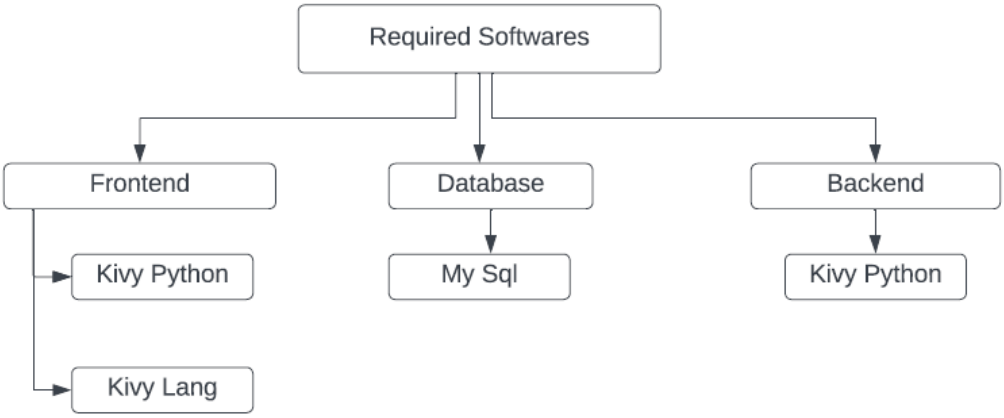
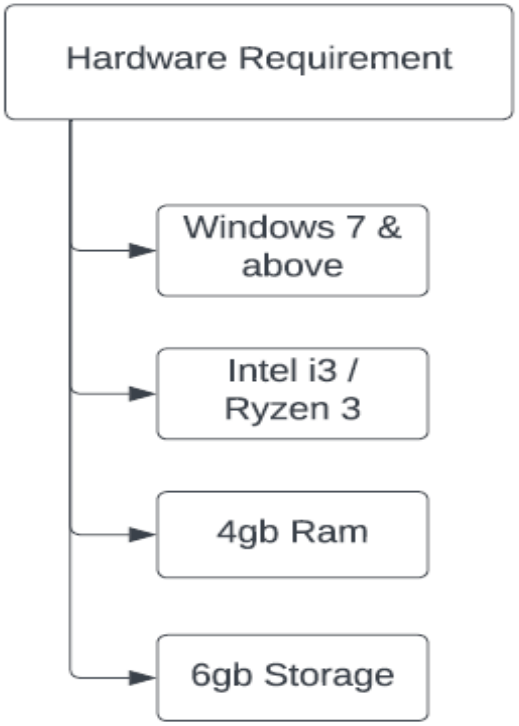
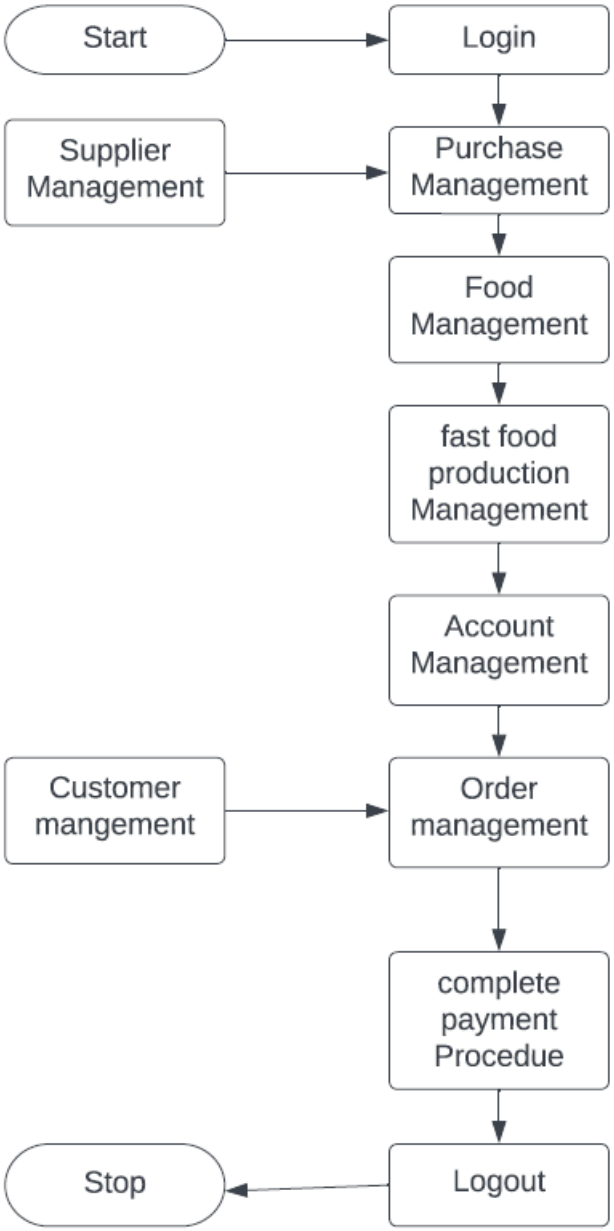
Software and Hardware Requirement:-

