Synopsis

1 Title:

Sales and Distribution Management System

2 Statement about Problem:

Business model and industries are growing a lot these days. Many multi-national companies have a lot of shops or branches for sale of their products. Keeping a bookish record of products sold, remaining and data of employees is a very difficult. A companies expenditure increases on physical storage of data, papers on which data is stored and a specialist who stores all these data. Sometimes, the data is not 100% accurate due to human errors.

3 Why this topic?

My purpose for choosing this topic is simple. If a system is provided to a company, they can circulate that system in all branches. Physical space is not required for these systems. One system can be used for years just maintenance is required for these systems. Computer make work easy as like taking any data of any branch can easily be done by the company. However, almost all of the companies rely on a system for keeping their record rather than paper work.

4 Objective and Scope:

The system is limited for that particular company and no other company can use it. Some objectives of my system are:

i. Storing information of daily sales in a shop.

ii. Noting down the top highest selling products.

iii. Providing an easy-to-use interface.

iv. Safety of data from getting corrupt or leak.

v. Some data about employees (not sure yet).

5 Methodology:

Incremental model is the suitable methodology for this system.

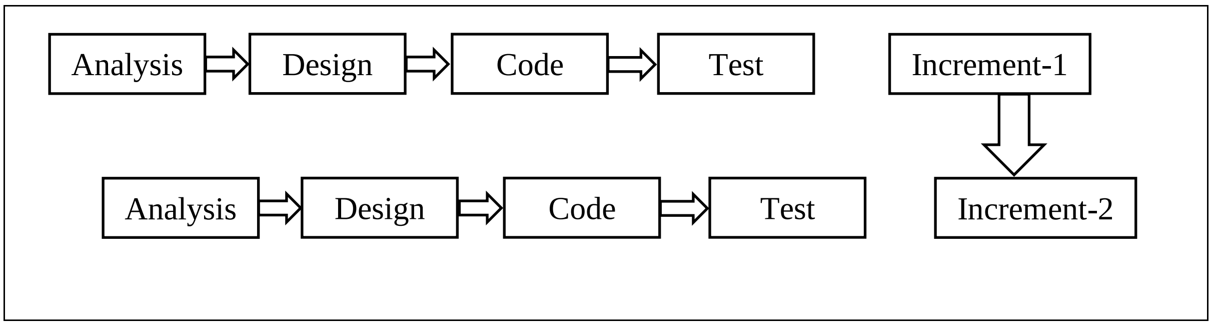


Figure 1.1 Incremental Model Diagram

6 Purpose Architecture:

The motive of the system is to take data from User Tier to Database Tier and this data can be accessed by Admin Tier.

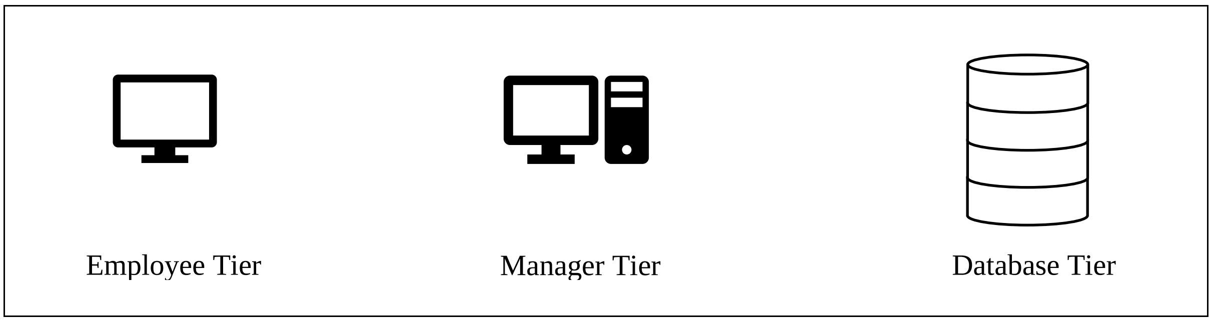


Figure 1.2 Proposed Architecture

7 Requirements:

i. Software Requirements

a. Frontend: Java API

b. Backend: Java

c. Database: MySQL

ii. Hardware Requirements:

a. Operating System: Windows

b. RAM: 2GB and more

c. ROM: 500MB free disk space required

8 Platform:

Java Development Kit

9 Contribution:

This system will contribute for the components to reduce their expenditure. Human work will get a lot less and accuracy of data storing can be improved. Large amount of data can be stores in the databases without any physical space. The server side can access any of the shops data using this system. Safety of all these data is very important which will be provided by the software. Highest selling products are displayed on the system so that company can provide some offers on that product in that area to increase the sale.

CHAPTER 1: Introduction

1.1 Background:

There are two ways through which a company gets their sales and distribution management system. Both ways have their own advantages and dis-advantages. First way is to hire an Information Technology i.e., IT Team and provide them all the necessary resources through which they can develop the system and maintain it. The second way is to sign a contract with some other company who is specialized to design this kind of software.

The way you choose for your business totally depends on what kind of business as the specialized companies carry their own large databases for your data. A small start-up company deals with comparatively lesser data so they can hire some IT people to develop a system for their company. This decision depends on certain aspects and owner of the company.

1.2 Objective:

i. Stock records of a particular branch can be displayed on one click.
ii. List of available products id displayed and sorted in number of products sold.
iii. Data about employees working for a branch is stored.
iv. Delivery of an easy to use and threat free installation.

1.3 Purpose, Scope and Availability:

1.3.1 Purpose:

This system aims towards making the business as easy as possible. The main purpose is to force a branch to rely on this system so that company will keep using it for a long time. Focus of this system is providing safety of data. No data will get corrupted or hacked from the database designed for the system.

1.3.2 Scope:

The system is limited only for Mumbai city branches for the time being. This system is only designed for storing and maintaining records. It won’t use those records for any other purpose. No automatic changes can be made by the system in database records without permission of admin. The admin is able to access data of all minor branches. Basically, each and every bit of data transfer is carried out by just few click.

1.3.3 Applicability:

This system will server as a helping hand for a growing company to grow more. Efficient data storage system leads to efficiency in business growing and customers acquisition is the market. A company can carry out different strategies to sell their product and apply some offer and discounts to be the market leader of that sector.

CHPATER 2: Survey of Technologies

1. MongoDB:

MongoDB is a No SQL database and it is a cross-platform, open-source and document-oriented database written in C++. MongoDB is an open-source document database that provides high availability, automatic scaling and high performance. MongoDB is suitable for all the modern applications which requires big data, flexible deployment, big data and the older database systems which are not competent enough. Features of MongoDB are Indexing, Replication, Duplication of Data, Supports Map Reduce tools and it is Schema-less database language.

2. ReactJS:

ReactJS is one of the most popular JavaScript front-end libraries which has a strong foundation and a large community. ReactJS is a flexible, declarative and efficient JavaScript library for building reusable User Interface (UI) components. ReactJS library is responsible only for the view layer of the application. The main objective of ReactJS is to develop User Interfaces (UI) that improves the speed of the apps. Virtual DOM (JavaScript object) helps ReactJS to improve the performance of the application. JavaScript virtual DOM is faster than the regular DOM. ReactJS can be used on both the client side and the server side and also with the other frameworks. It uses component and data patterns that improve readability and helps to maintain large apps.

3. GitHub:

GitHub is an immense platform for code hosting which supports version controlling and collaboration. GitHub helps to host the source code of project in different programming languages and keeps a track of all the changes made by the programmers. It supports version controlling and collaboration and allow developer to work on project together. GitHub provide Distributed Version Control and Source Code Management functionality of Git. Bug Tracking, Feature Requests and Task Management collaboration features are provided for every project. GitHub make is easy to contribute to the open-source projects. Collaboration, Graphical Representation of Branches, Git Repositories Hosting, Project and Team Management, Code Hosting and Track and Assign Tasks are some significant features of GitHub.

4. AngularJS:

AngularJS is an open-source JavaScript framework by Google to build web application. It can be used, changed and shared to anyone freely. It is an excellent framework for building single phase applications and line of business applications. AngularJS is designed in such a way that we test right from the start. Hence, it is easy to test any of the components through end-to-end testing and unit testing. It is perfect for Single Page Applications. It is continuously growing and expanding framework which provides better ways for developing web applications.

5. Flutter:

Flutter is a User Interface toolkit for building fast, beautiful, natively compiled applications for desktop, web and mobile with one programming language and single codebase. Flutter is open-source and free. Flutter was developed from Google and it is now managed by an ECMA standard. Flutter applications uses Dart programming language for creating an application. Developing a mobile application is a very complex and challenging task and there are many frameworks which provide excellent features to develop mobile applications. Previously, to develop and application for both Android and iOS operating system, we needed to different languages and framework but now there are several frameworks which supports both OS along with desktop applications know and cross-platform. Hence, Flutter was a new framework which was introduced by Google in the cross-platform development family. Flutter uses its own high-performance rendering engine to draw widgets. It gives easy and simple methods to start building beautiful mobile and desktop apps with rich set of material design and widgets.

Why this language?

Java:

Java is both a programming language and a platform. Java is high-level, object-oriented and secure programming language. Java has an Application Programming Interface (API) and Java Runtime Environment (JRE) hence it is a most suitable and fully functional platform to develop an application. Java is used to created Standalone (Desktop or Windows-based) Applications, Web Application, Enterprise Application and Mobile Application. As this system is a Windows-based Application and it deals with a lot of data transfer and complexity, Java is the most suitable language to create this system.

SQL and MySQL:

MySQL is a Relational Database Management System (RDBMS) which is based on the Structured Query Language (SQL). SQL is used to perform operations on records stored in the database such as inserting, updating and deleting records and creation, modification or deletion of database tables and views. This is a most efficient tool to handle a structured data. This system will mainly deal in structured data and will require the concepts of keys to work with data efficiently. Hence, SQL is the most suitable database language and MySQL is the most suitable database for this system.

Reference: javatpoint.com (23-06-2022)

CHAPTER 3: Requirement and Analysis

3.1 Problem Definition:

A company faces a lot of problems which scaling their business product al over the world and maximise profit. To maximise profit, company has to bring some new marketing strategies selling techniques and offers and discounts for the customer. For making such necessary decisions, company requires data of each and every shop of selling unit This data should be in proper manner and arranged state by state. On basis of this data, valid conclusions are drawn to scale the product in market and acquire customer acquisition.

Sub-Problems:

1. Data Safety:

In the modern technology work, the chances of data getting leaked or hacked increases. A company's data is so confidential. A company cannot afford their data to leaked and misused by some other person.

2. Booking Records:

In this modern world, a company feels it very costly to keep bookish records. The company has large number of records. Registering them one by one on book is very time consuming and effort taking. It also consists physical space. Again, there is a risk of data getting leaked in wrong hands or it is unsafe to keep these books.

3. Managing Inventory:

This is the most important factor of a branch of any company. Keeping a record of stock leaded and products sold is very crucial for a company. These factors help a company to decide prices of products and conclusion of discount. Hence, the inventory management should be accurate.

3.2 Requirement Specification:

3.2.1 Requirement Gathering:

3.2.1.1 Ways of Requirement Gathering:

i. One-on-One Interviews:

Introduce yourself and summarize the project including it scope and any timeline. Prepare some pre-determined questions to ask the interviewee.

ii. Group Interviews:

Group interviews is best for interviewees of same job position or level as they are expert in all those topics.

iii. Brainstorming:

Brainstorming helps gather many ideas as possible from as many people as possible to identify, categorize and assign tasks, opportunities and potential solutions quickly.

iv. Survey:

Survey is preparing a questionnaire which allows us to collect information from many people in a relatively short amount of time. Most of the questions are yes and no questions. Google Forms is one of the best-way to do a survey.

v. Prototyping:

In prototyping, an early version is created to show to the client on the basis of surveys and brainstorming sessions.

3.2.1.2 Technique I used for Requirement Gathering:

One-to-one Interview is the most widely used technique of requirement gathering. The clients are interviewed in this technique. Some questions are asked to clients related to system to get the need and expectation of client from the system. According to answers given by the client, analysis is done and system requirements are prepared. Before interviewing the interviewee, a basic questionnaire is prepared and asked. Later on, while interview it can be extended accordingly. The ways to prepare a perfect interview are:

i. Asking Open Ended Questions:

Asking more Yes and No type questions is always better in an interview. A large number of questions are solved in minimal time.

ii. Pin Point Details:

People answers questions differently. So, taking only necessary information from the answers and ignoring unnecessary part is very important.

iii. Interview Right People:

Choosing a right person for interview is very important. A person who has knowledge about software system and business. A person who has clear idea about what type of system to be created.

Questions and Answers:

1. What is the need of a Sales and Distribution Management System.
   - ➢ Now-a-days, due to Information Technology, major part of business is reliable on technology. Technology makes it very easy to store data records. Hence, it is important.
2. What things are focused while developing a Sales and Distribution Management System?
   - ➢ First of all, this type of system requires a strong, huge and secured database. The system should be performance efficient and quick. As the transactions are happening every minute it is very important to carry forward them as quickly as possible.
3. Do you think the display graphs of monthly sales and all different factor would help to grow business?
   - ➢ Yes, it will help. The statistics show an accurate measure through which future strategies and offers can be made. Statistics have a huge importance in scaling of business.

4. If the system suggests or changes price of a particular product according to its demand how flexible it would be?

- The price of product cannot be changed by the system as it is fixed by the company by calculating production charges and minimal profit margin. Change in price task can only be carried out by the company and not by system.

5. How many data space will be required for such kind of systems?

- Depends on what kind of business you are in. If the business has wide range of products available with huge customer market, then the database should be with more storage capacity.

6. What kind of interface should be created for such system?

- This king of systems should have clean and easy to use interface. The sales officer should not feel it very boring and irritating to use as it will affect his/her selling ability. The system is used continuously during shop open hours so it should have and interface which attracts and motivates sales officer to work more.

3.2.2 Requirement Analysis:

3.2.2.1 Functional Requirements:

i. The system provides a database which will save data of all shops through the system. The data can be modified according to buying and selling of products.

ii. The system has an option were, new product can be added or existing one can be removed from the interface.

iii. Number of quantities of a product will change according to the sell of product an it can be updated if stock increases.

iv. The system will show list of products available in the shop. That list can be sorted by most selling product and less selling products. It will even display the number of units sold.

v. The system will also display graphs and charts of the monthly sales of all the shops.

vi. The system will have a use panel and an admin panel. User panel will
have limited functionality and access to database. The admin panel can access the database and it can even keep a watch on the use panel.

3.2.2.2 Non-Functional Requirements:

i. Security:

A safe and secure database server will be provided to the system. The queries will be fired so efficiently that the data will get stored in proper structured format. No database can be accessed by the user side model. Data cannot be hacked by anyone or it won’t get corrupted.

ii. Availability:

The system will be available to user all the time. The system will not get down automatically. For going under maintenance, the system will go down only under the permission of that branch manager. If in case, the system fails, it can be recovered easily and quickly.

iii. Capacity:

As the main objective of the system is to store data, capacity of the database will be at a greater extent. Multiple data products can be stored in the database.

iv. Usability:

The system can be used by a company who has their own multiple branches. The admin panel will be the main panel and user panel will be distributed around all the branches of the company.

v. Data Integrity:

The function of data integrity is provided by the system. The data will be stores in data in a very structured manner by using proper attributed. The data in this system will be modifying continuously as the products sold and new stocks are loaded from all branches.

3.2.2.3 System Requirements:

i. Data Communication (sending and retrieving data):

Function: To communicate data which is main functionality.

Description: Communicates data between admin and database according to requirement of admin. Update of data as quickly as possible.

Inputs: Products sold from user side and stock loaded of user's side.

Source: The user side and admin side or database.

Outputs: Displays information from database in structured manner.

Destination: Bi-directional. Database and User-Admin side.

Action: When the data is entered by the use or amin side when new stock is loaded, the data is taken as input by user and respective query is fired to store the data into the database in this case, database is the destination and user or admin side is source. If any product is sold again same process is followed. When data is requested from the database, database becomes the source and the user or admin panel becomes the destination.

Requirements: In case of receiving data, a query must be fired by the admin side. To enter data into database data is required to entered by the admin or user side.

Pre-Condition: Database server should be proper.

