generate alerts when stock levels fall below the reorder threshold.

1.3.3. Customer Management

1.3.3.1. Customer Data Entry

Customer will register themselves through the bar code provided and can shop and scan without the help of the cashier or user themselves.

1.3.3.2. Purchase History Tracking

They can track and analyze customer purchase history and can learn on the purchasing patterns of the customer.

1.3.4. Automated Sales Promotion

Automated message alerts on offers and discounts on special events like birth-weeks, festivals etc., prior to one-two weeks of the event.

1.4 Non- Functional Requirements:

1.4.1 **Security:**

- **Password Protection**: The system should enforce strong password policies, including requirements for length, complexity, and periodic updates.
- **Data Encryption**: All sensitive data must be encrypted both in transit and at rest using industry-standard encryption protocols.
- **Access Control**: Implement role-based access control (RBAC) to ensure users can only access data and functions necessary for their roles.
- **Audit Logs**: Maintain comprehensive audit logs of all user activities and system events to detect and investigate security incidents.

1.4.2 **Reliability:**

- **Uptime**: The system must ensure 99.9% uptime, excluding scheduled maintenance periods.
- **Redundancy**: Implement redundant systems and failover mechanisms to ensure continuous operation in the event of hardware or software failures.
- **Error Handling**: Include robust error handling and logging to capture and address errors efficiently, minimizing impact on users.

1.4.3 **Availability:**

- **Quick Access**: Customer information and critical business functions should be quickly accessible from any authorized device.
- **Disaster Recovery**: Establish a comprehensive disaster recovery plan, including regular backups and offsite storage to ensure data can be restored quickly in case of data loss.
- **Load Balancing**: Use load balancing to distribute workloads evenly across servers, ensuring high availability even during peak usage times.

1.4.4 **Maintainability:**

- **Modular Design**: The application should be designed in a modular fashion to facilitate easy updates, enhancements, and bug fixes.
- **Documentation**: Maintain thorough documentation for system architecture, code, and user operations to assist in maintenance and onboarding of new developers.

1.4.5 **Portability:**

- **Cross-Platform Compatibility**: The application should be operable on multiple operating systems, including Windows, macOS, and Linux.
- **Device Independence**: Ensure the application functions seamlessly across various devices, including desktops, tablets, and smartphones.

- **Minimal Dependencies**: Reduce reliance on platform-specific features or proprietary technologies to enhance portability.

1.4.6 **Reusability:**

- **Modular Components**: Design system components to be reusable in different contexts, allowing for easy adaptation to new retail locations or business units.
- **Template-Based Setup**: Use templates for common business functions to streamline the setup process for new deployments.

1.5 Hardware and Software Requirements

1.5.1 Hardware

- Any devices which can support any browser, with an internet connection.
- (Can be Optional, depending upon the medium which the client chooses) Barcode scanner and receipt printer

1.5.2 Software

Backend

- Node.js
- Express.js

Frontend

- React.js, UI extensions

Database

- MongoDB

Other tools used in development

- Git for version control
- VS Code for development

1.6 Data Dictionary:

1.6.1 login_table

To store admin, customer, and cashier login details.

Primary Key: user_name

Sr. No.	Name	Datatype	Constraint	Description
1	user_name	Varchar(10)	Primary key	To store the user name
2	user_type	Char(1)	Not null	To store user type
3	password	nvarchar(10)	Not null	To store the password
4	user_id	Varchar(10)	Not null	Unique id of the user

1.6.2 sales_table

- To store sales transaction details.
- **Primary Key:** transaction_id

Sr. No.	**Name**	**Datatype**	**Constraint**	**Description**
1	transaction_id	varchar(20)	Primary key	To store the transaction ID
2	item_id	varchar(10)	Not null Foreign key reference	To store item ID,
3	quantity	int	Not null	To store the quantity sold
4	customer_id	Varchar(10)	Foreign key reference	To refer Customer table
5	employee_id	Varchar(10)	Foreign key reference	To refer employee table

1.6.3 inventory_table

- To manage inventory details..
- **Primary Key:** item_id

Sr. No.	Name	Datatype	Constraint	Description
1	item_id	Varchar(10)	Primary key	To store the item ID
2	item_name	Varchar(50)	Not null	To store the item name
3	stock_level	Int	Not null	To store the current stock level
4	reorder_level	Int	Not null	To store the reorder threshold
5	price	Decimal(10,2)	Not null	To store the item price

1.6.4 customer_table

To store customer information.

- **Primary Key:** customer_id

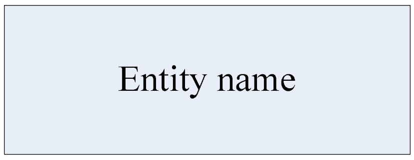
Sr. No.	Name	Datatype	Constraint	Description
1	customer_id	Varchar(10)	Primary key	To store the customer ID
2	first_name	Varchar(50)	Not null	To store the customer's first name
3	last_name	Varchar(50)	Not null	To store the customer's last name
4	email	Varchar(50)	Not null	To store the customer's email
5	phone_number	Varchar(15)	Not null	To store the customer's phone number

Diagram 1: Entity-Relationship Model (ER model)

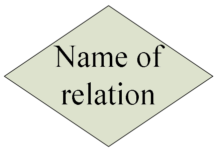
Entity-Relationship model is used to represent a logical design of a database to be created. In ER model, real world objects (or concepts) are abstracted as entities, and different possible associations among them are modeled as relationships. We represents the attributes, entities and relation using the ER diagram. Using this ER diagram, table structures are created, along with required constraints. Finally, these tables are normalized in order to remove redundancy and maintain data integrity. Thus, to have data stored efficiently, the ER diagram is to be drawn as much detailed and accurate as possible.

Symbols used in ER diagram:

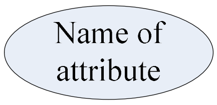
Entity



Relation



Attributes



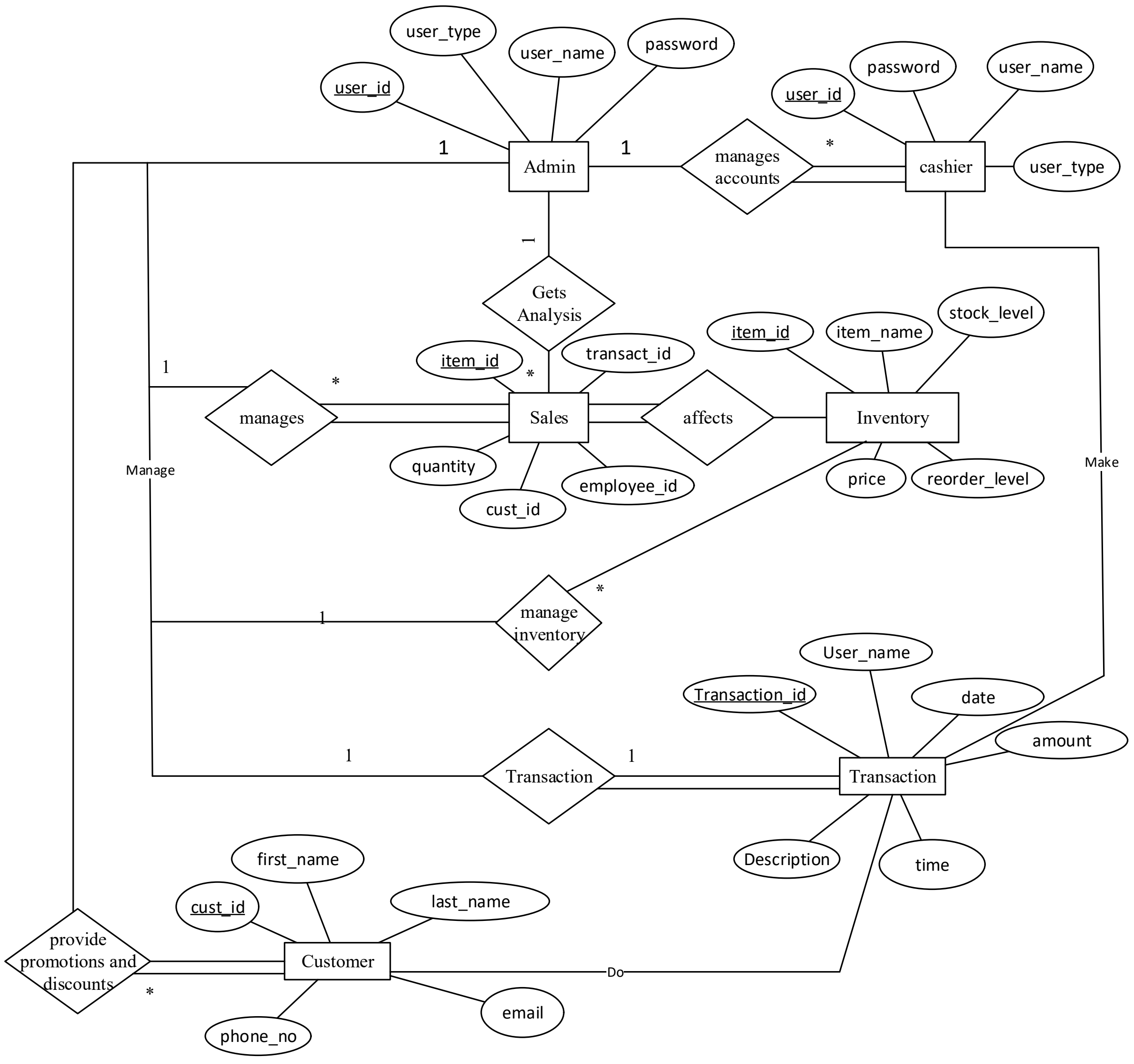


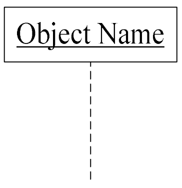
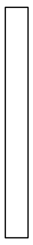
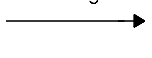
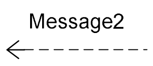
Figure 1.1 ER diagram

Diagram 2: Sequence diagram:

Sequence diagram represents the behavioural aspects of a system. Sequence diagram shows the interactions between the objects by means of passing messages from one object to another with respect to time in a system.