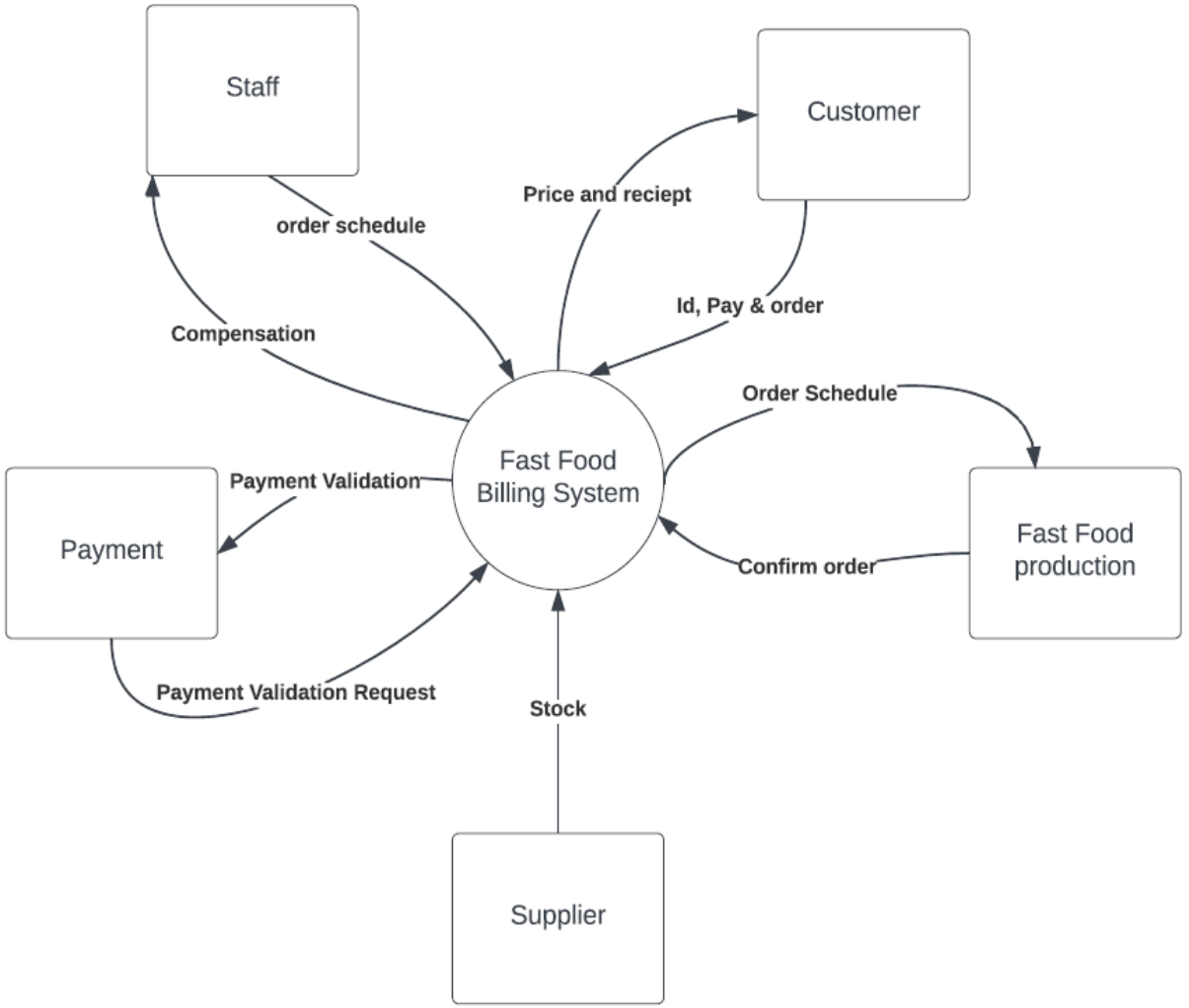




Flow Chart For Fast food Billing System:-



Context Diagram



Experiment No.-03.