Post-Condition: Displaying to the user that data is entered or retrieved successfully.

Side-effects: Probability of error in data.

ii. Admin Login:

Function: Provide credentials for admin to login.

Description: Admin login will be a step for admin of the company to get the access of database and all other users. An admin should be secured properly as it can access everything.

Inputs: Login credentials from admin.

Source: Admin creating his login credentials.

Output: Provide user id and password to admin which should be kept safe by him.

Destination: Admin will receive his credentials and also the data will be stored in the database.

Action: The admin has to register his/her information and create login id and a strong password. This registration must be authorized by the company or higher authority. After everything being done perfectly, admin can now login and access everything from the system.

Requirements: Some personal information from user.

Pre-Condition: Person trying to register should be an employee of the company.

Post-Condition: Admin will get successful access to the database.

iii. Creating User Login by Admin:

Function: Provides interface to the employees of a company.

Description: The user (employee) can't create his/her own login credentials as user. This will be created by the admin and provided to the users.

Inputs: Data of user in minimal amount.

Source: Admin

Output: User account will be created.

Destination: Login credentials will be sent to the user and database will store the data.

Action: The admin will enter all details of the employee according to a unique identifier is generated by the system along with a strong password which will be provided to the user.

Requirements: Certain information of user (employee).

Pre-Condition: The user should be an employee of the company and authorized by the company.

Post-Condition: User can now login and get access of some certain features of application or system.

3.3. Planning and Scheduling:

3.3.1 Activity Table:

Activity No.	Chapter No.	Chapter Name	Start Date	End Date
A1	0	Synopsis	15-06-2022	18-06-2022
A2	1	Introduction	20-06-2022	25-06-2022
	2	Survey of Technologies		
A3	3	Requirements and Analysis	27-06-2022	02-07-2022
	3.1	Problem Definition		
A4	3.2	Requirement Specification	04-07-2022	09-07-2022
	3.2.1	Requirement Gathering		
	3.2.2	Requirement Analysis		
A5	3.2.3	System Requirements	11-07-2022	16-07-2022
A6	3.3	Planning and Scheduling	18-07-2022	23-07-2022
	3.4	Software and Hardware Requirements		
	3.5	Conceptual Models		
	3.5.1	Entity-Relationship Diagram		
A7	3.5.2	Schema Diagram	25-07-2022	29-08-2022
A8	3.5.3	Data Flow Diagram	01-08-2022	20-08-2022
A9	3.5.4	Class Diagram	22-08-2022	27-08-2022
A10	3.5.5	Use Case Diagram	29-08-2022	10-09-2022
A11	3.5.6	Sequence Diagram	12-09-2022	14-09-2022
A12	3.5.7	Activity Diagram	14-09-2022	17-09-2022
	3.5.8	State Diagram		

Table 3.1 Activity Table

3.3.2 Gantt Chart:

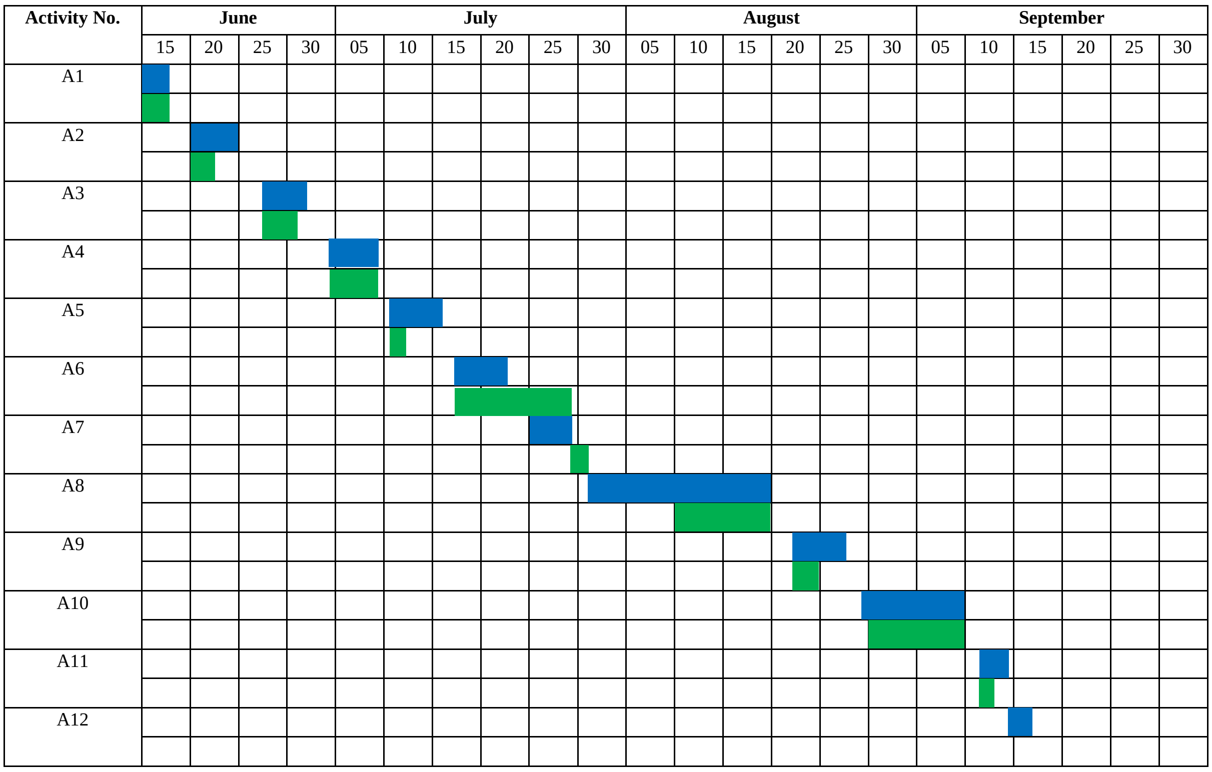


Figure 3.1 Gantt Chart

3.4 Software and Hardware Requirements:

3.4.1 Software Requirements:

Frontend: Java API

Backend: Java

Database: MySQL

3.4.2 Hardware Requirements:

Operating System: Windows

RAM: 2GB or more

ROM: 500MB of disk space available

Processor: Intel Duo+ or Ryzen 3+

3.5 Conceptual Models:

3.5.1 Entity-Relationship Diagram:

An Entity-Relationship Diagram perceives the real world and consist of entities and relationships as objects. Entity-Relationship Diagram is used to design and illustrate relational database in Software Engineering.

An entity is a thing or an object in the real world that is distinguishable from all other objects. An entity set is a set of entities of the same type that share the same properties. An entity set is represented by a set of attributes which are descriptive properties possessed by each member of an entity set. Each entity has a value for each of its attribute.

A Relationship is an association among several entities. A relationship set is a set of relationships of the same type.

The above table show the symbols used in an Entity-Relationship Diagram along with its description.

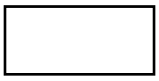
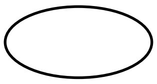
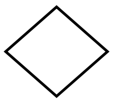
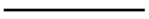
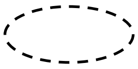
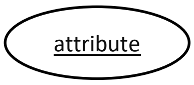
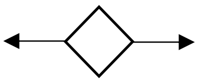
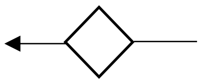
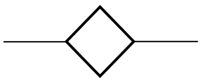
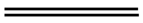
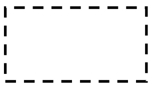
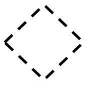
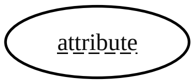
Name	Symbol	Description
Rectangle		A rectangle represents an entity set
Ellipse		An ellipse represents attributes of an entity set
Diamond		A diamond represents a relationship set
Line		A line links attributes to entity sets and entity sets to relationship sets.
Dotted Ellipse		A dotted ellipse represents a derived attribute
Underline		An underline to attribute name represents a primary key which is a unique identifier
One-to-one		It represents a one-to-one relationship between entities
One-to-many		It represents a one-to-many relationship between entities
Many-to-many		It represents a many-to-many relationship between entities
Double Line		Double line represents the total participation
Dotted Rectangle		A dotted rectangle is used to describe a weak entity set
Dotted Diamond		A dotted diamond is used to describe a weak relationship set
Dotted Underline		A dotted underline is used to represent discriminating attribute for a weak entity set

Table 3.2 Entity-Relationship Diagram Symbols

Reference: Silberschatz−Korth−Sudarshan: Database System Concepts, Fourth Edition

3.5.1.1 Entity Sets:

1. manager_login set:

The manager_login set stored login credentials of all the managers available. The attributes of manager_login set are managername and m_password. Primary key for this set is managername.

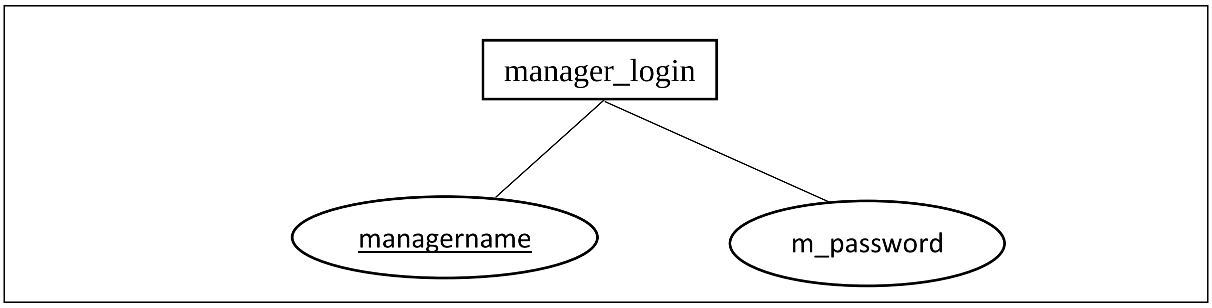


Figure 3.2 manager_login set

2. manager set:

The manager entity set is used to store all data of a manager working in the shop. This set will help manager to login to its dashboard. The manager set consist of attributes manager_id, m_phone, m_name, m_email, m_dob and m_age. Primary key for this set is manager_id.

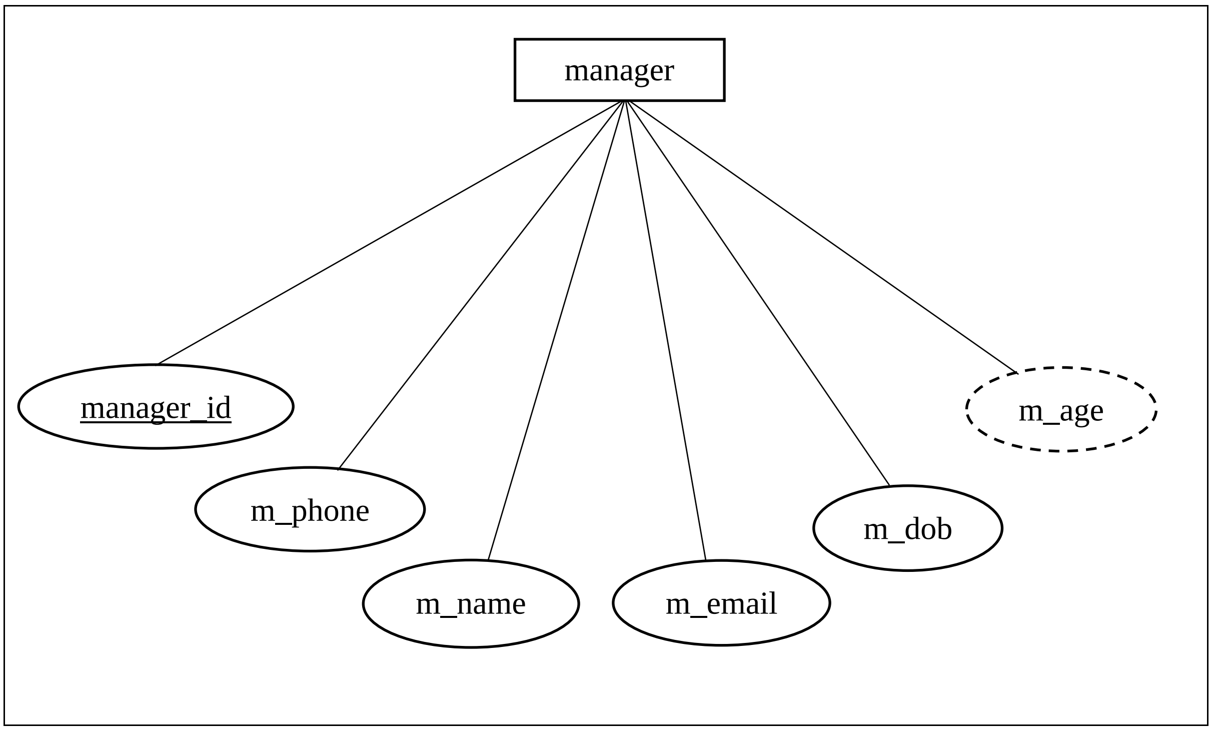


Figure 3.3 manager set

3. employee_login set:

The employee_login entity set is used to store login credentials of employees working in the shop. employeename and password are the two attributes used by this set where emplooyeename is the primary key.

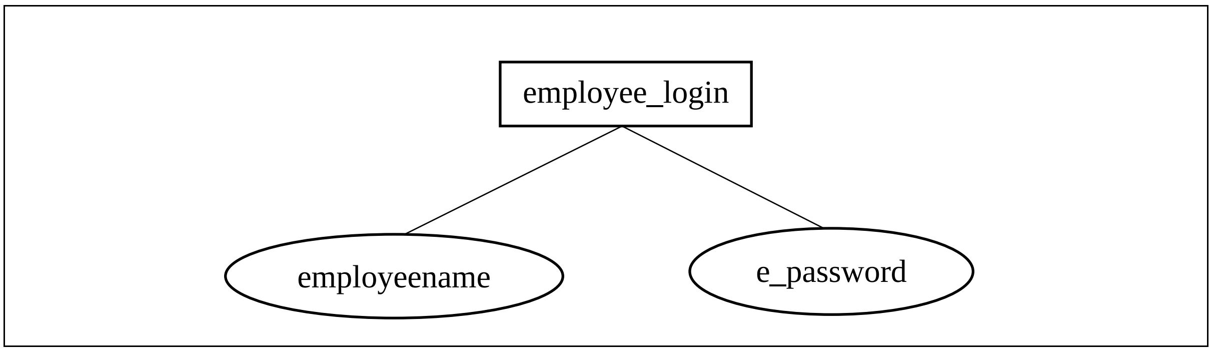


Figure 3.4 employee_login set

4. employee set:

The employee entity set is used to store details of emplyees currently working in the company. This entity set will help employee to login to the employee dashboard and use the features of system for employee. This entity set consists attributes employee_id, e_name, e_phone, e_email, e_dob, e_age where employee_id is the primary key.