Sequence diagram contains the objects of a system and their life-line bar and the messages passing between them. Objects appear at the top portion of sequence diagram. Object is shown in a rectangle box. Name of object precedes a colon ':' and the class name, from which the object is instantiated. The whole string is underlined and appears in a rectangle box. A down-ward vertical line from object-box is shown as the life-line of the object. A rectangle bar on life-line indicates that it is active at that point of time. Messages are shown as an arrow from the life-line of sender object to the life-line of receiver object and labelled with the message name.

Symbols used in Sequence diagram:

Object	
Life line bar	
Call message	
Return message	

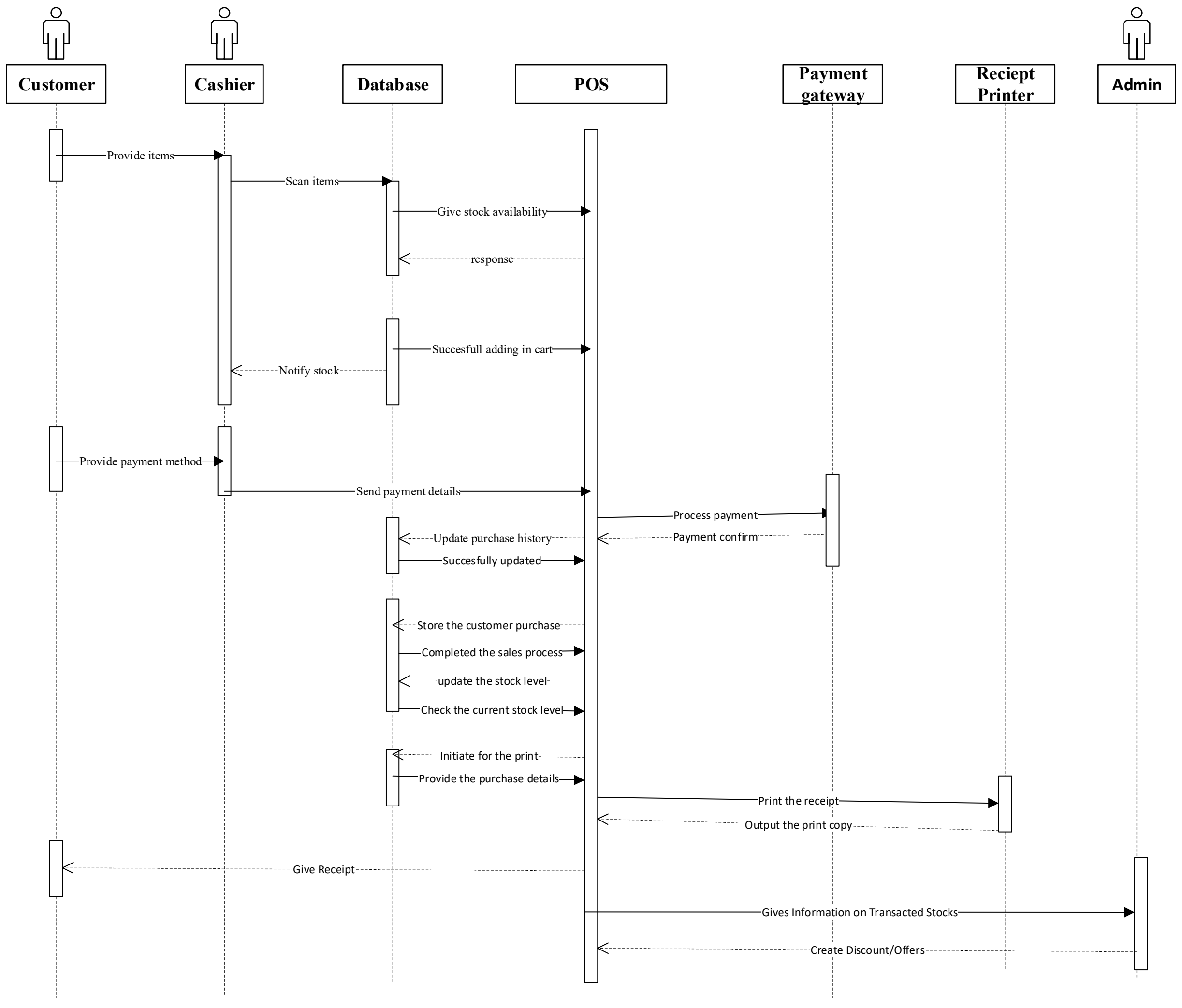
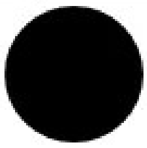
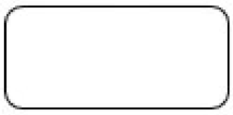
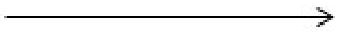
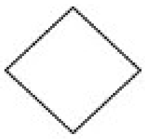
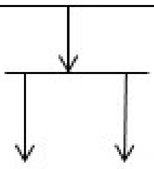
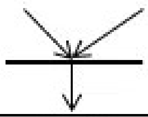
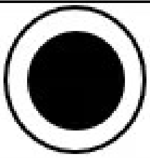


Figure 1.2 Sequence Diagram

Diagram 3:Activity Diagram

An activity diagram is a type of UML (Unified Modeling Language) diagram that represents the dynamic aspects of a system. It visually illustrates the flow of control or data within a system, detailing the sequence of activities and their interdependencies. Activity diagrams are particularly useful in modeling the workflow of a business process, the steps involved in a use case, or the flow of control in a system's operation.

Symbols used:

Sr. No	Name	Symbol
1.	Start Node	
2.	Action State	
3.	Control Flow	
4.	Decision Node	
5.	Fork	
6.	Join	
7.	End State	