Aim:- ER (Entity Relation) Diagram for fast food billing system

Attribute is represented by the oval shape. This will be the columns or fields of each table in the Online Food Ordering System.

Relationship is represented by diamond shape. This will determine the relationships among entities. This is usually in a form of primary key to foreign key connection.

We will follow the 3 basic rules in creating the ER Diagram.

1. Identify all the entities.
2. Identify the relationship between entities and
3. Add meaningful attributes to our entities.

In the Online Food Ordering System we have the following entities

- User
- Site Information
- Payment
- Order
- Order Details
- Customer
- Rating
- Menu
- Menu Type

Our design of Online Food Ordering System consists of 9 entities; the specified entities will be our database tables in the design and implementation of Online Food Ordering database schema.

We will now draw the entities of the Online Food Ordering System specified above and it will be represented by a rectangle shape. The image below is the entities identified in the scope of the Online Food Ordering System

- The users manage/update the site information (1 to 1 relationship).
- The user processes the orders of the customers (1 to many relationship).
- The user processes the payment of the customers (1 to many relationship).
- The customer places their orders (1 to many relationship).
- Order information can contain 1 or more items (1 to many relationship).
- An order detail contains 1 or more menu (1 to many relationship).
- The order information will be linked to the payment module (1 to 1 relationship).
- The customer gives their rating on a menu (1 to 1 relationship).
- A menu has multiple ratings from the customers (1 to many relationship).
- A menu belongs to a specific menu type (1 to 1 relationship).

The last part of the ERD process is to add attributes to our entities.

User Entity has the following attributes:

- ID – primary key represented with underline
- Full name
- Contact
- Email address
- Username
- password

Site Information Entity has the following attributes:

- ID – primary key represented with underline
- Name
- Description
- Contact Info
- Address
- User ID – foreign key
- Last Update

Payment Entity has the following attributes:

- ID – primary key represented with underline
- Order ID – foreign key
- Amount
- Paid By
- Date
- Processed By – foreign ey

Order Entity has the following attributes:

- ID – primary key represented with underline
- Customer ID – foreign key
- Order Date
- Total Amount
- Order Status
- Processed By – foreign key

Order Details Entity has the following attributes:

- ID – primary key represented with underline
- Order ID – foreign key
- Menu ID – foreign key
- Amount
- No of Serving
- Total Amount

Customer Entity has the following attributes:

- ID – primary key represented with underline
- First name
- Last name

- Middle name
- Email
- Phone Number
- Landline
- Profile Image
- Username
- Password
- Account Status

Rating Entity has the following attributes:

- ID – primary key represented with underline
- Menu ID – foreign key
- Score
- Remarks
- Date Recorded
- Customer ID – foreign key

Menu Entity has the following attributes:

- ID – primary key represented with underline
- Name
- Price
- Type ID – foreign key
- Image
- Ingredients
- Status

Menu Type Entity has the following attributes:

- ID – primary key represented with underline
- Type Name
- Description

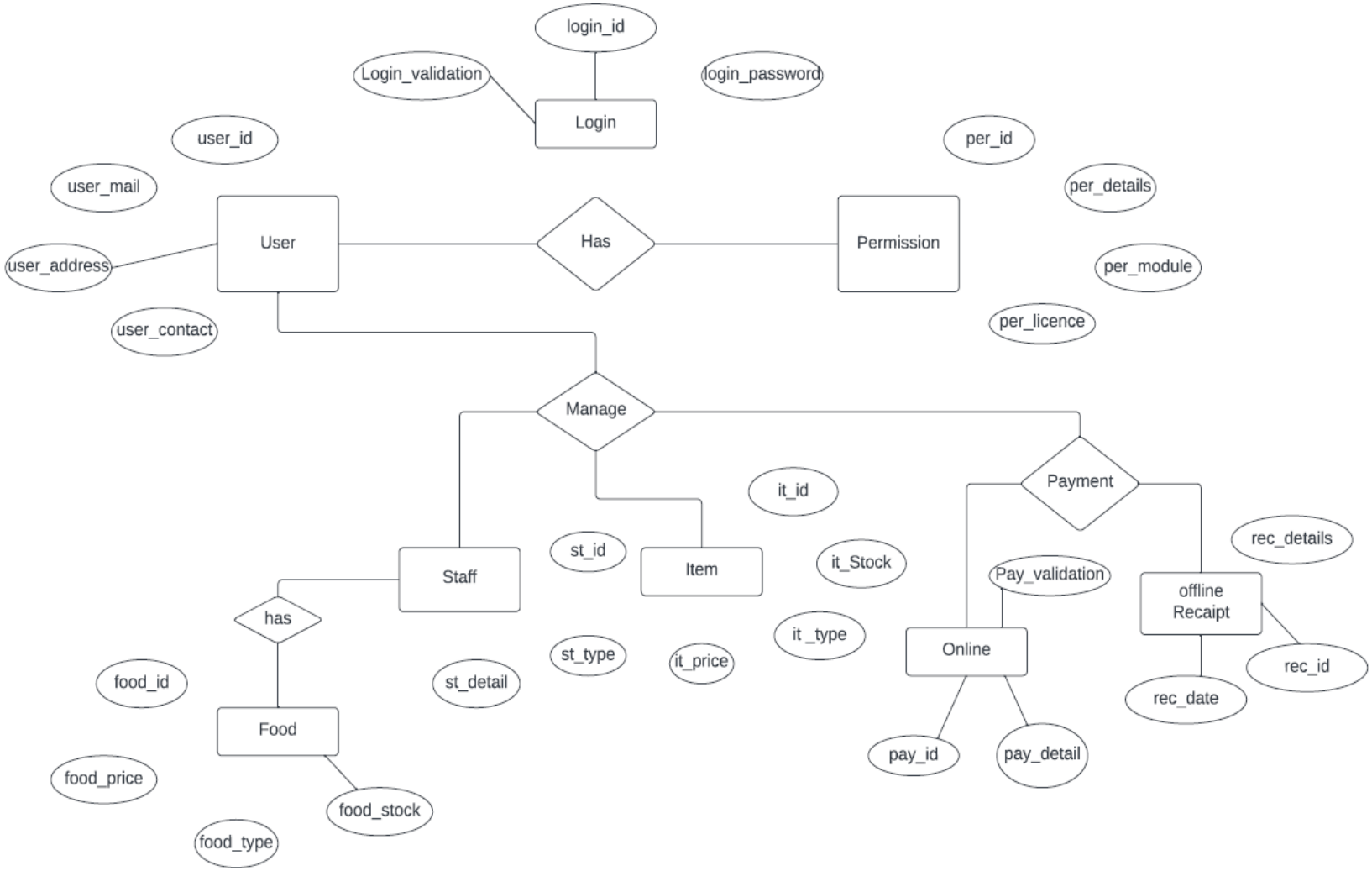
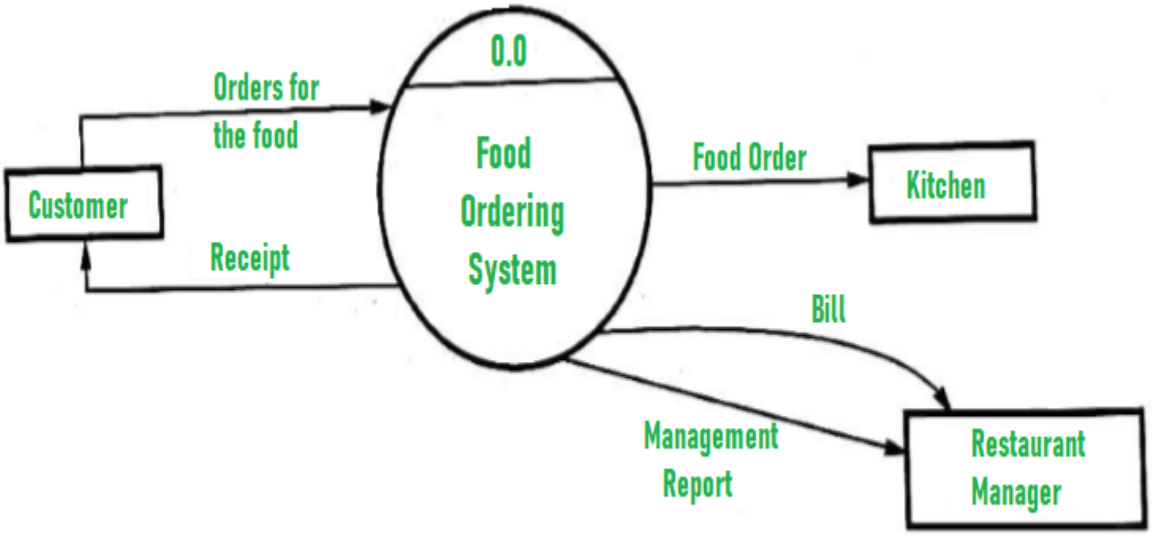


fig:-ER Diagram for Fast Food billing System

Experiment No.-04.