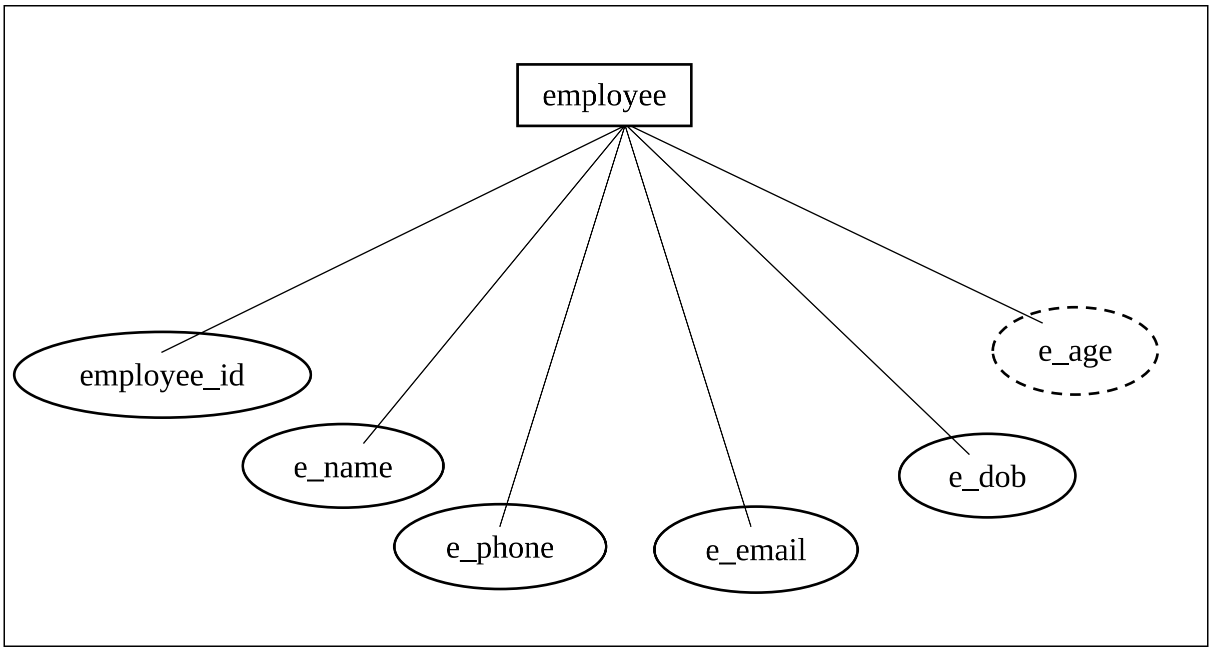


Figure 3.5 employee set

5. product set:

The product entity set is used to save details of products in the company. This entity set will help system to carry out all operations related to products of the company. The attributes of product set will be product_id, product_name, product_price and description. The product_id is a primary key and the product_price is divided into selling_price and cost_price.

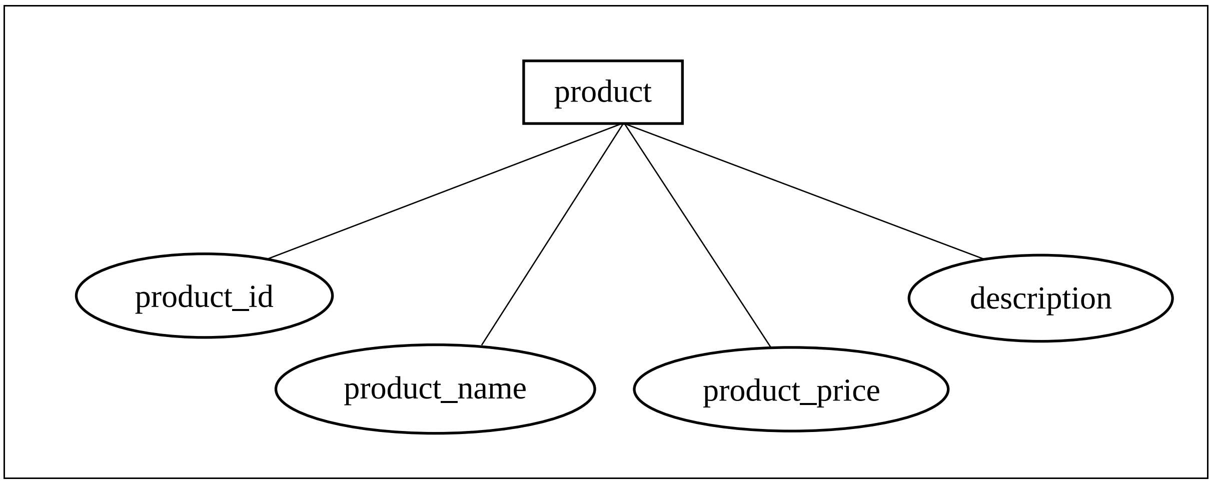


Figure 3.6 product set

6. shop set:

The shops set is used to store the data of all shops available under the company. The attributes of shop entity set are shop_id, shop_name, address, city, state, pincode. The shop_id attribute will be the primary key of this set.

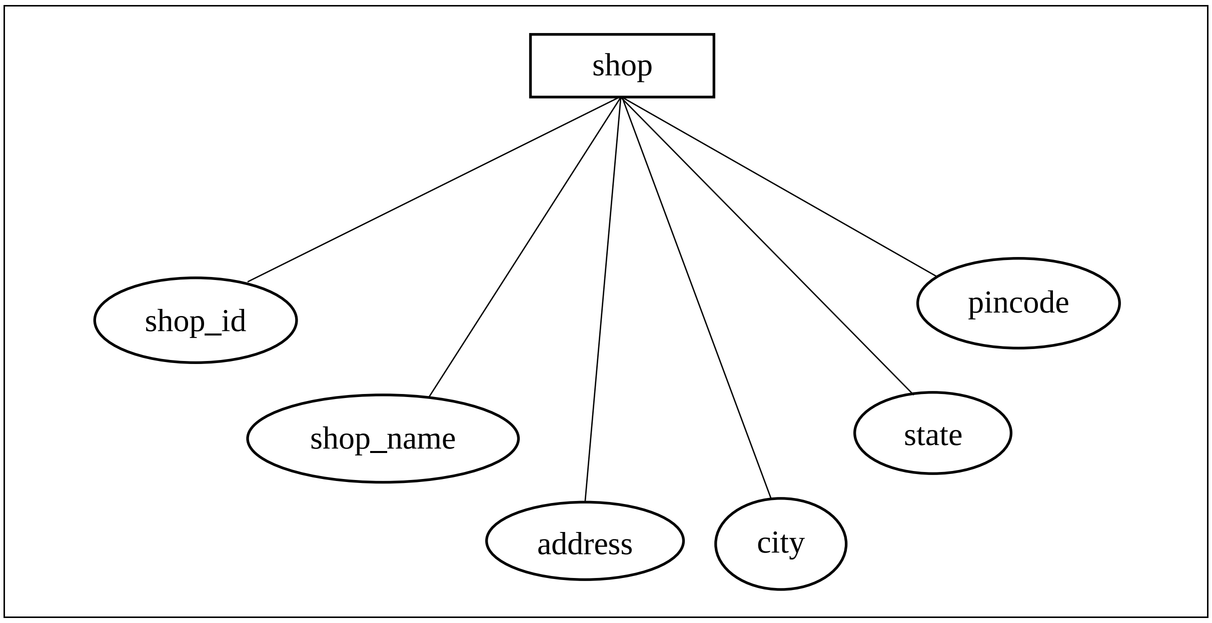


Figure 3.7 shop set

7. sales set:

The sales set is used to store the sales of daily data of a product from a shop at what price and quantities in accordance with date. The attributes of this sales entity set are order_id, date, selling price and quantity.

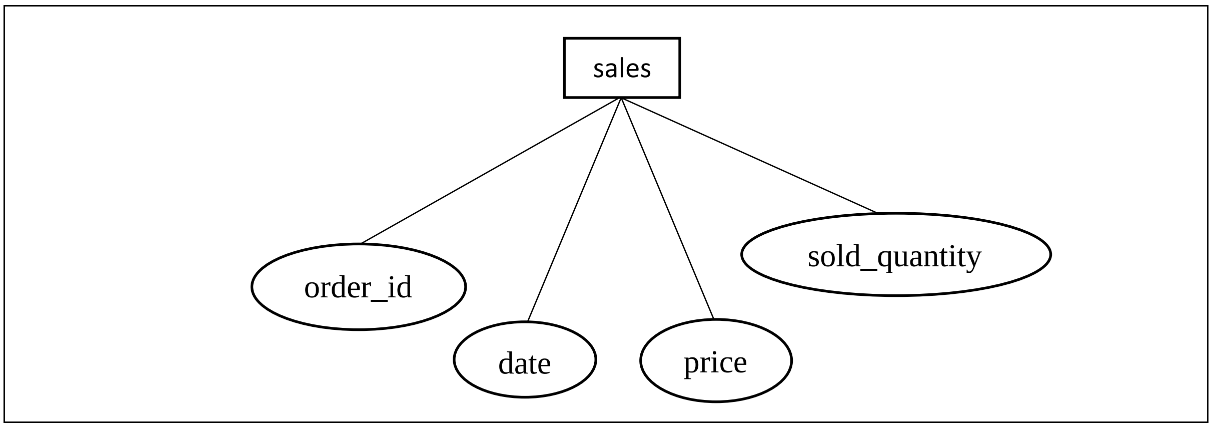


Figure 3.8 sales set

8. stock set:

The stock entity set is a weak entity set which will be used to store the data of stock available for that particular product in a shop. The attributes for this entity set are last_updated_date, quantity. last_updated_date is the discriminating attribute for this weak entity set.

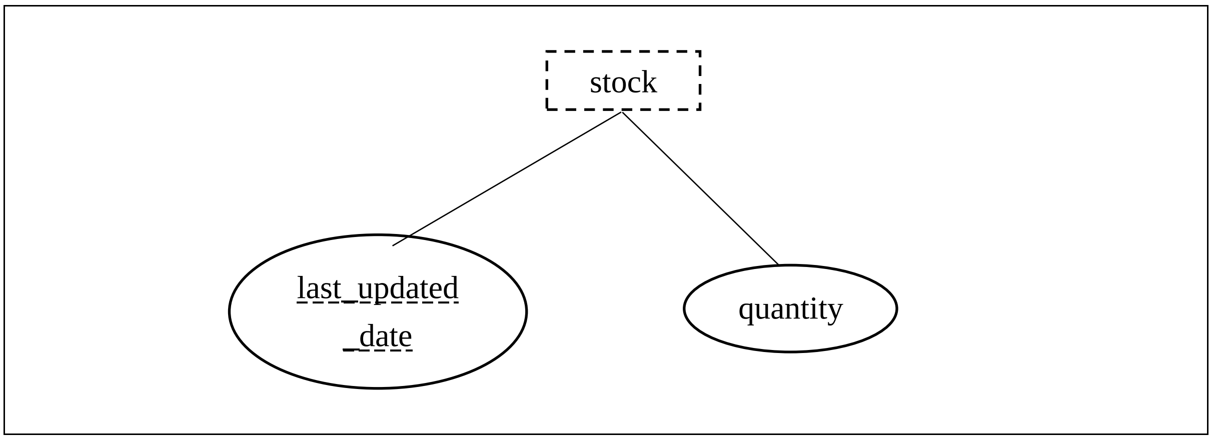


Figure 3.9 stock set

3.5.1.2 Relationship Sets:

1. manager_login directs to manager:

This relationship will help to create a path for manager to login to go this his own account. The login credentials of manager are stored different from the its personal information. manager_login set will help the system to verify the manager and give him access to manager set.

Participation: Total Participation

Mapping Cardinality: One-to-One Relationship

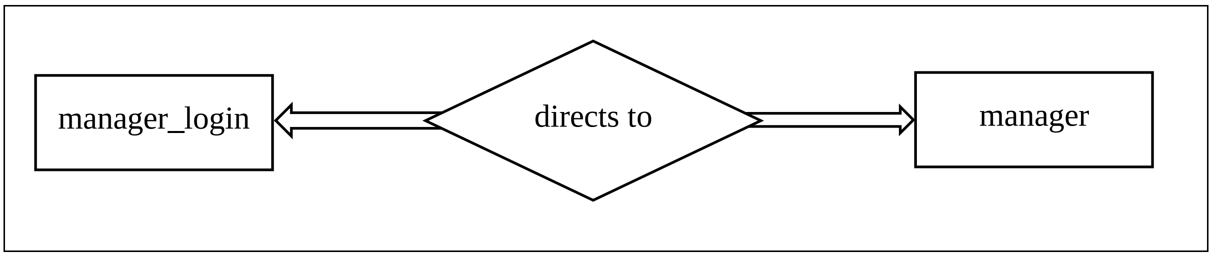


Figure 3.10 manager_login directs to manager set

2. employee_login directs to employee:

This relationship will help to create a path for employees to login to his account and start working with the features of this system.

Participation: Total Participation

Mapping Cardinality: One-to-One Relationship

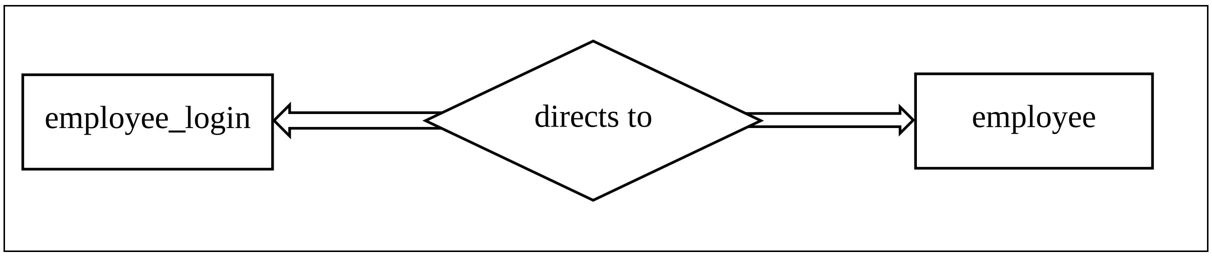


Figure 3.11 employee_login directs to employee set

3. manager operates employee:

The manager can create a new employee and it can be added to the system. The manager will enter details of the employee and create an account for employee to access features of system.

Participation: Total Participation

Mapping Cardinality: One-to-Many Relationship

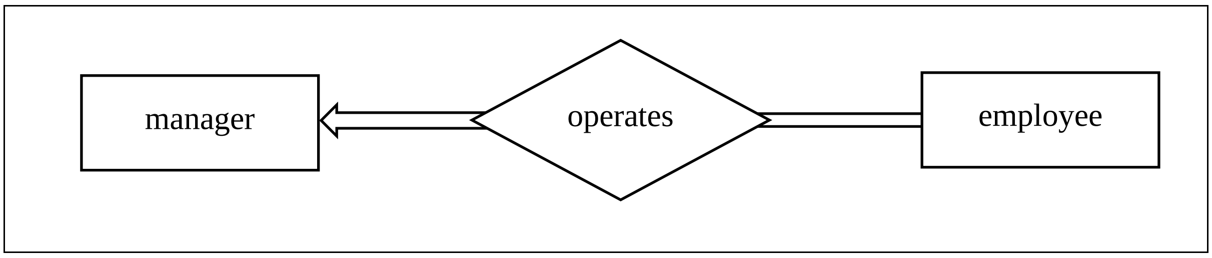


Figure 3.12 manager operates employee set

4. manager operates product:

If a new product is introduced by the company the manager will create that product and add it to the database. This relationship will help manager to add a product and carry out operations on it.

Participation: Total Participation

Mapping Cardinality: Many-to-Many Relationship

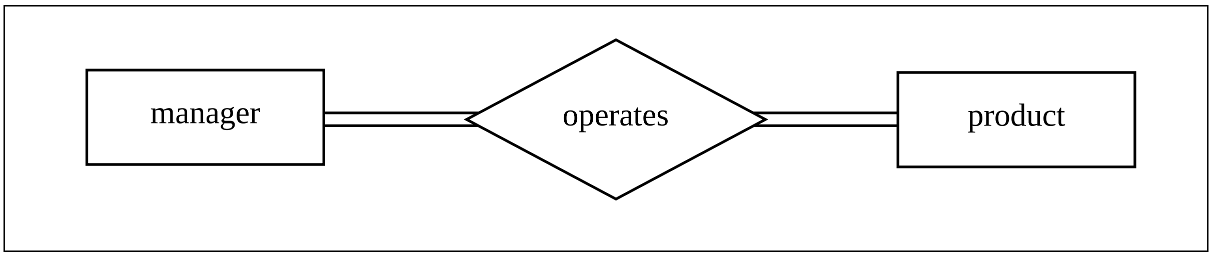


Figure 3.13 manager operates product set

5. manager operates shop:

If a new shop is purchased by the company the manager can add that shop to the system with the help of this relationship.

Participation: Total Participation

Mapping Cardinality: Many-to-Many Relationship

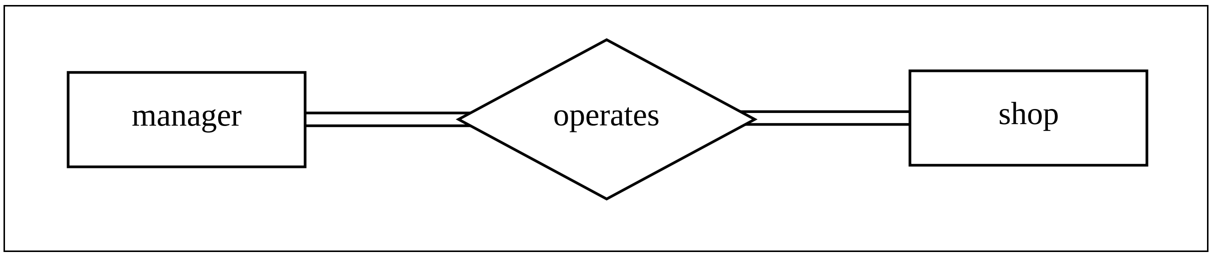


Figure 3.14 manager operates shop set

6. stock of product:

The stock entity set is a weak entity set and it don't have any primary key. The stock stored in this entity set is of a product. So, the primary key of product entity set i.e., product_id will be used to complete this weak entity set.