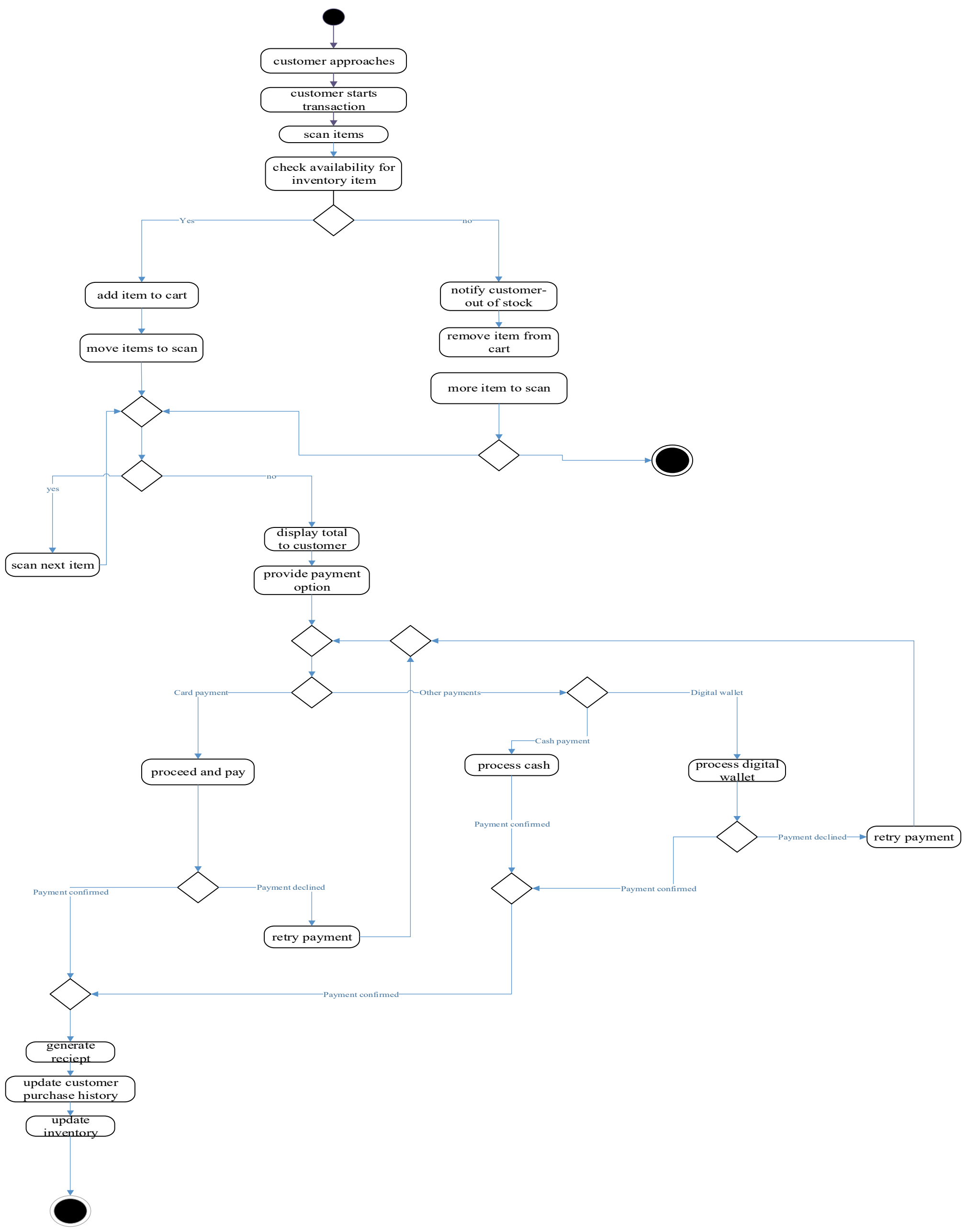
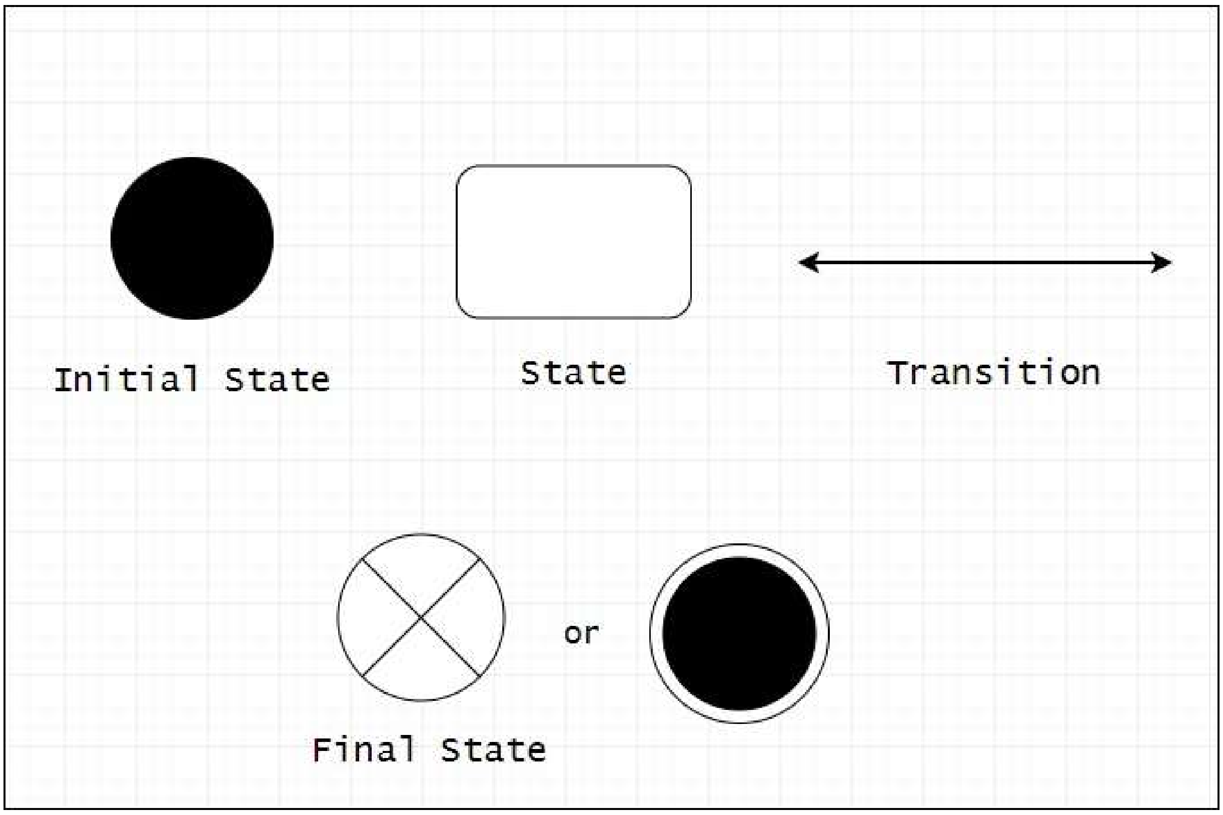


Figure 1.3 Activity Diagram

Diagram 4:State Diagram

A state diagram, also known as a state machine diagram, is another type of UML diagram that depicts the states of an object and the transitions between these states throughout its lifecycle. It is particularly useful for modeling the behavior of objects that have a finite number of conditions or states, making it essential for understanding complex systems where different conditions trigger different behaviors.

Symbols used:



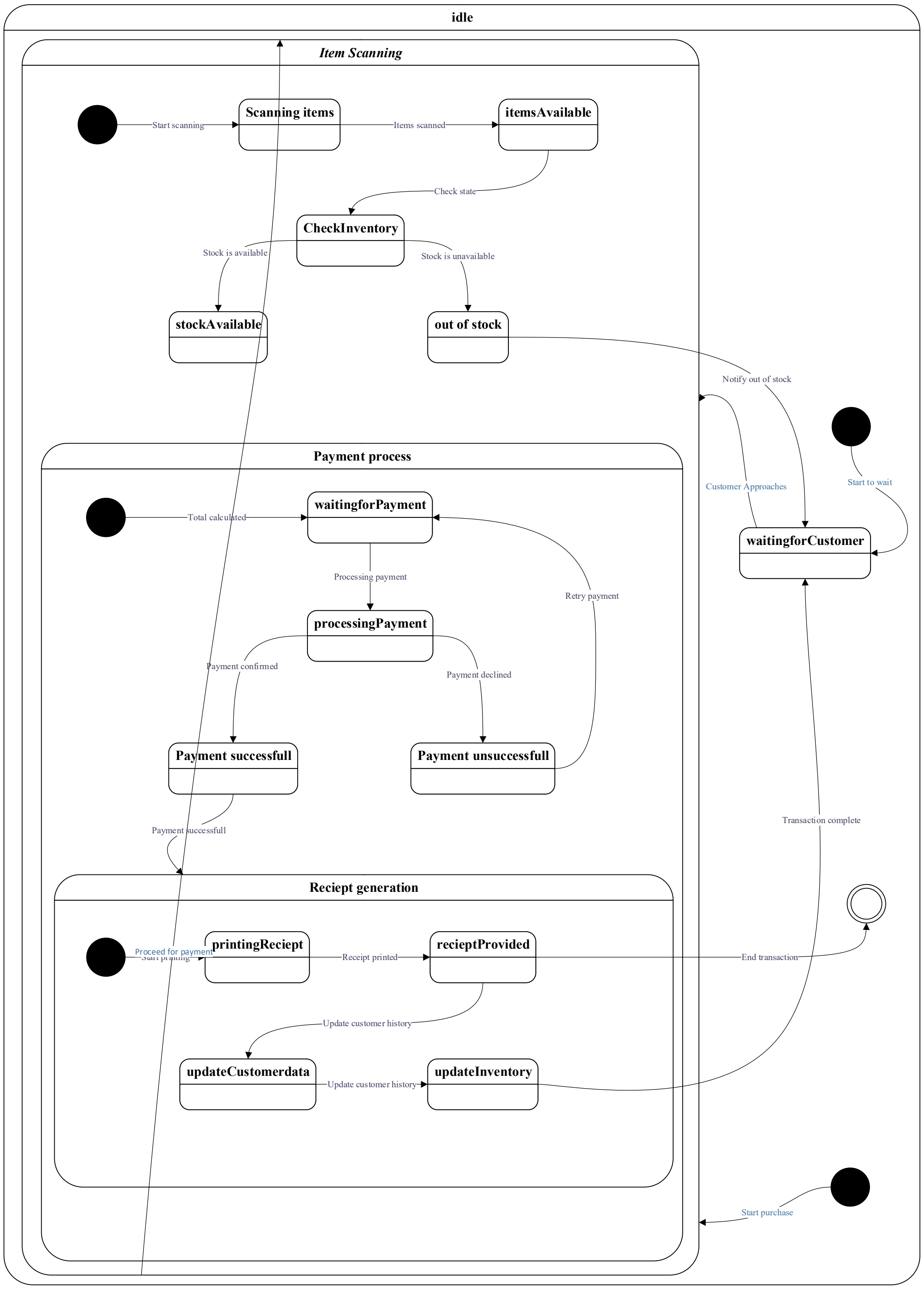


Figure 1.4 State Diagram

Diagram 5:Class Diagram

The class diagram is the main building block of object-oriented modelling. It is used both for general conceptual modelling of the systematics of the application, and for detailed modelling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

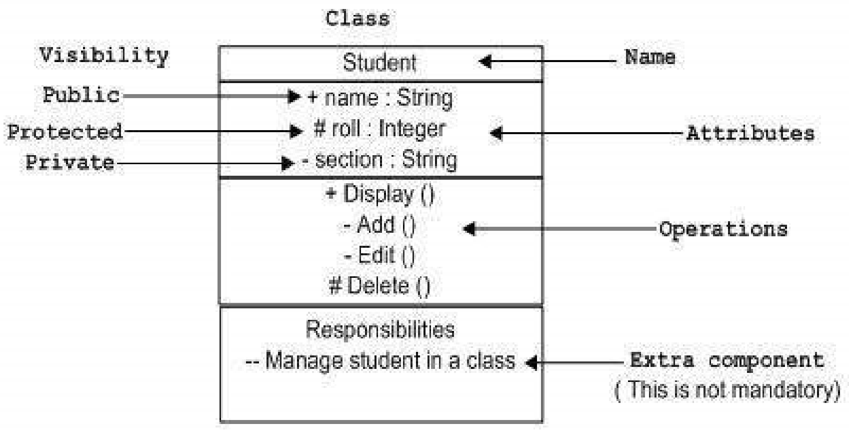
In the diagram, classes are represented with boxes that contain three compartments:

3. The top compartment contains the name of the class. It is printed in bold and centered, and the first letter is capitalized.
4. The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase.
5. The bottom compartment contains the operations the class can execute. They are also left-aligned and the first letter is lowercase.