Aim:- Data Flow Diagram at Level 0 and Level 1 For Fast Food Billing System

1. **Level 0 DFD –**
   At this level, the Input and Output of the system are shown. The system is designed and established across the world with input and output at this level.



Level 0 DFD (Context Level

   Food Ordering System has the following input :

   - Food order is input as the customer's order for food.

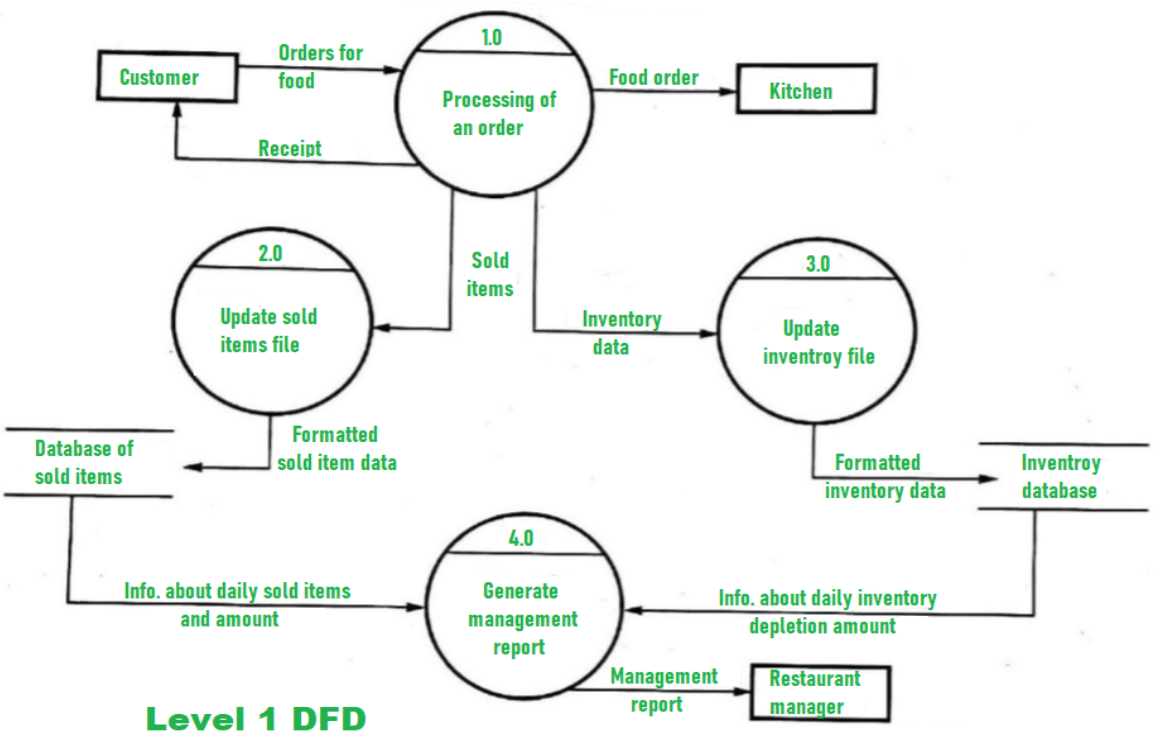
   Food Ordering System has the following output:

   - Receipt of the order.
   - For further processing the order, the food order is passed to the kitchen.
   - The restaurant manager gets the report of Bill and Management.

2. **Level 1 DFD –**
   For processing the order, process 1.0 is responsible. For food, the housekeeping activities involved are represented by processes 2.0, 3.0, and 4.0. The detailed information about daily sold items should be available to create and report management and the list of items that are available 'in-stock' should be kept by maintaining the inventory data (describes the records of datasets such as their name, their content, source, many useful information, etc.) at the same time.
   Hence, two data stores are used in this level of DFD given below :

   - Database of Sold items

- Inventory database



In the end, with the use of the amount of daily sold items and daily inventory depletion, it is easy to prepare a report of management. Further, the restaurant manager gets this report of management.