Participation: Total Participation

Mapping Cardinality: Many-to-One Relationship

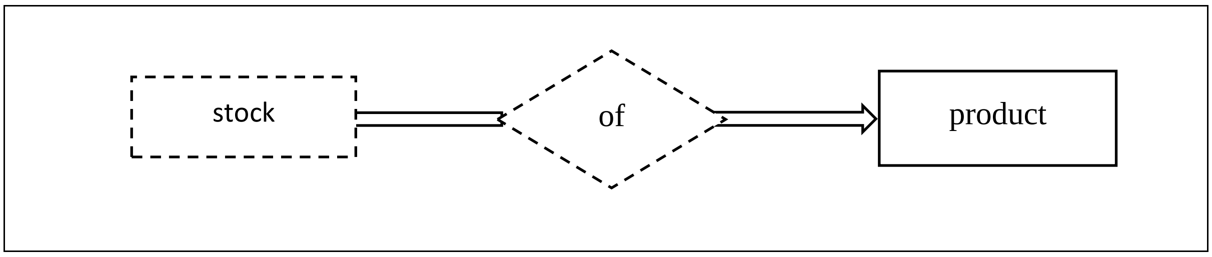


Figure 3.15 stock of product

7. stock in shop:

This relationship will show the stock of product available for a particular shop.

Participation: Partial Participation

Mapping Cardinality: Many-to-Many Relationship

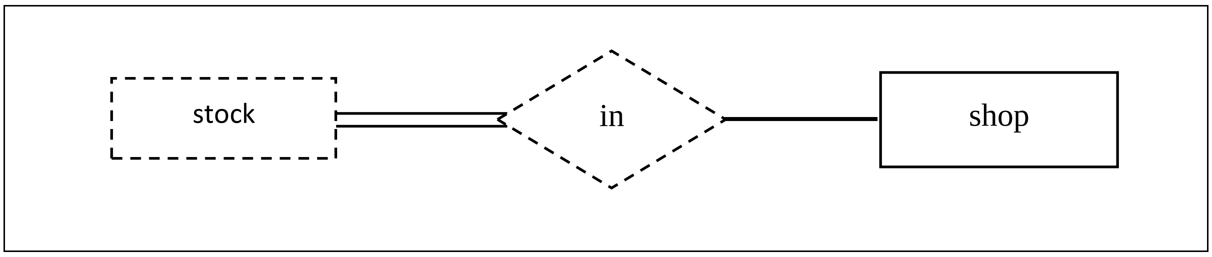


Figure 3.16 stock in shop

8. employee update sales:

In this relationship, the employee will keep updating sales of products accordingly and the employee doing modification will be noted down.

Participation: Total Participation

Mapping Cardinality: One-to-Many Relationship

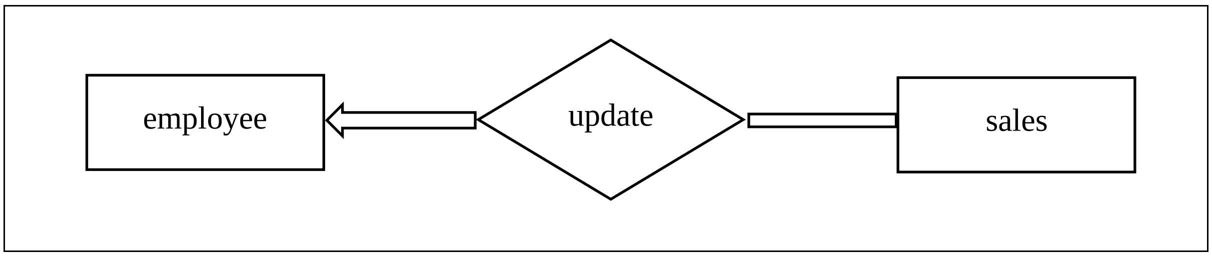


Figure 3.17 employee update sales set

9. manager updates stock:

This relationship will help manager keep updating stock of the products once they are added in the inventory.

Participation: Total Participation

Mapping Cardinality: Many-to-One Relationship

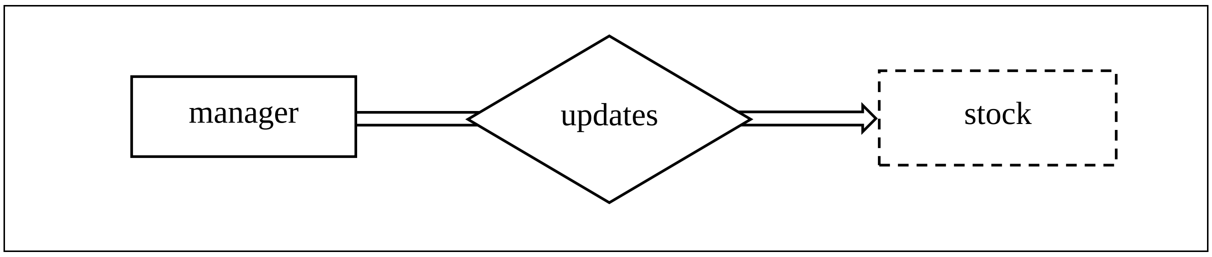


Figure 3.18 manager updates stock set

3.5.1.3 Diagram:

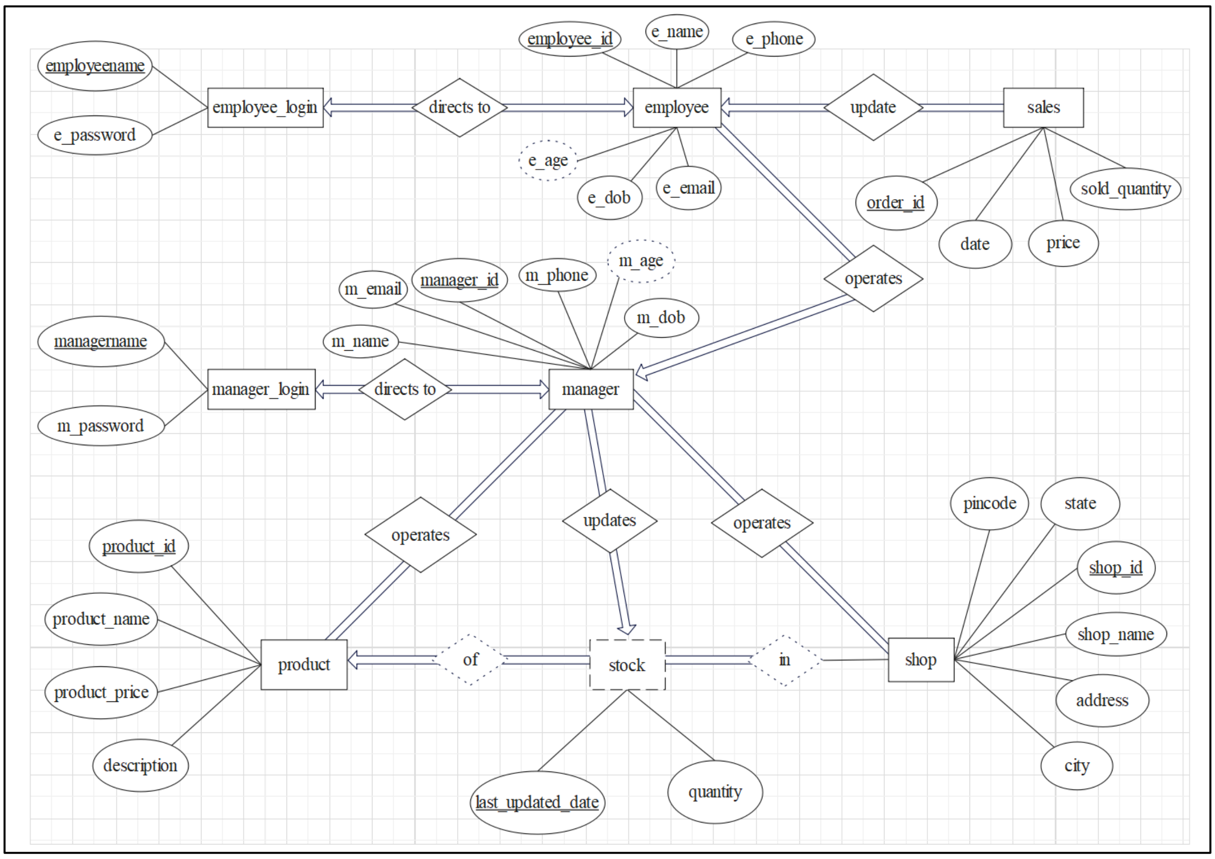


Figure 3.19 Entity-Relationship Diagram

3.5.3 Schema Diagram:

A database schema along with its primary key and foreign key dependencies can be depicted pictorially by a Schema Diagram. In a Schema Diagram, each relation appears as a box with attributes listed inside it and the relation-name above it. If there are primary key attributes, a horizontal line crosses the box with the primary key attributes listed about the line.

Foreign key dependencies appear as arrows from the foreign key attributes of the referencing relation to the primary key of the referenced relation. The difference between Entity-Relationship Diagram and Schema Diagram is that Entity-Relationship Diagram do not show foreign key attributes explicitly, whereas Schema Diagram show them explicitly.

The above symbols are used to design a Schema Diagram for Database Schema:

Name	**Symbol**	**Description**
Table	Attribute1 Attribute 2	The table is used to list all the attributes of an entity set
Primary Key	**PK** **Attribute**	A primary key is shown in the above way
Foreign Key	**FK** **Attribute**	A foreign key is shown in the above way along with relationship arrow
One-to-One	┼──┼	This symbol is used to define One-to-One relationship
One-to-Many	┼──<	This symbol is used to define One-to-Many relationship
Many-to-Many	>──<	This symbol is used to define Many-to-Many relationship

Table 3.3 Schema Diagram Symbols

Diagram:

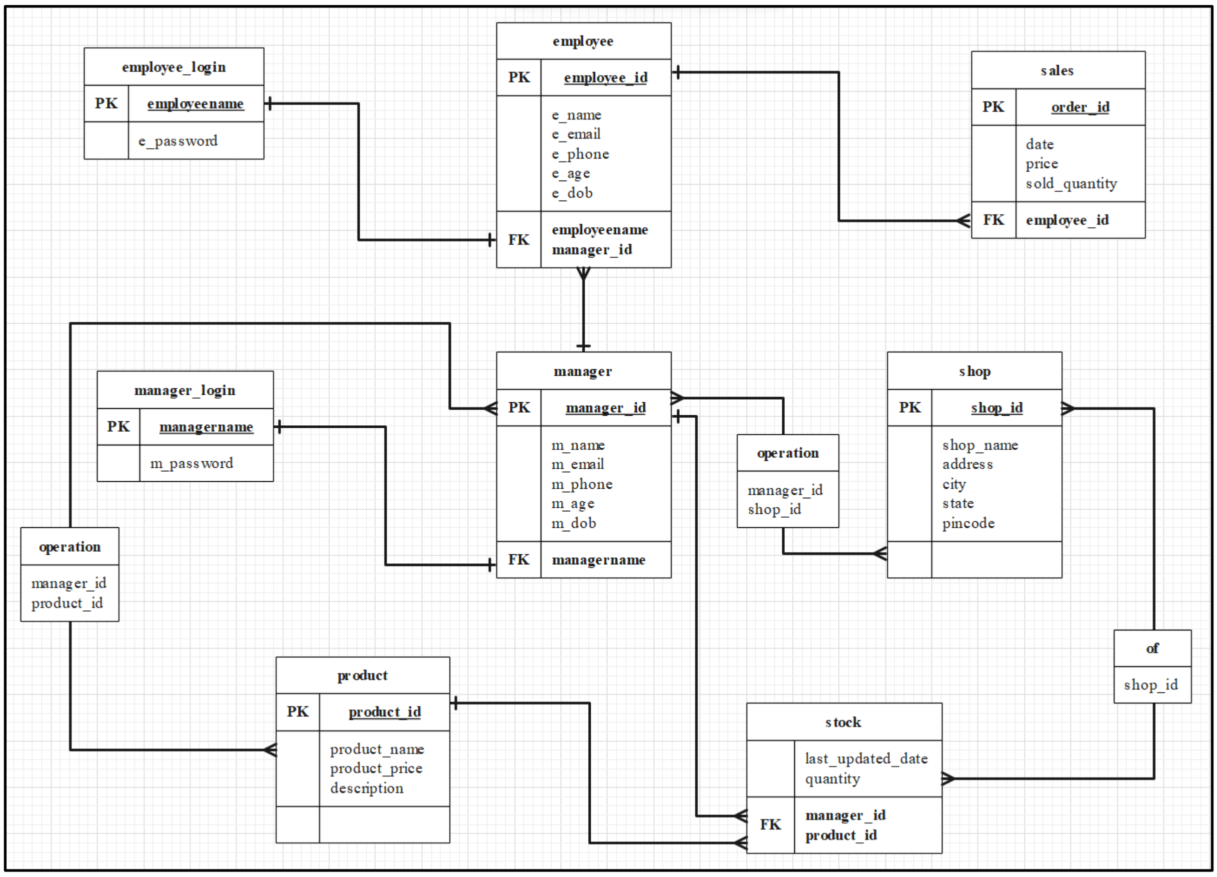


Figure 3.20 Schema Diagram

3.5.4 Data Flow Diagram:

A Data Flow Diagram maps out the flow of information for any system or process. It uses symbols like circles, rectangles, arrows with short text label, storage points and the routes between each destination. Data Flow Diagrams range form simple i.e., hand-drawn process overviews to in-depth multi-level Data Flow Diagrams that dig progressively deeper into how the data is handled. They can be used to analyse an existing system or model a new one.