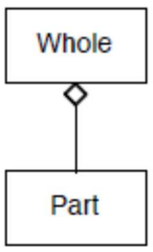
In the design of a system, a number of classes are identified and grouped together in a class diagram that helps to determine the static relations between them. With detailed modelling, the classes of the conceptual design are often split into a number of subclasses.

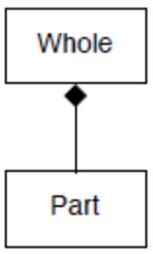
Symbols used in Class diagram:

Class

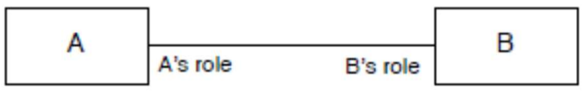


Aggregation **Composition**

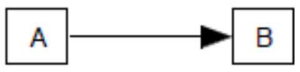




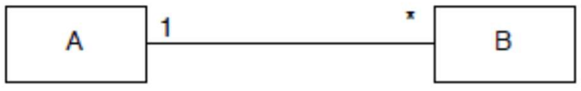
Association



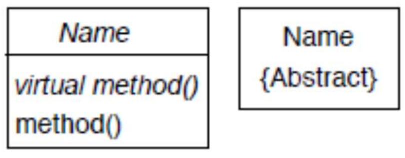
Inheritance



Multiplicity in Aggregation, Composition, or Association



Abstract Class



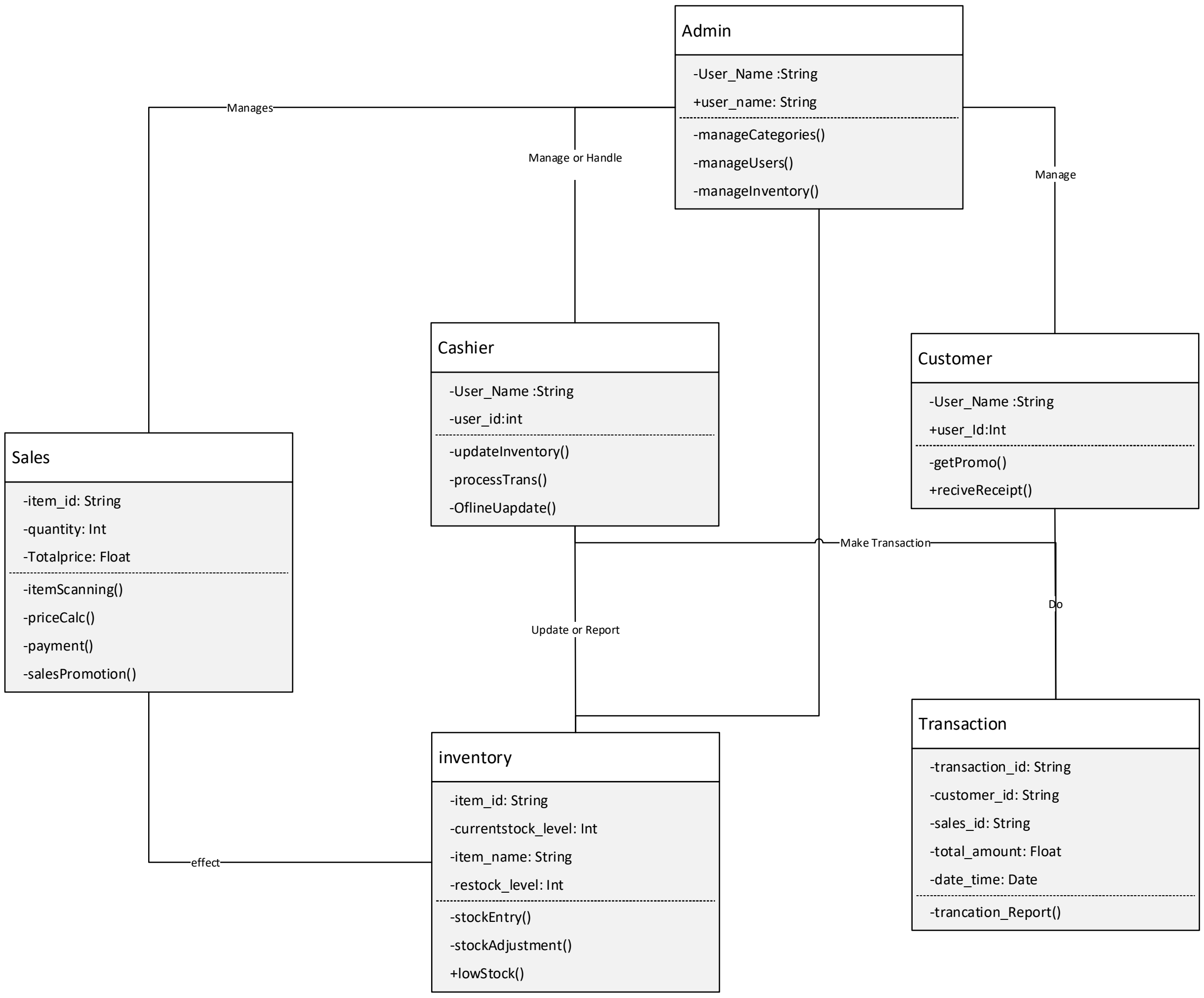
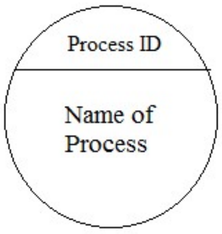
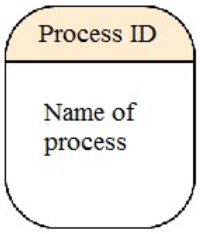
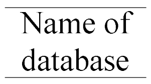
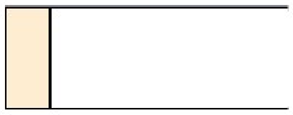
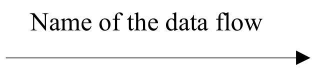
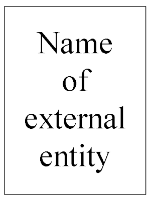


Figure 1.5 Class Diagram

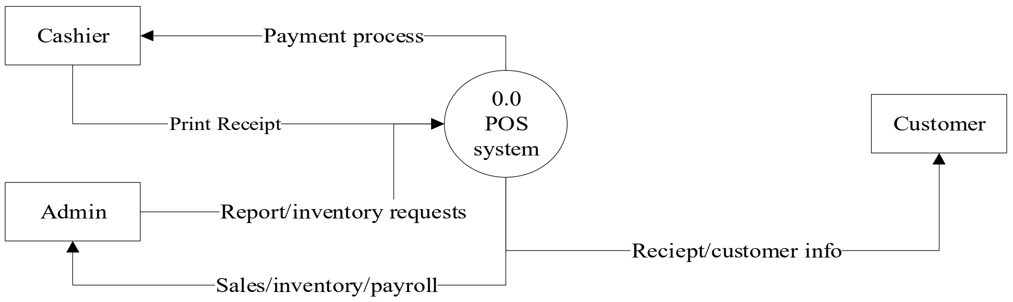
Diagram 6:Data Flow Diagram (DFD)

DFD provides the functional overview of a system. The graphical representation easily overcomes any gap between 'user and system analyst' and 'analyst and system designer' in understanding a system. Starting from an overview of the system it explores detailed design of a system through a hierarchy. DFD shows the external entities from which data flows into the process and also the other flows of data within a system. It also includes the transformations of data flow by the process and the data stores to read or write a data.