Data Flow Diagram works properly for the real-time or data-oriented software and systems. The symbols used to draw a Data Flow Diagram are:

Name	**Symbol**	**Description**
Rectangle	▭	A recta1gle is used to show an external entity
Rounded Corner Rectangle	▢	A rounded corner rectangle is used to define process of data flow
Data Store	◫	This symbol is used to define the data entity storage
Line	→	The line symbol describes flow of data

Table 3.4 Data Flow Diagram Symbols

Diagram:

1. Context Level:

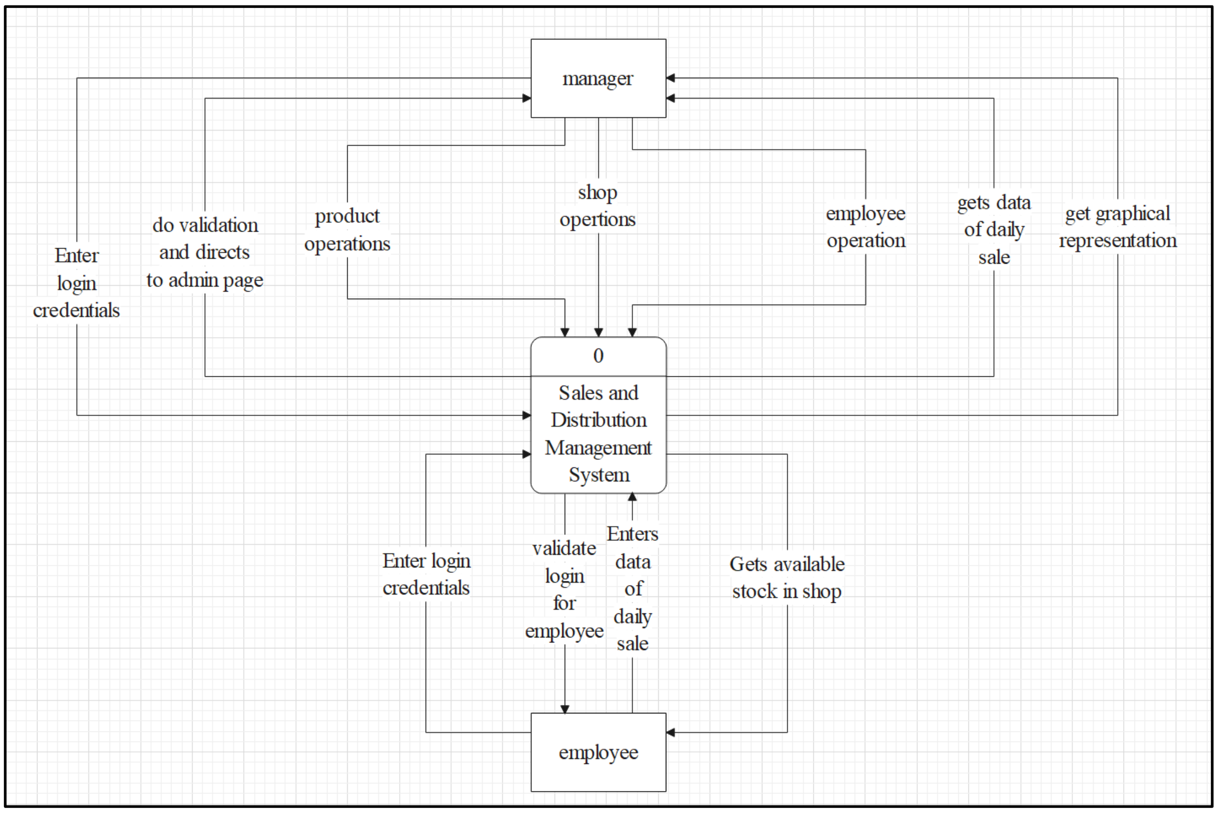

manager
Enter login credentials
do validation and directs to admin page
product operations
shop opertions
employee operation
gets data of daily sale
get graphical representation
0
Sales and Distribution Management System
Enter login credentials
validate login for employee
Enters data of daily sale
Gets available stock in shop
employee

Fig 3.21 Level 0 Data Flow Diagram

2. Level 1 Diagram:

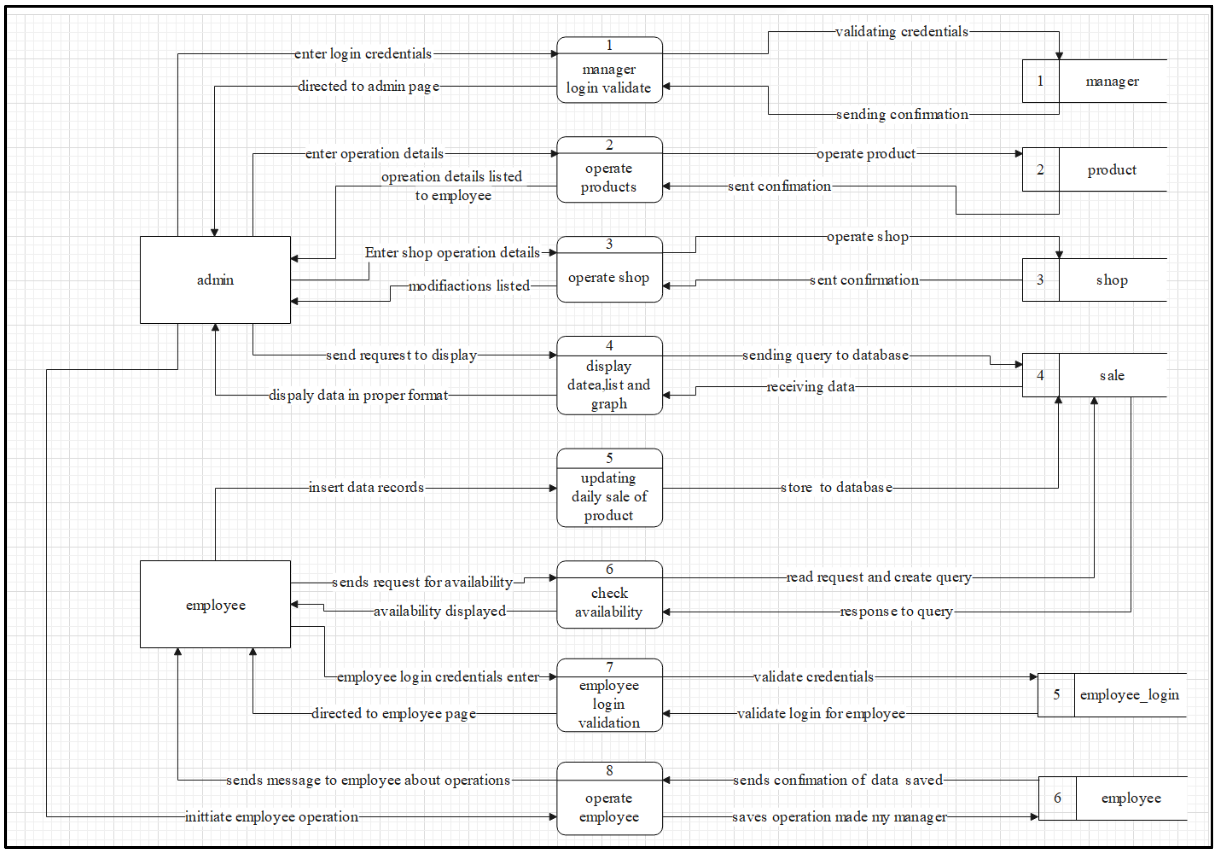


Fig 3.22 Level 1 Data Flow Diagram

3. Level 2 Diagram:

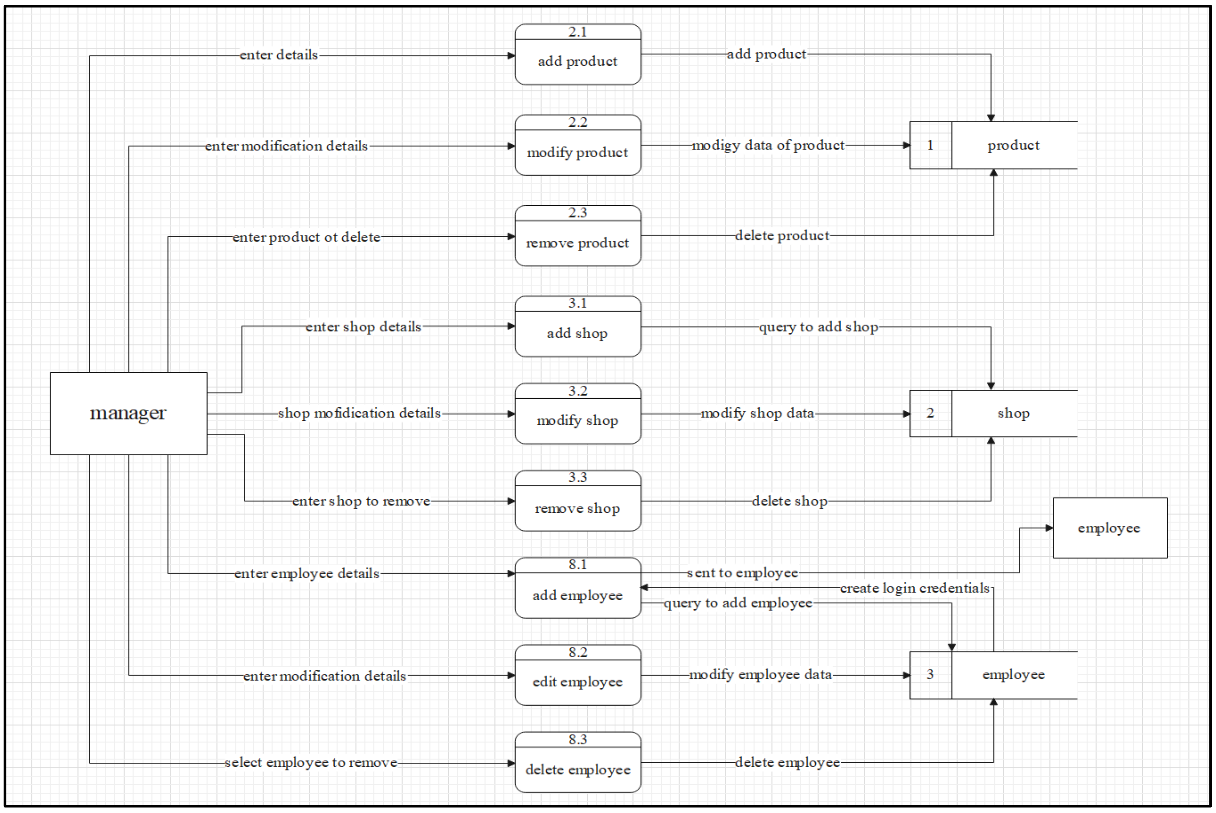


Figure 3.23 Level 2 Data Flow Diagram

3.5.5 Class Diagram:

Class diagrams are used in developing an object-oriented system model to show the classes in a system and the associations between these classes. An object class is a general definition of one kind of system object. An association is a link between classes that indicates that there is a relationship between these classes. Each class may need to have some knowledge of its associated class. The symbols used in class are as follow:

Name	Symbol	Description
Class	**Name** + attribute1: type - attribute2: type # atrribute3: type / attribute4: type +operation1()	This box is used to define a class and the attributes and operation are listed in it along with access modifiers
Public	+	The plus sign states that attribute can be accessed in any other class
Private	-	The dash sign state that attribute cannot be accessed by any other class
Composition	◆——	A composition states the one class is totally depended on the other class
Aggregation	◇——	An aggregation states that one class is partially associated with the other class
One	—— 1	This describes only one entity participation in the association
Many	—— 1..*	This describes one or more entity participation in the association

Table 3.5 Class Diagram Symbols

References: https://www.edrawsoft.com/article/class-diagram-relationships

Diagram:

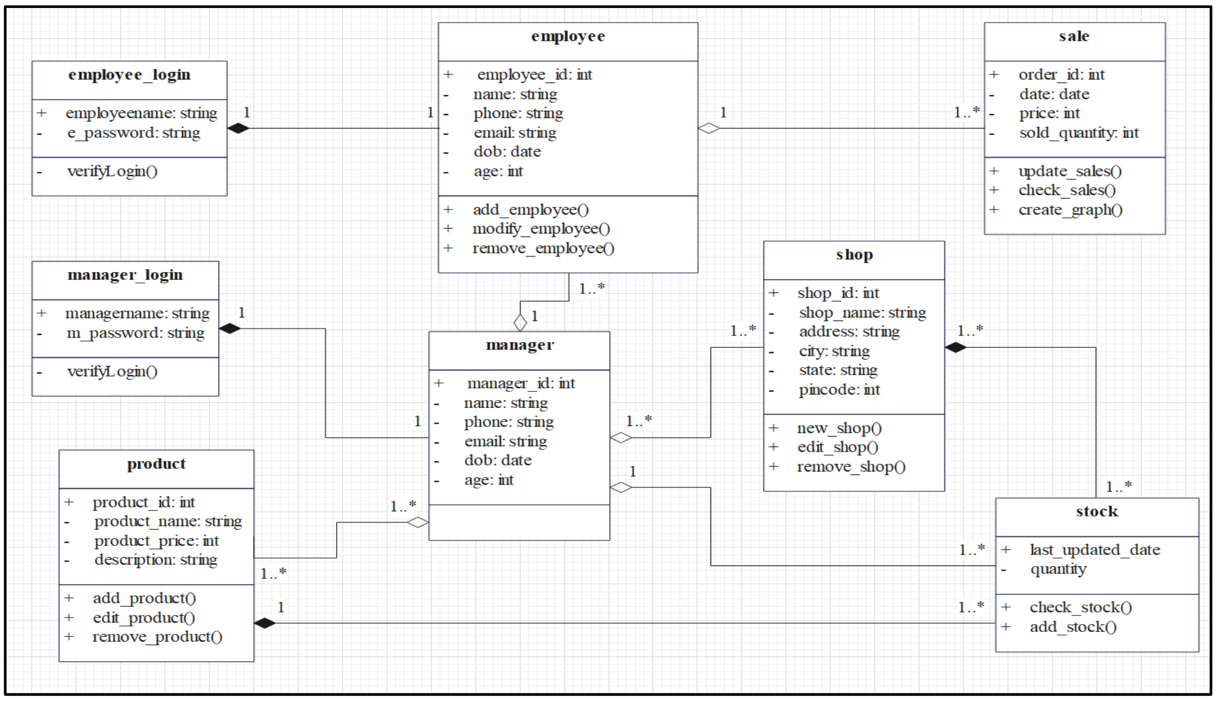


Figure 3.24 Class Diagram

3.5.6 Use Case Diagram:

Use Case Diagram is used to show interaction between the system and its environment. A use case can be taken as a simple scenario that describes what a user expects from a system. Each use case represents a discrete task that involves external interaction with the system. A use case is shown in ellipse with the actors involved in the use case. Use Case Diagram uses line without arrow and line with arrow indicates the direction of flow of messages. The symbols used to create a Use Case Diagram are as follow:

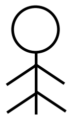
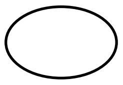
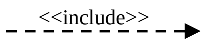
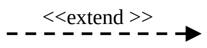
Name	Symbol	Description
Actor		The actors are used to define the environment interacting with the system
Use Case		A Use Case describes the functionality of the system
Association		An association is used to show interaction of actors with use cases
Include		This association states that the base use case is executed with the help of include use case
Extend		The extend states that the extend use case will be executed after the execution of base use case but it will not always execute

Table 3.5 Use Case Diagram Symbols

References: Software Engineering 9th Edition

https://www.lucidchart.com/pages/uml-use-case-diagram

Diagram:

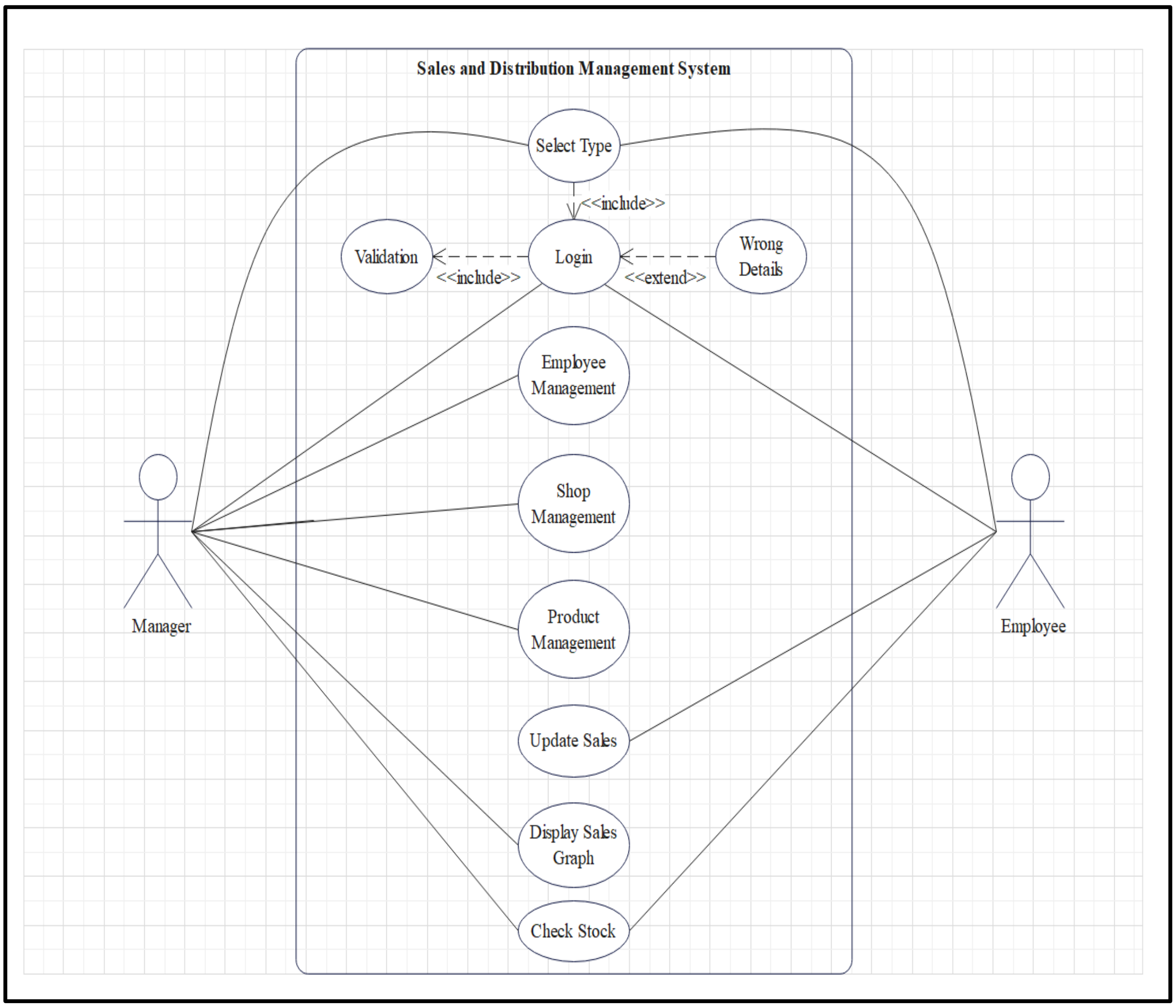


Figure 3.25 Use Case Diagram

Description for Use Cases:

1. Select Type

Description: The user needs to select the type of login whether user is admin or employee. This will be a normal button click process.

Actors: Manager and Employee.

Data: Click from user and setup of the application with ample amount of disk space available.

Stimulus: Request from both Manager or Employee.

Response: Directed to the manager or employee login page according to selection.

Comments: Requires setup of the software in machine.

2. Login

Description: The user who selected manager will be redirected to manager login and manager must enter credentials to login. Same for the user selected employee. Functionality of login page will change according to selection.

Actors:	Manager and Employee.
Data:	The manager panel will require managername and password from the manager. The manager login credentials are necessary. The employee panel will require employeename and password.
Stimulus:	Manager’s and Employee’s request for login and work with software
Response:	If the login credential is proper then a short message of login successful is displayed and manager or employee is directed to manager dashboard or employee dashboard respectively.
Comments:	The Manager or Employee should have login credential provided by company.

3. Employee Management

Description:	This will help the manager to operate employee. The manager can perform add, update, and delete operation on the employee who is working for the company or left the company.
Actors:	Manager.
Data:	Bio data of the employee to get some details of employee and a confirmation from the company.
Stimulus:	Manager’s request for the operation of employee.
Response:	If all the operation request are proper, employee operation will be successfully carried out and if not, then error will be generated.
Comments:	The manager must have a permission to carry out the operations.

4. Shop Management

Description:	This use case will carry all operations related to shops e.g., add new shop, update existing shop or delete shop data.
Actors:	Manager.
Data:	The shop id of the shop should be entered by the manager to the system.
Stimulus:	Shop operation request from any manager.
Response:	The manager will get a message of changes save successfully.
Comments:	Manager needs an authorization from the company.

5. Product Management

Description:	This use case will help to carry on all the operations with product details including addition of new product, removal of existing product or specification update of product.
Actors:	Manager.
Data:	The product id field from the admin is required.
Stimulus:	Request of change in product data details or removal of product.

Response:	The manager will get message of modification confirmation after all conditions satisfied.
Comments:	Authorization permission from company.

6. Update Sales

Description:	This will help the employee to keep maintaining records of daily sale regularly which will be reflected in database.
Actors:	Employee.
Data:	Products, sold quantity and price should be provided by employee.
Stimulus:	Sale request from employee as the product got sold.
Response:	The employee will get a message of data modified and changes will be reflected in other fields respectively through database.
Comments:	Employee should have a valid employee account validated by manager.

7. Display Sales Graph

Description:	This use case creates a sale graph and will show it accurately with all details.
Actors:	Manager.
Data:	Minimal data from manager such as date and products with shops.
Stimulus:	Need of viewing sale of products.
Response:	The graph along with product details will be displayed to the manager in structured manner.
Comments:	The employee should keep updating data time to time and manager should have permission to view sales.

8. Check Stock

Description:	This use case is used for checking availability of stock in a particular shop.
Actors:	Manager and Employee.
Data:	The product id and name from the employee or manager is required.
Stimulus:	Stock checking request from manager or employee.
Response:	Remaining stock of the selected product will be displayed to the Manger or Employee who requested for stock.
Comments:	Manager or Employee should have access granted to view stock of shops.

3.5.7 Sequence Diagram:

Sequence Diagrams in the UML are used to model the interactions between the actors and the object in system and the interaction between the objects themselves. The UML has a rich syntax for sequence diagrams, which allows many kinds of interaction to be modelled.

A sequence diagram shows the sequence of interactions that take place during a particular use case or use case instance. The objects and actors involved are listed along the top of the diagram with a dotted line drawn vertically from these. Interactions between objects are indicated by annotated arrows. The rectangle of the dotted lines indicates the lifeline of the object concerned i.e., the time that object instance is involved in the computation.

The sequence diagram is read from top to bottom. The annotations on the arrows indicate the calls to the object, their parameters, and the return values. The about symbols are used to draw a sequence diagram.

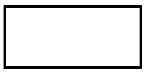
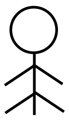
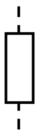
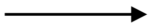
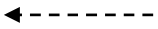
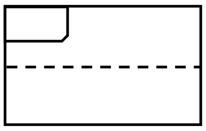
Name	Symbols	Description
Objects		This symbol is used to define the data objects
Actor		The actors are used to define the environment interacting with the system
Lifeline of the Object		The amount of time for which the object instance is involved in the computation.
Message		It is used to show the interactions between actor and objects.
Return Message		It returns a message for a request
Alternative Frame		This frame is used to show two different conditions and their working.

Table 3.6 Sequence Diagram Symbols

Diagram:

1. Select Type:

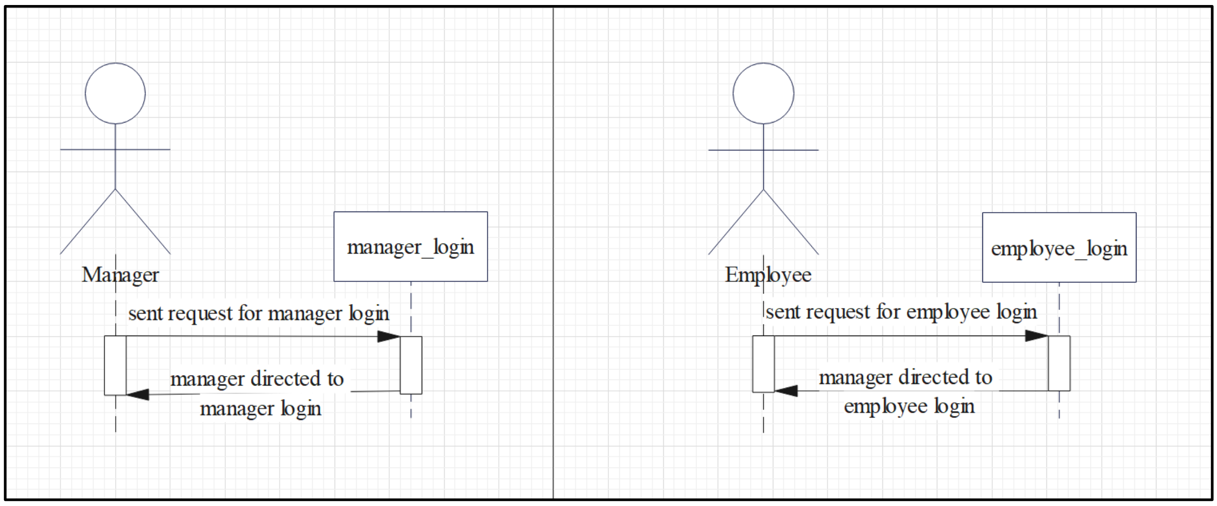

Manager
manager_login
sent request for manager login
manager directed to manager login
Employee
employee_login
sent request for employee login
manager directed to employee login

Figure 3.26 Select Type Sequence Diagram for Manager and Employee

2. Login:

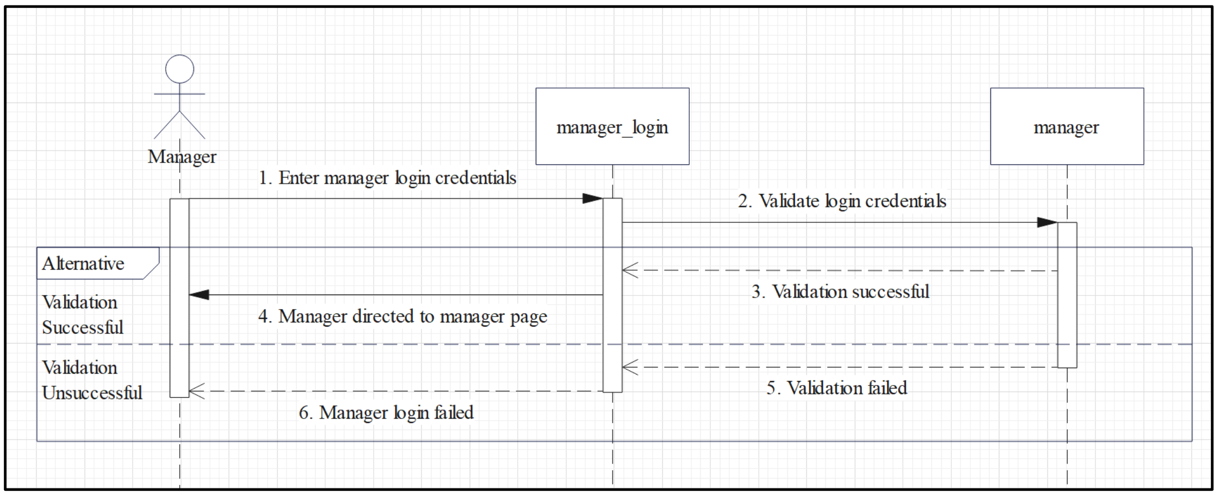


Figure 3.27 Login Sequence Diagram for Manager

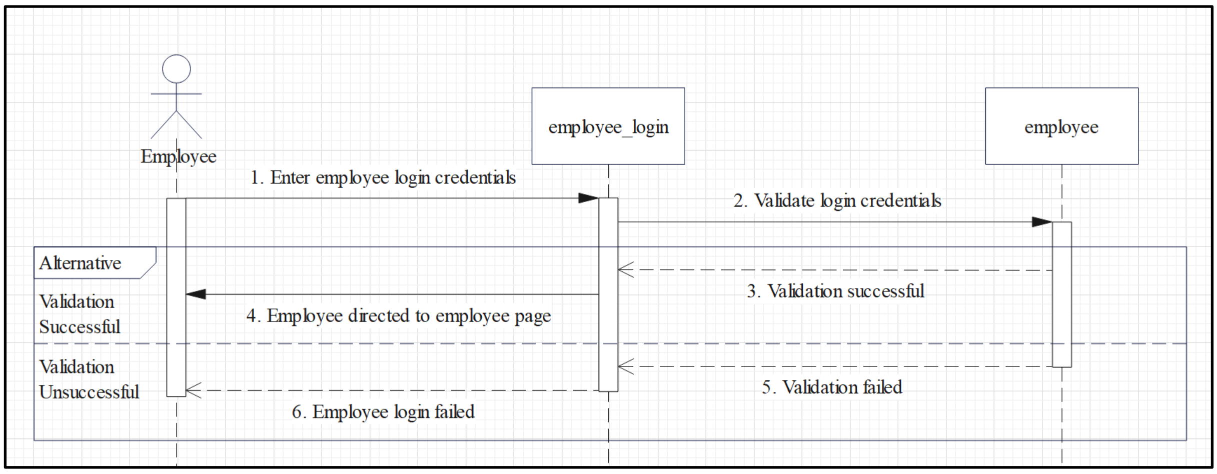


Figure 3.28 Login Sequence Diagram for Employee

3. Employee Management:

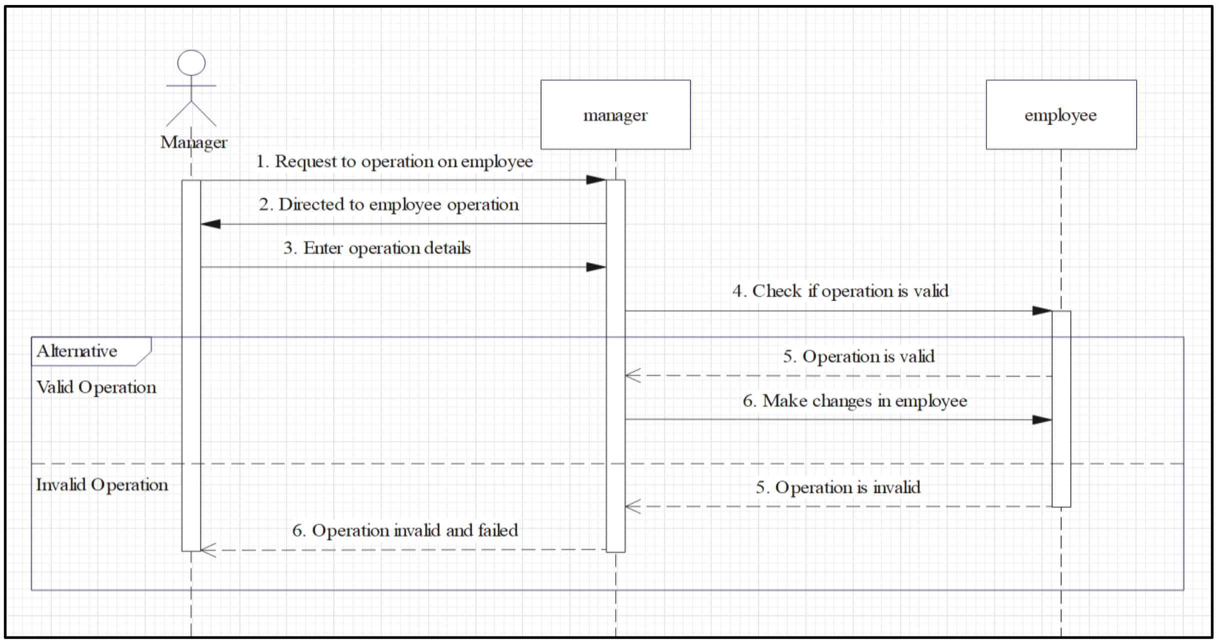
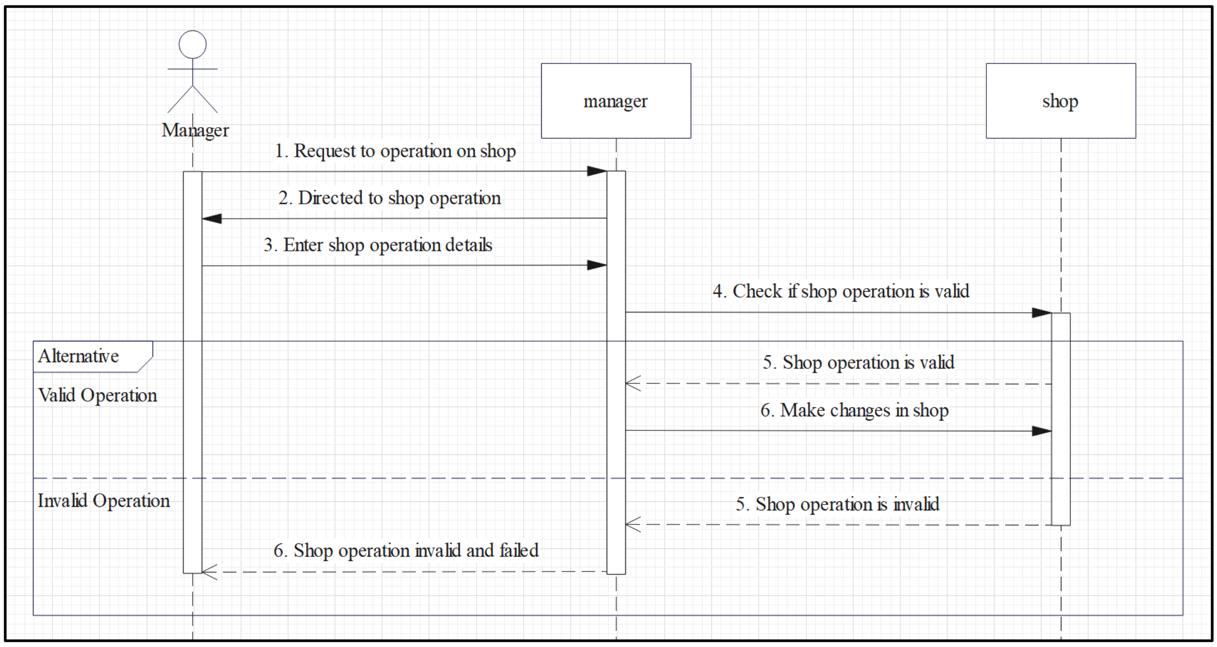