Symbols used in Data Flow diagram:

Process		or	
Data store		or	
Data flow			
External entity			

Level 0



Level 1

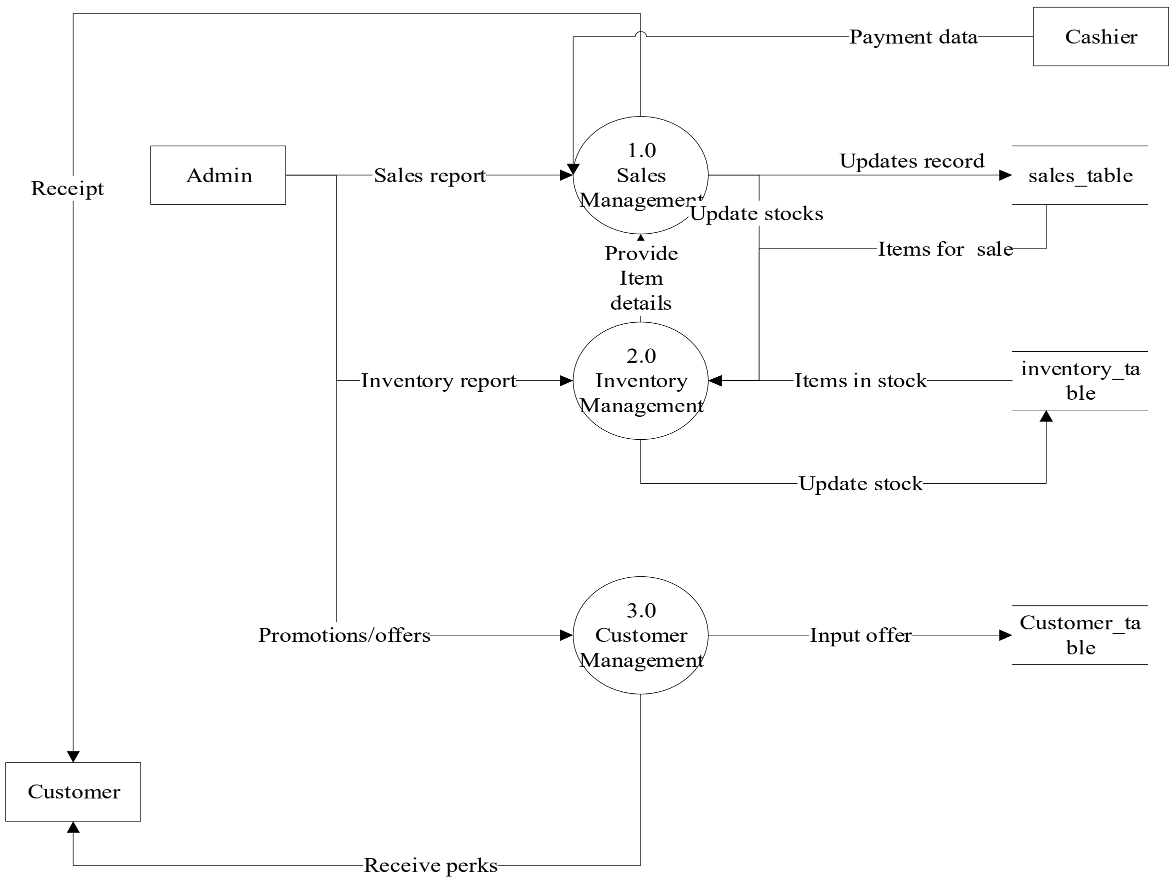
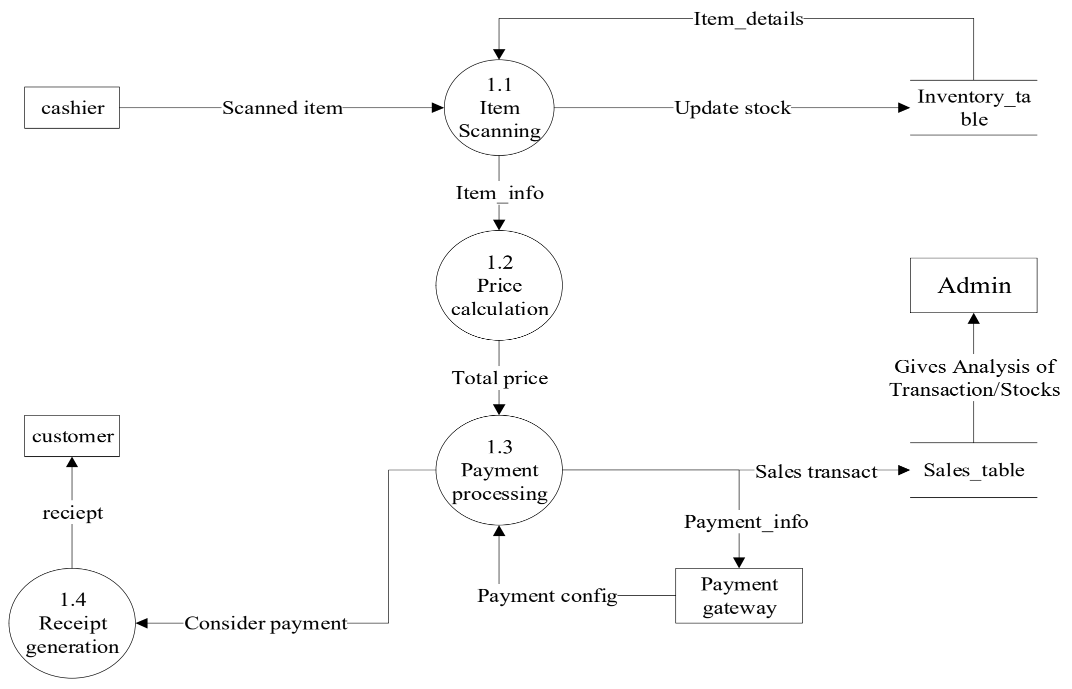


Figure 1.6.1 DFD diagram

Level 2 (Sales Management)



Level 2(customer management)

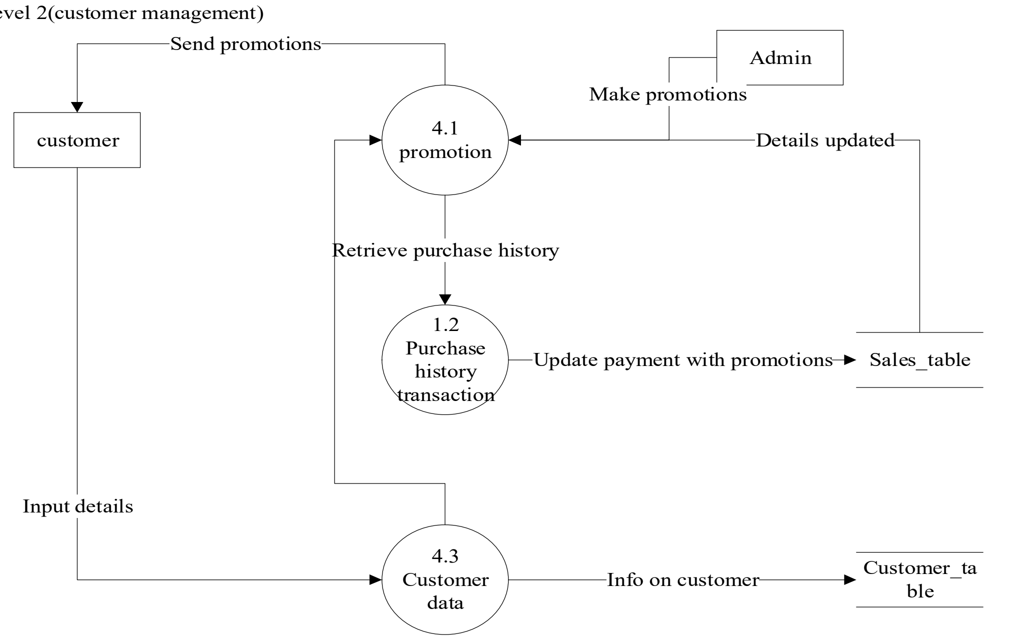


Figure 1.6.2 DFD diagram

Level 2(inventory management)

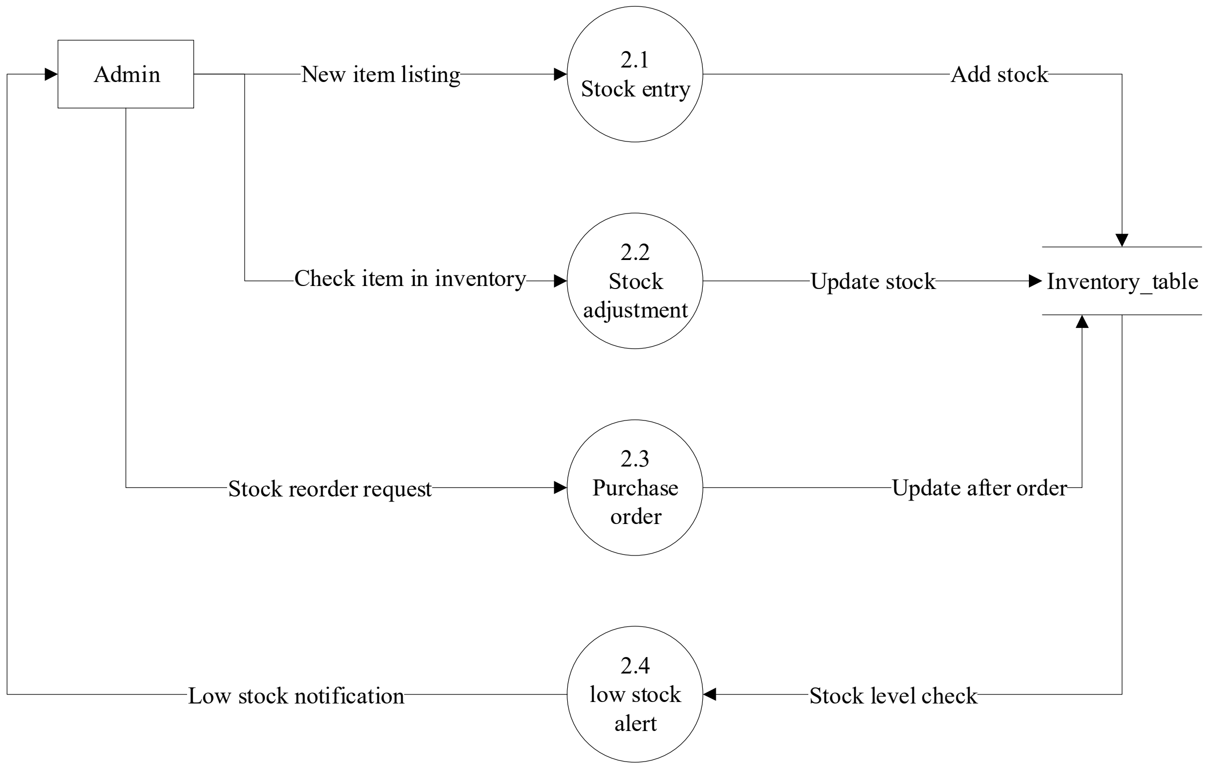


Figure 1.6.3 DFD Diagram

Diagram 7: Use Case Diagram

Use case diagrams are a common way to communicate the major functions of a software system. A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.

Use cases are nothing but the system functionalities written in an organized manner. Now another thing which is relevant to the use cases are the actors. Actors can be defined as something that interacts with the system.

So in brief, the purposes of use case diagrams can be as follows:

6. Used to gather requirements of a system.
7. Used to get an outside view of a system.
8. Identify external and internal factors influencing the system.
9. Show the interacting among the requirements are actors.

Symbols used in Use Case diagram:

⬭	Use Case	«I» --->	Include
—	Association	«E» <---	Extend
🧍	Actor	--->	Dependency
▯	System	◁—	Generalization

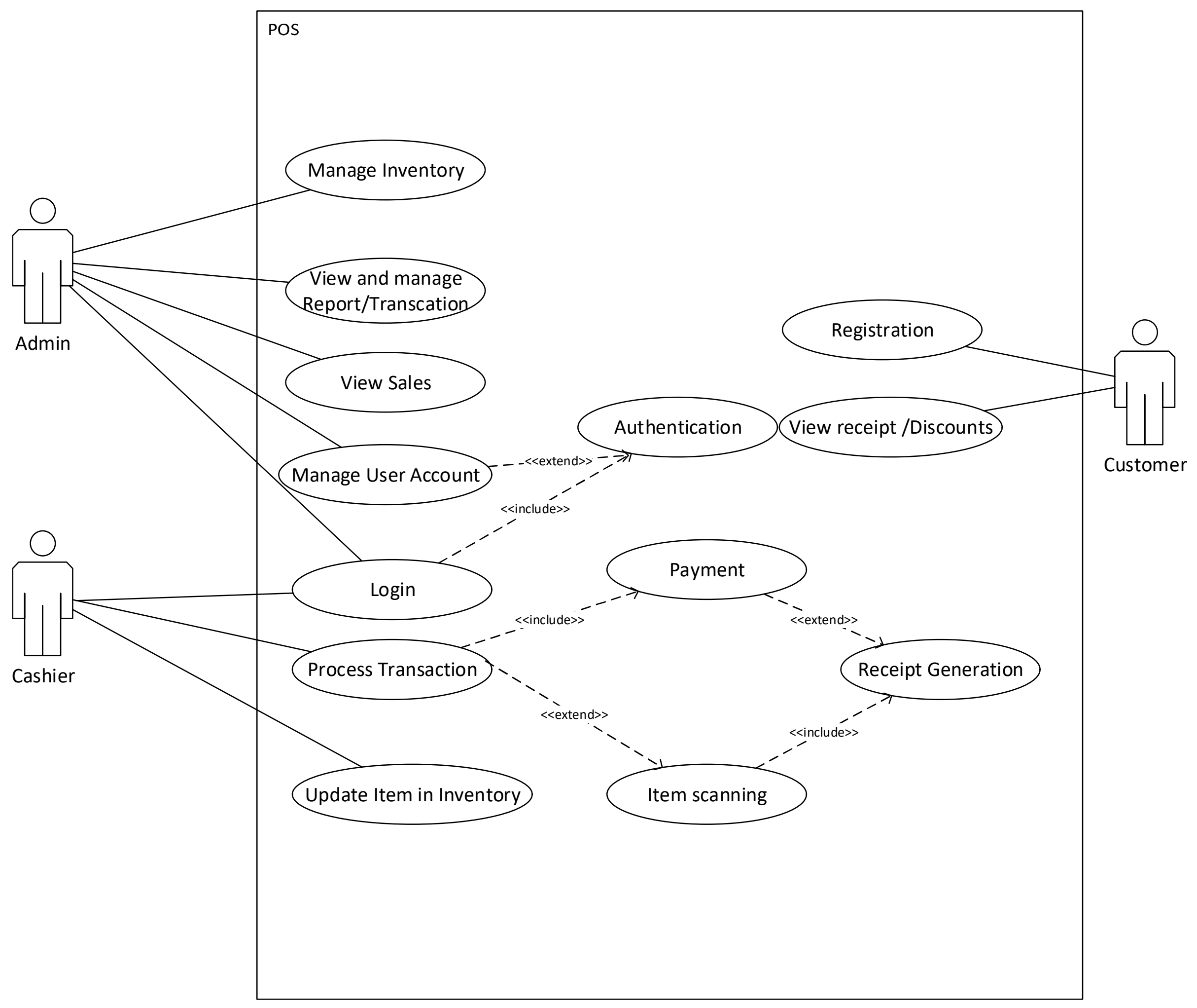
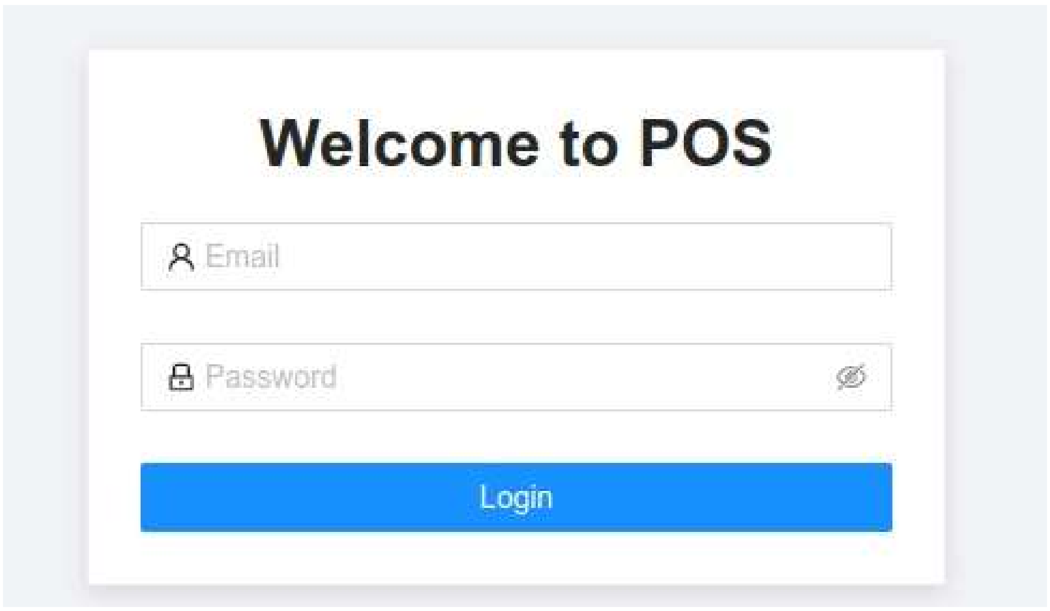


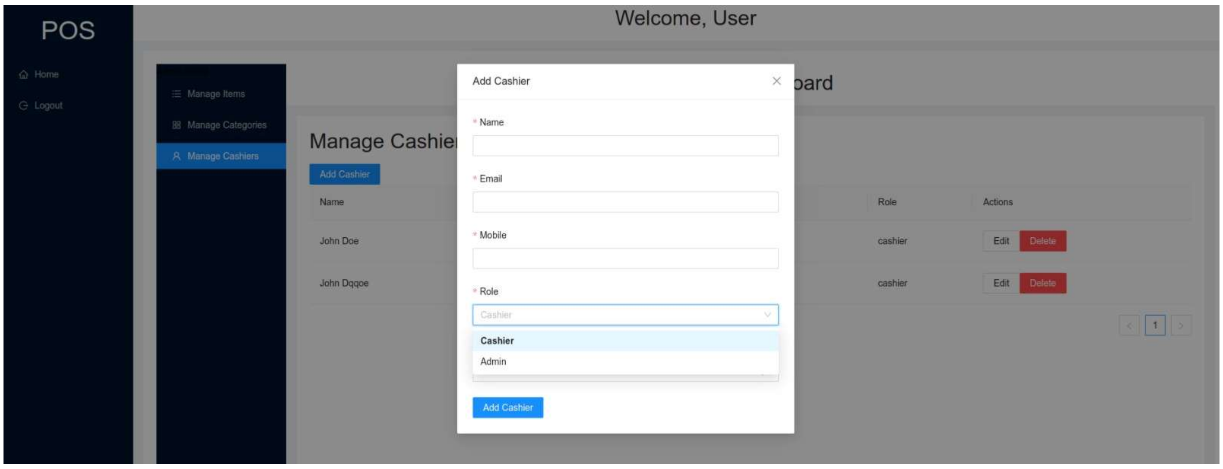
Figure 1.7 Use case Diagram

Implementation Of Project :