Manager
manager
employee
1. Request to operation on employee
2. Directed to employee operation
3. Enter operation details
4. Check if operation is valid
Alternative
Valid Operation
5. Operation is valid
6. Make changes in employee
Invalid Operation
5. Operation is invalid
6. Operation invalid and failed

Figure 3.29 Employee Management Sequence Diagram

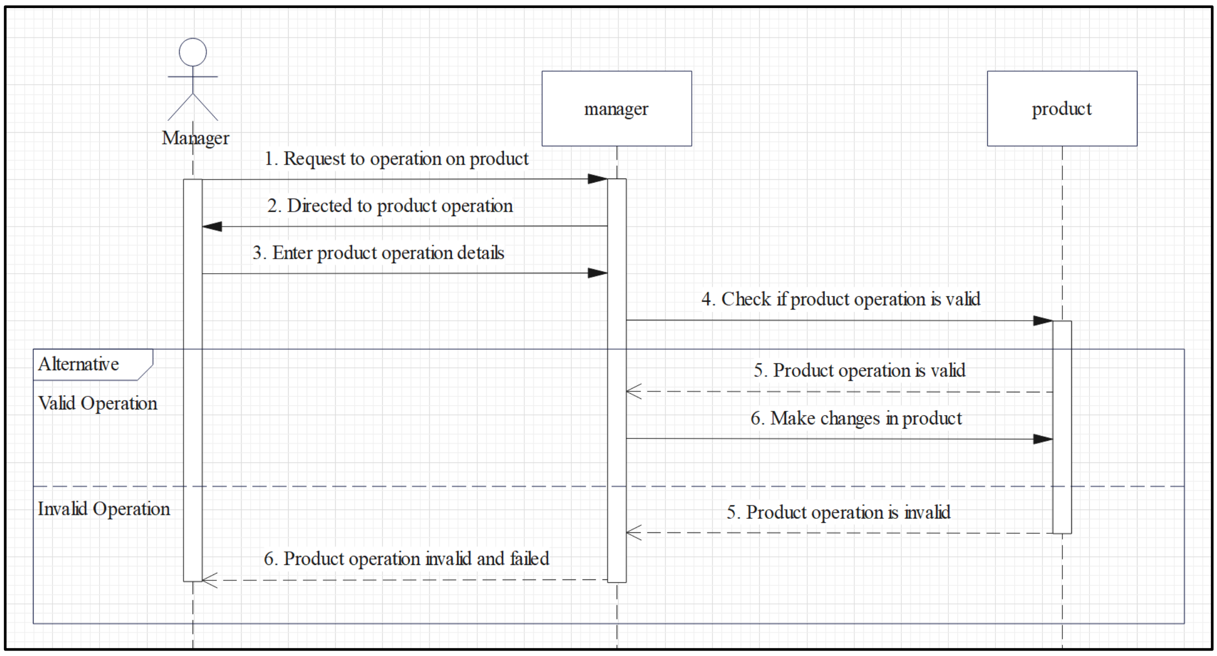
4. Shop Management:



Manager
manager
shop
1. Request to operation on shop
2. Directed to shop operation
3. Enter shop operation details
4. Check if shop operation is valid
Alternative
Valid Operation
5. Shop operation is valid
6. Make changes in shop
Invalid Operation
5. Shop operation is invalid
6. Shop operation invalid and failed

Figure 3.30 Shop Management Sequence Diagram

5. Product Management:



Manager
manager
product
1. Request to operation on product
2. Directed to product operation
3. Enter product operation details
4. Check if product operation is valid
Alternative
Valid Operation
5. Product operation is valid
6. Make changes in product
Invalid Operation
5. Product operation is invalid
6. Product operation invalid and failed

Figure 3.31 Product Management Sequence Diagram

6. Update Sales:

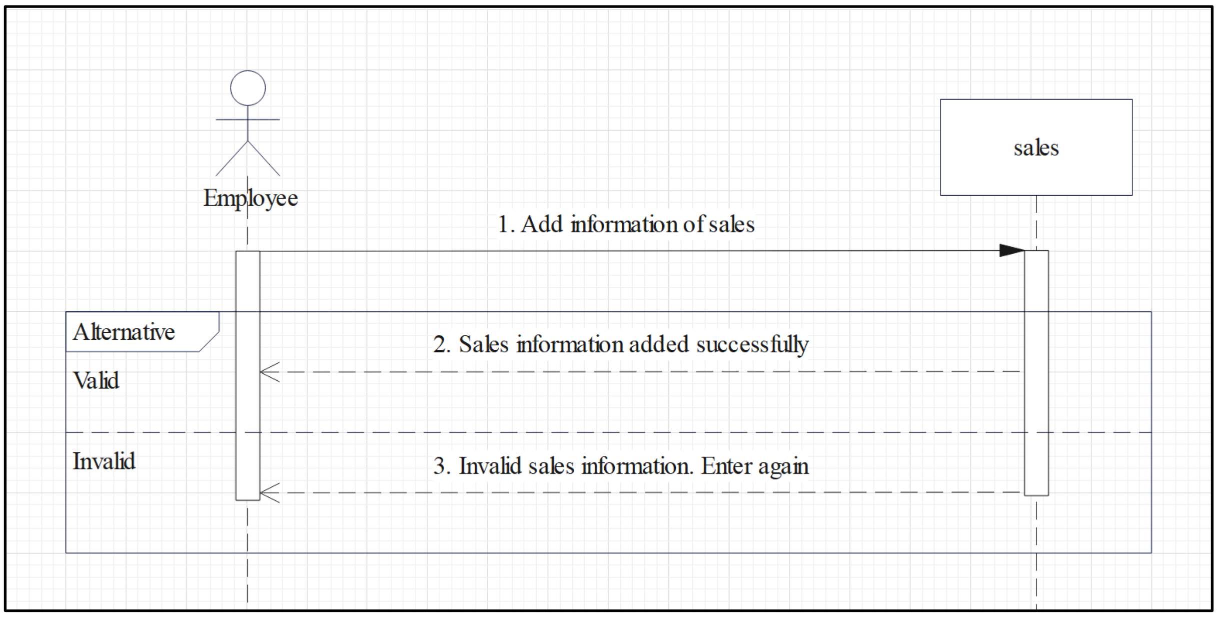

Employee
sales
1. Add information of sales
Alternative
Valid
2. Sales information added successfully
Invalid
3. Invalid sales information. Enter again

Figure 3.32 Update Sales Sequence Diagram

7. Display Sales Graph:

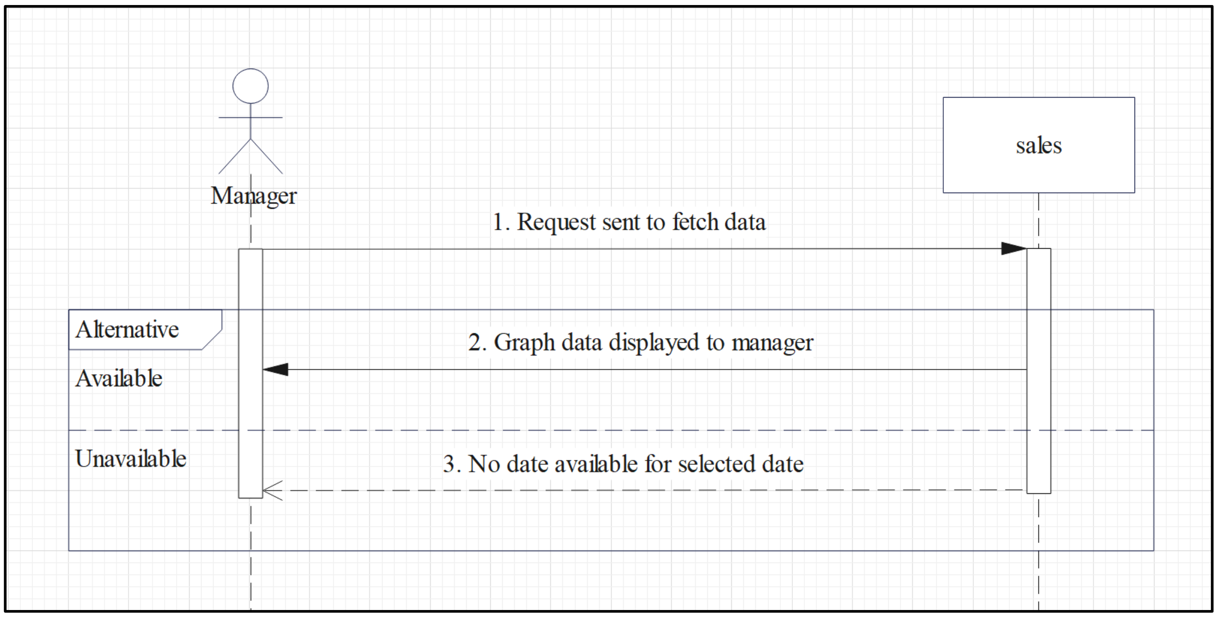

Manager
sales
1. Request sent to fetch data
Alternative
Available
2. Graph data displayed to manager
Unavailable
3. No date available for selected date

Figure 3.33 Display Sales Graph Sequence Diagram

8. Check Stock:

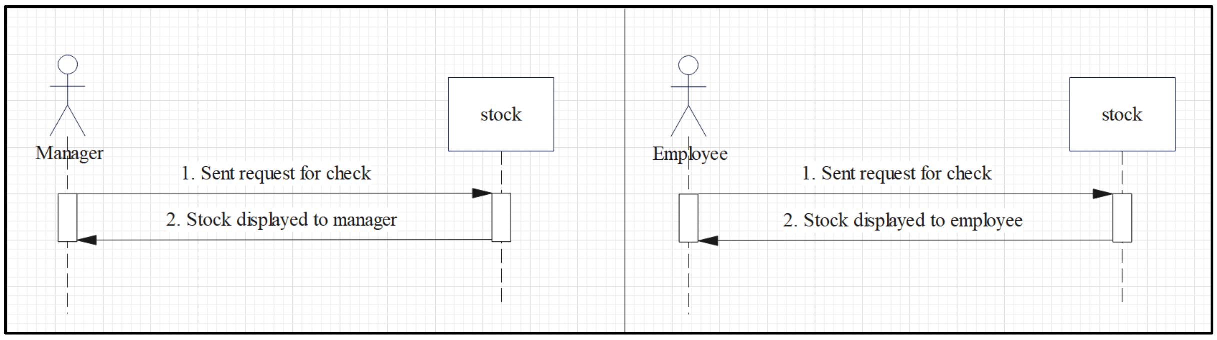

Manager
stock
1. Sent request for check
2. Stock displayed to manager
Employee
stock
1. Sent request for check
2. Stock displayed to employee

Figure 3.34 Check Stock Sequence Diagram for Manager and Employee

3.5.8 Activity Diagram:

An Activity Diagram show the activities involved in a process or in data processing. Activity Diagrams are intended to show the activities that make up a system process and the flow of control from one activity to another. The start of a process is indicated by a filled circle and the end by a filled circle inside another circle. Rectangles with round corners represent activities i.e., the specific sub-processes that must be carried out. An arrow is used to represent the flow of work form one activity to another.

A solid bar is used to indicate activity coordination. When the flow from more than one activity leads to a solid bar then all of these activities must be complete before progress is possible. When the flow from solid var leads to several activities, these may be executed in parallel. The symbols used in an Activity Diagram are:

Name	**Symbol**	**Description**
Start	○	This symbol indicates start of an activity
End	◉	This symbol indicates end of an activity
Activity	▢	This symbol indicated the activities of a particular system.
Flow of Work	→	This symbol is used to show the flow of work from activity to activity
Decision	◇	This symbol represents the decision parameter and used where a decision is need to be taken

Table 3.7 Activity Diagram Symbol

Reference: Software Engineering 9th edition

Diagram:

1. Manager:

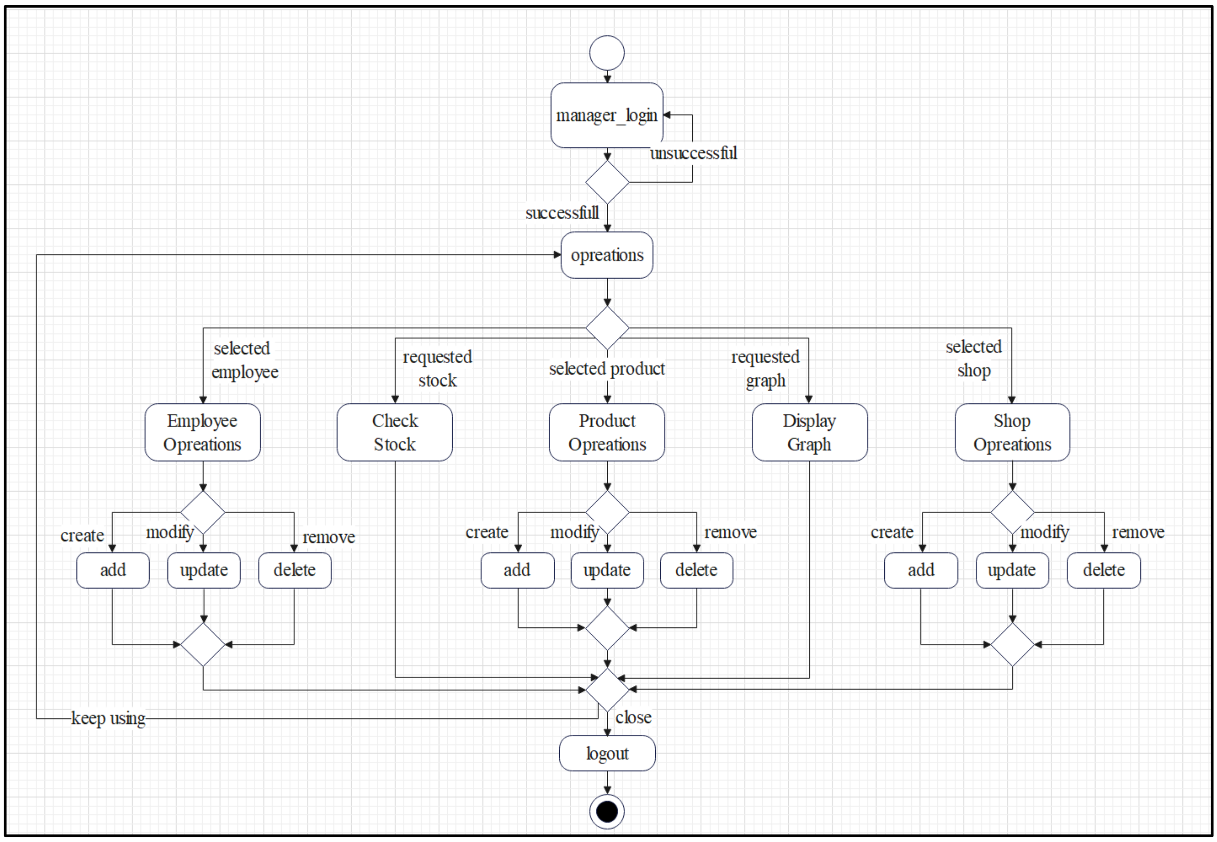


Figure 3.35 Admin's Activity Diagram

2. Employee:

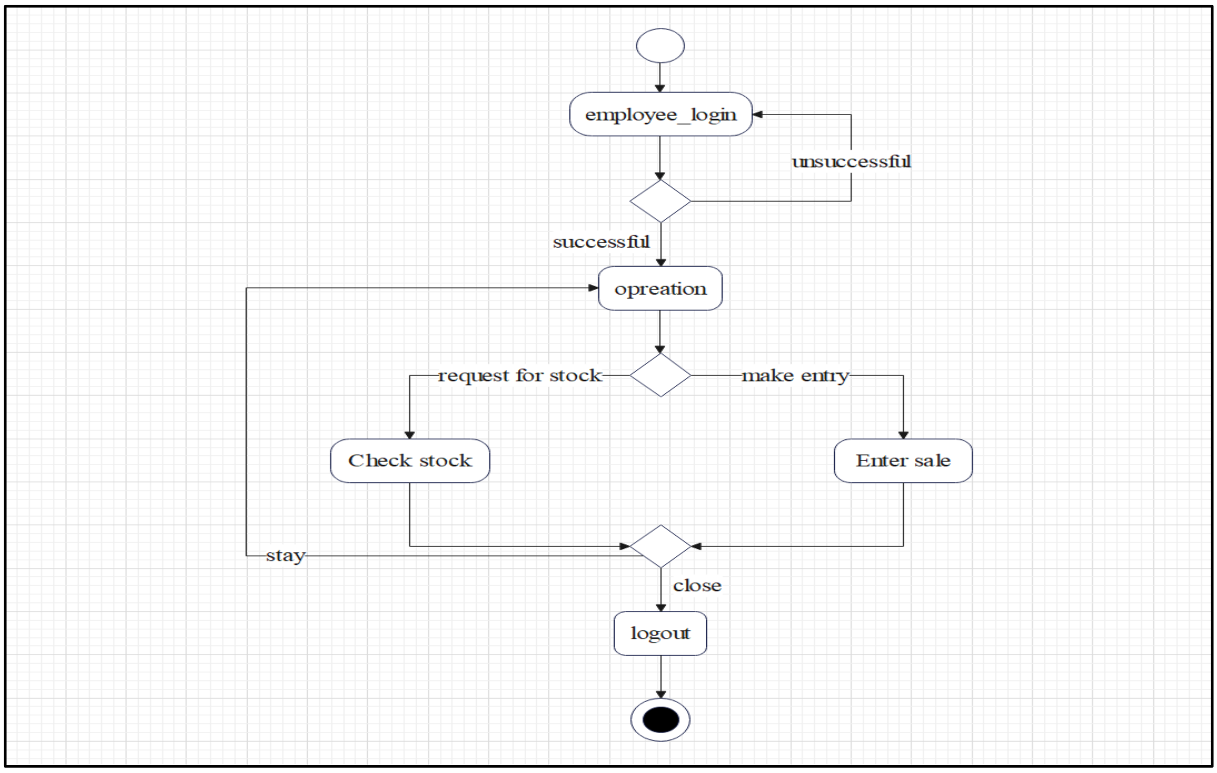


Figure 3.36 Employee's Activity Diagram

3.5.9 State-Chart Diagram:

A State-Chart Diagram show system states and events that cause transitions from one state to another. They do not show the flow of data within the system but may include additional information on the computation carried out in each state. In State Chart Diagrams, rounder rectangles represent system states. They may include brief description of the actions taken in that state by using do keyword. The labelled arrow represents stimuli that force a transition from one state to another. It show the states of a system and the events that trigger a transition from one state to another. The symbols used to draw a State-Chart Diagrams are:

Name	**Symbol**	**Description**
Start	○	This symbol indicates start of an activity
End	◉	This symbol indicates end of an activity
Activity	▢	This symbol indicated the activities of a particular system.
Flow of Work	→	This symbol is used to show the flow of work from activity to activity

Table 3.8 State-Chart Diagram Symbol

Diagram:

1. Login:

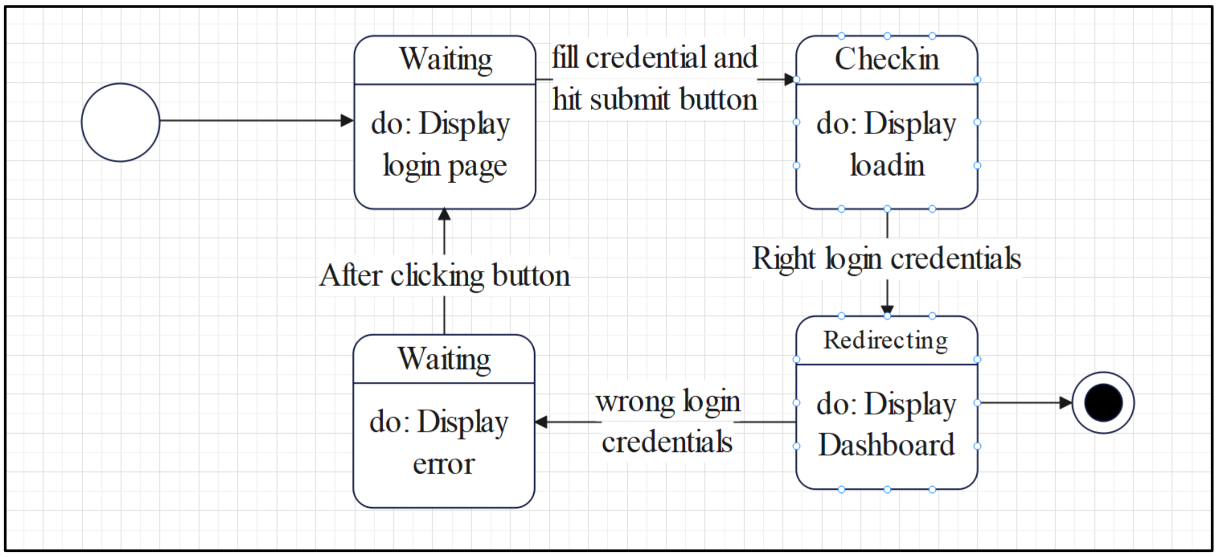


Figure 3.37 Login State-Chart Diagram

2. Employee Management:

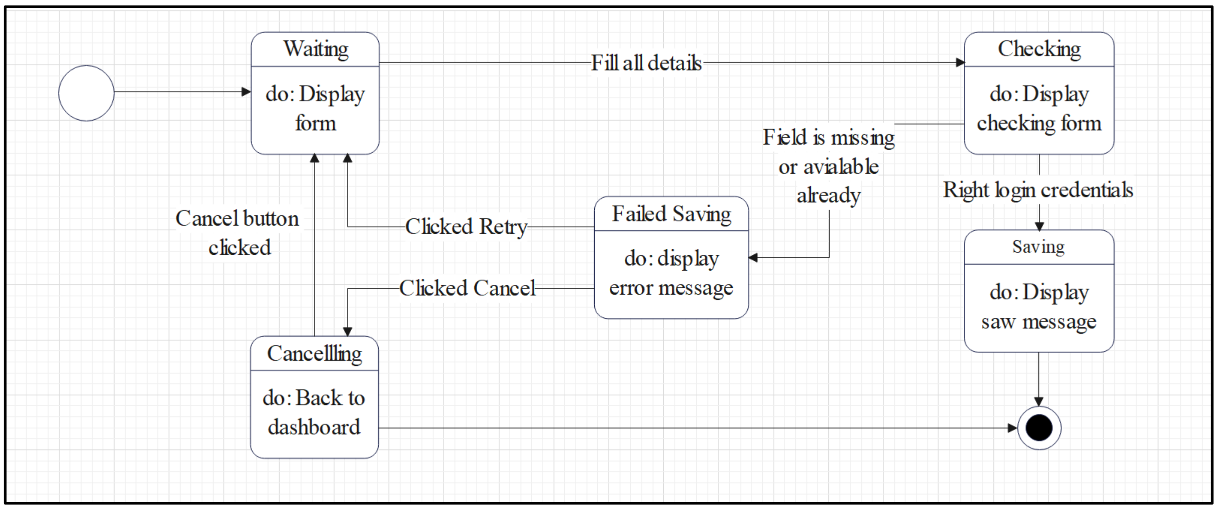


Figure 3.38 Employee Management State-Chart Diagram

3. Check Sales:

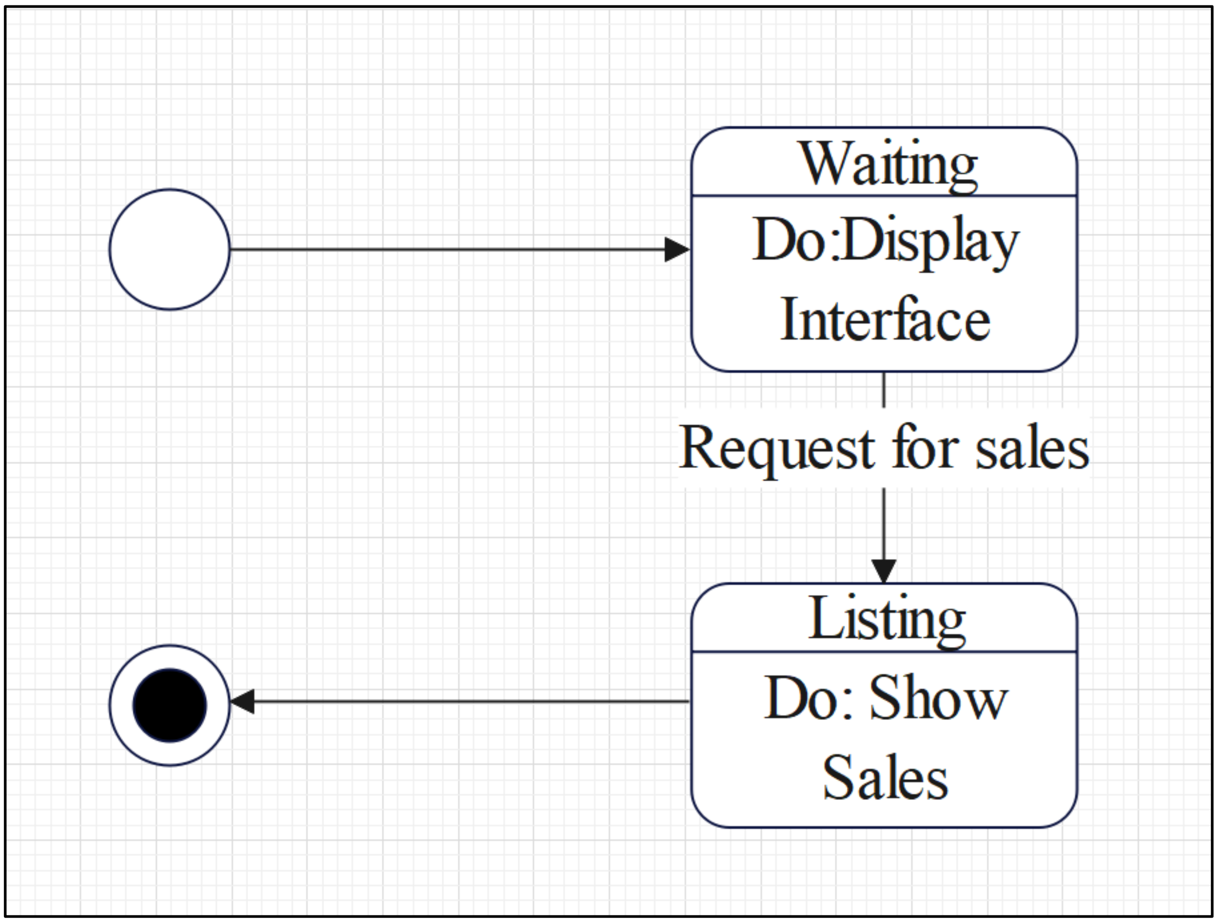


Figure 3.39 Check Sales State-Chart Diagram

4. Shop Management:

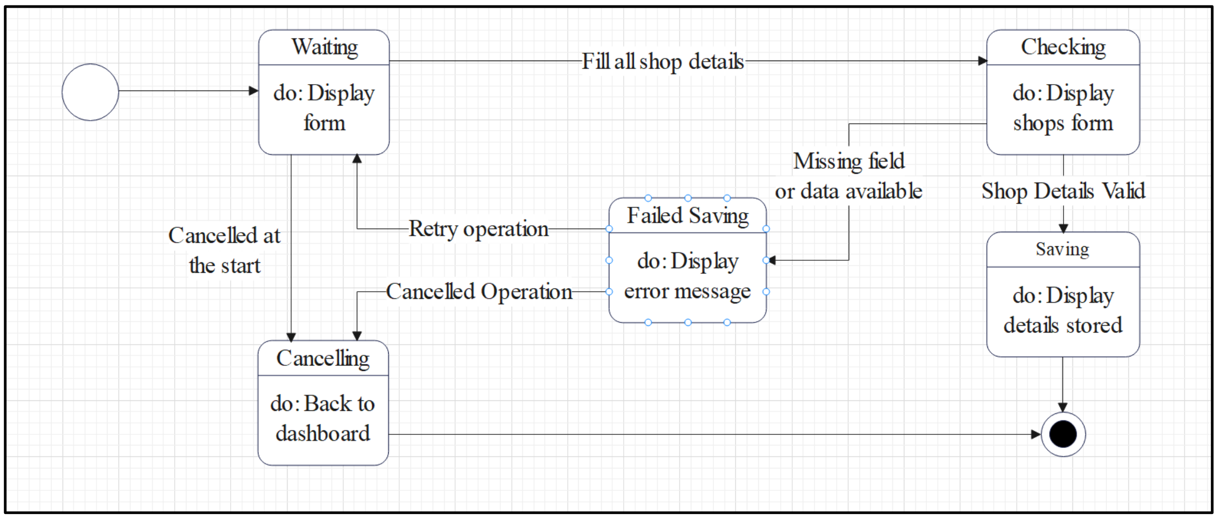


Figure 3.40 Shop Management State-Chart Diagram

5. Product Management:

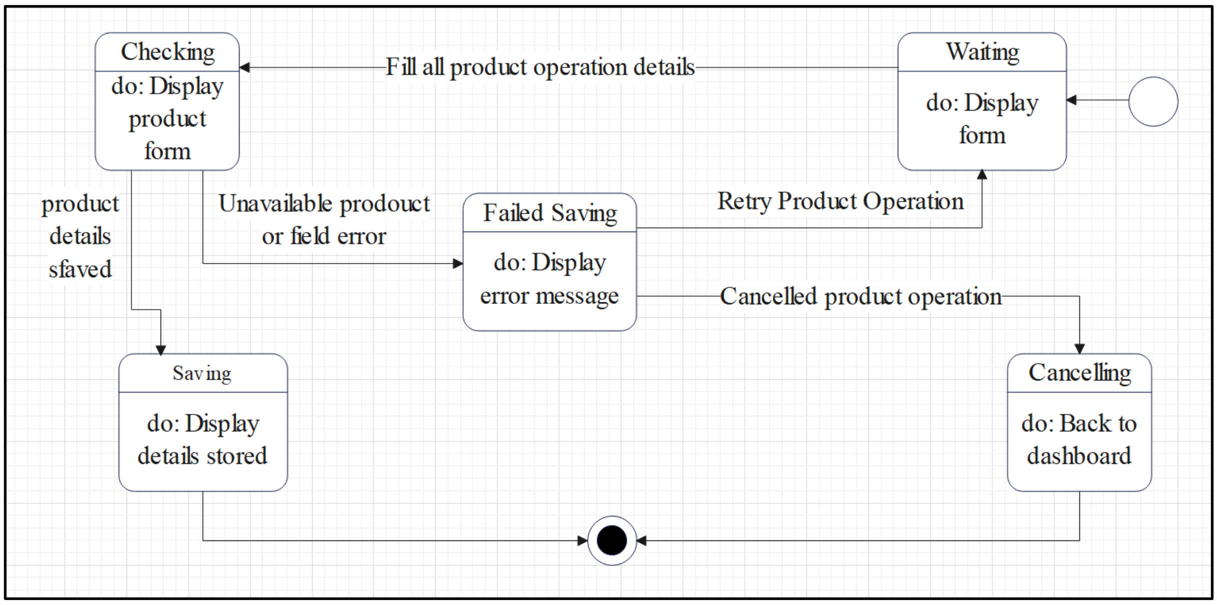


Figure 3.41 Product Management State-Chart Diagram