Login Page(Cashier/Admin)

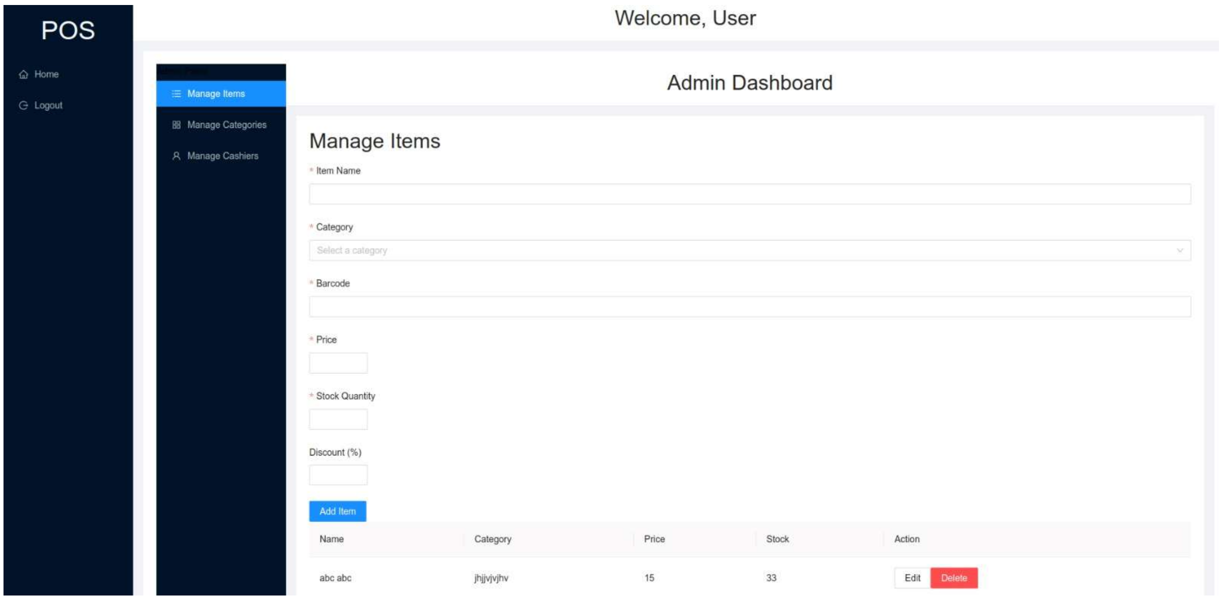


Add Cashier/Admin



Name	Role	Actions
John Doe	cashier	Edit Delete
John Dqqoe	cashier	Edit Delete

Admin Dashboard:



POS

- Home
- Logout

Welcome, User

- Manage Items
- Manage Categories
- Manage Cashiers

Admin Dashboard

Manage Items

* Item Name

* Category

Select a category

* Barcode

* Price

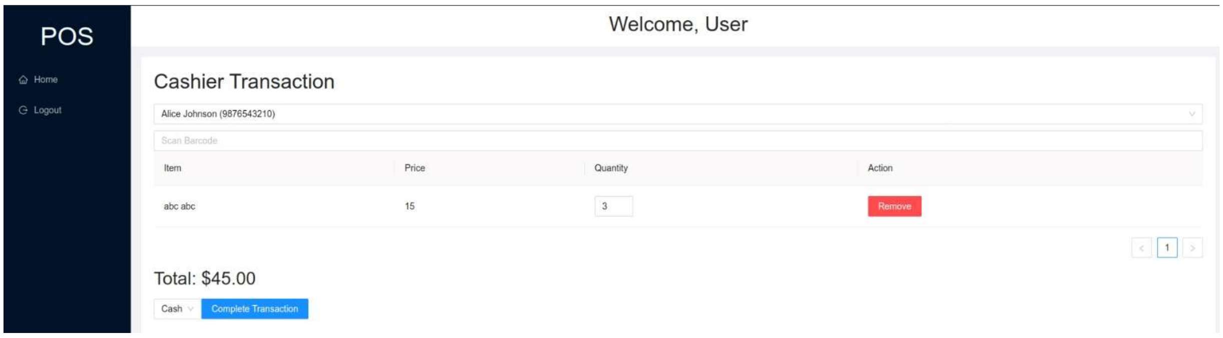
* Stock Quantity

Discount (%)

Add Item

Name	Category	Price	Stock	Action
abc abc	jhjjvjvjhv	15	33	Edit Delete

Truncation page



Welcome, User

Cashier Transaction

Alice Johnson (9876543210)

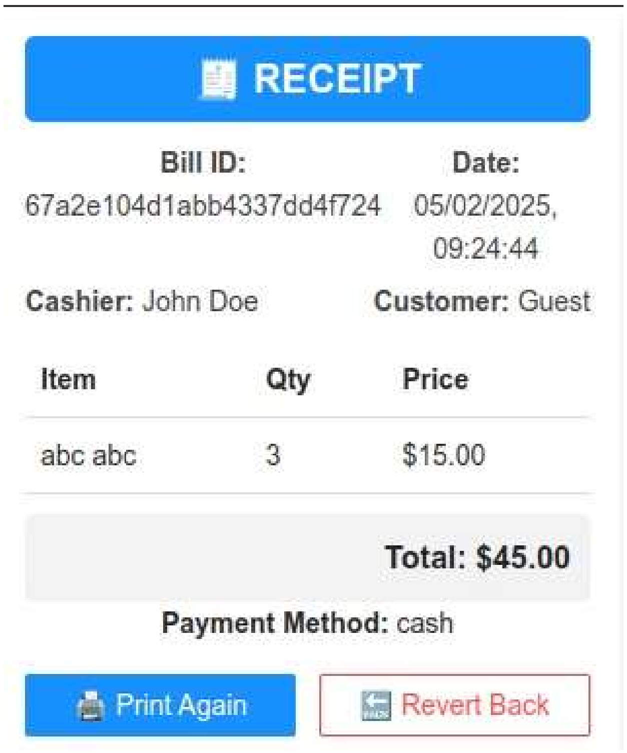
Scan Barcode

Item	Price	Quantity	Action
abc abc	15	3	Remove

Total: $45.00

Cash

Generate Receipt:



RECEIPT

Bill ID: 67a2e104d1abb4337dd4f724

Date: 05/02/2025, 09:24:44

Cashier: John Doe

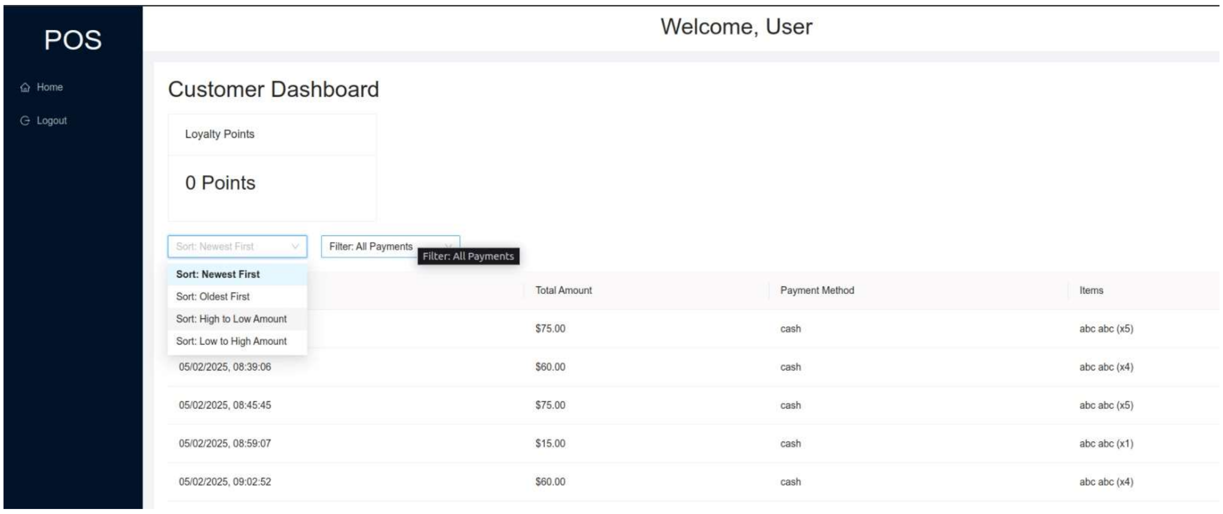
Customer: Guest

Item	Qty	Price
abc abc	3	$15.00

Total: $45.00

Payment Method: cash

Customer page(All Past Order):



POS

Home

Logout

Welcome, User

Customer Dashboard

Loyalty Points

0 Points

Sort: Newest First

Filter: All Payments

Sort: Newest First

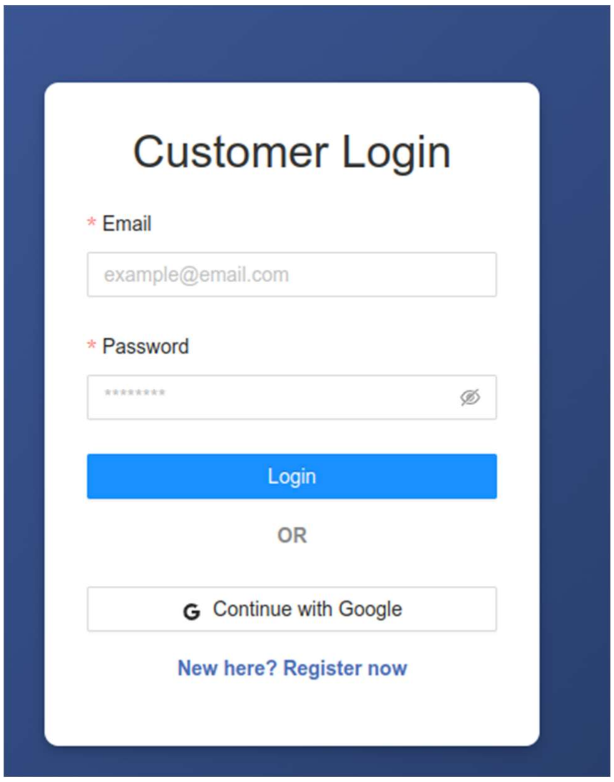
Sort: Oldest First

Sort: High to Low Amount

Sort: Low to High Amount

	Total Amount	Payment Method	Items
	$75.00	cash	abc abc (x5)
05/02/2025, 08:39:06	$60.00	cash	abc abc (x4)
05/02/2025, 08:45:45	$75.00	cash	abc abc (x5)
05/02/2025, 08:59:07	$15.00	cash	abc abc (x1)
05/02/2025, 09:02:52	$60.00	cash	abc abc (x4)

Customer Registration & Login



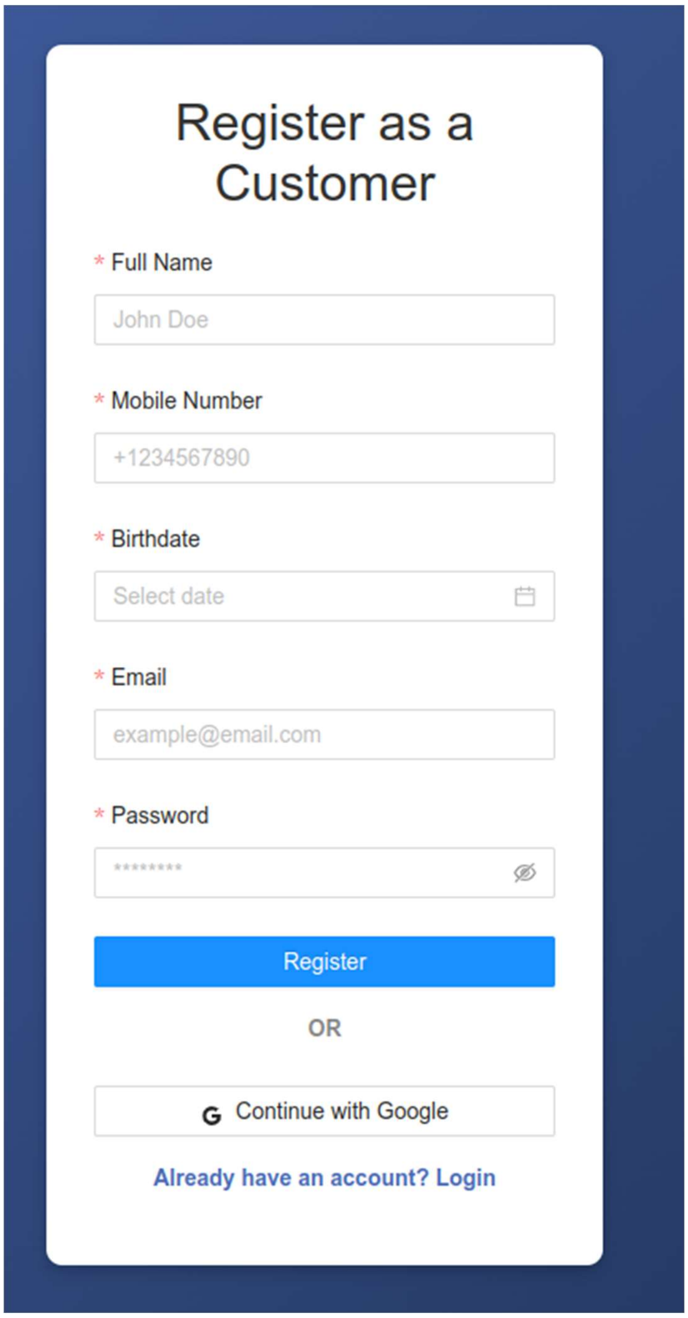
Customer Login

* Email

example@email.com

* Password

****



Register as a Customer

* Full Name

John Doe

* Mobile Number

+1234567890

* Birthdate

Select date

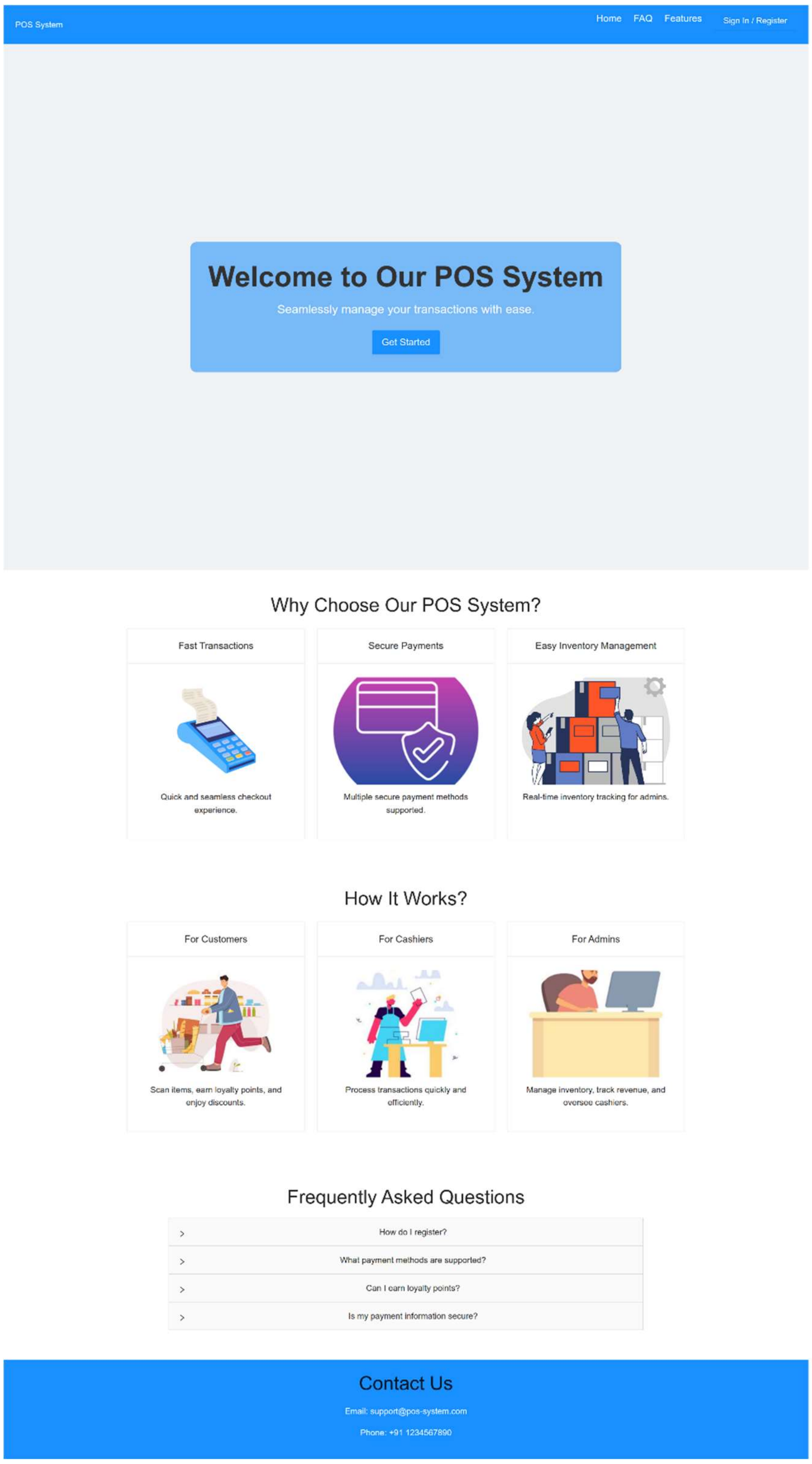
* Email

example@email.com

* Password

****

Customer Homepage :



POS System

Home FAQ Features Sign In / Register

Welcome to Our POS System

Seamlessly manage your transactions with ease.

Get Started

Why Choose Our POS System?

Fast Transactions

Quick and seamless checkout experience.

Secure Payments

Multiple secure payment methods supported.

Easy Inventory Management

Real-time inventory tracking for admins.

How It Works?

For Customers

Scan items, earn loyalty points, and enjoy discounts.

For Cashiers

Process transactions quickly and efficiently.

For Admins

Manage inventory, track revenue, and oversee cashiers.

Frequently Asked Questions

> How do I register?

> What payment methods are supported?

> Can I earn loyalty points?

> Is my payment information secure?

Contact Us

Email: support@pos-system.com

Phone: +91 1234567890