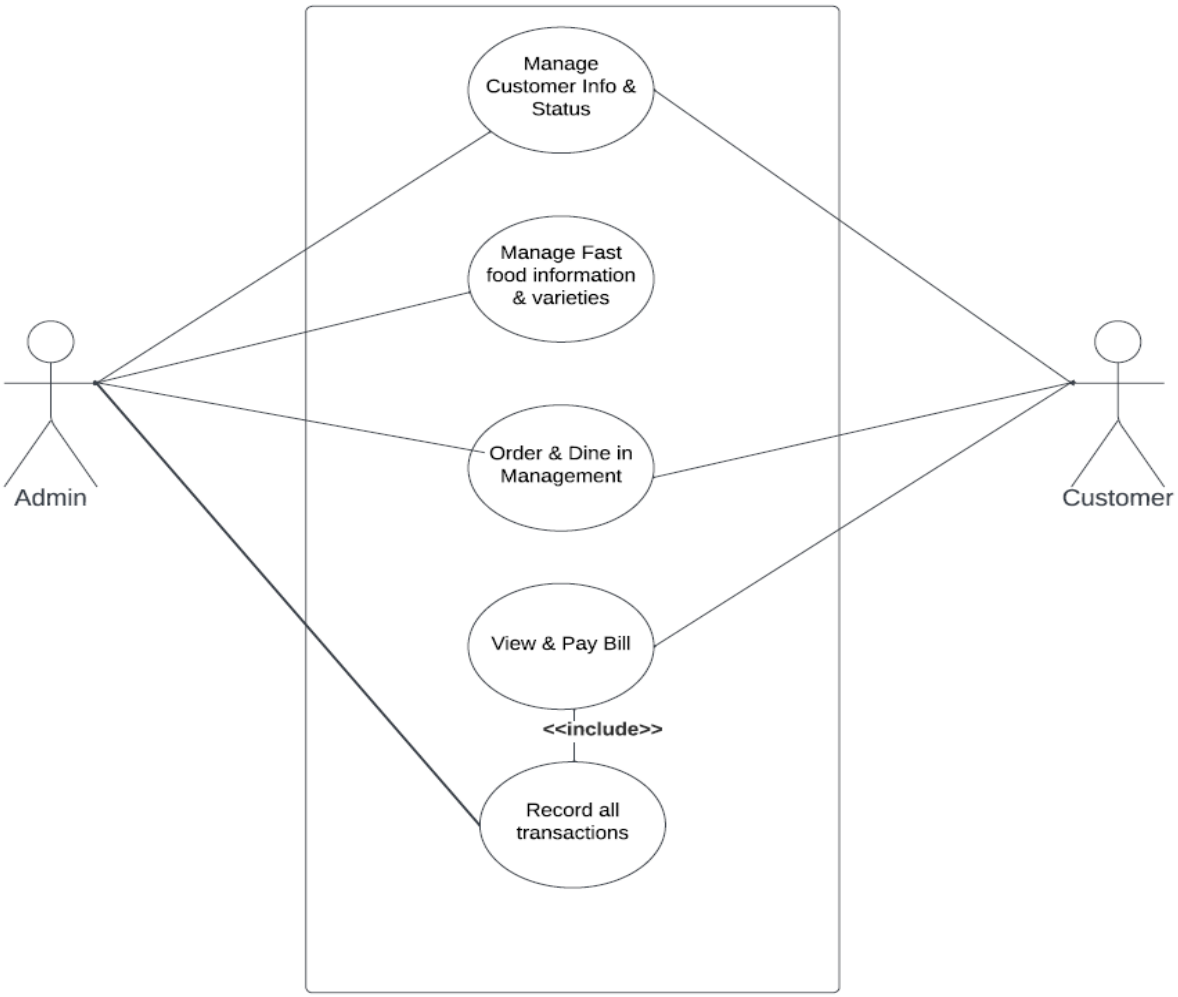
Experiment No.- 05.

Aim:- Use Case Diagram for Fast Food Billing System

A use case diagram is a visual representation of how a user might interact with a program. A use case diagram depicts the system's numerous use cases and different sorts of users. The circles or ellipses are used to depict the use cases.Fast food Billing System Project Use Case Diagram Potential Features:The features of the system depends on the users of it.

- Monitor and Manage Customers' Information and Status
- Manage Food Information and Varieties
- Orders Online and Dine In Management
- Manage Revenue and Expenses
- Record all Transaction

By creating the use case of the Restaurant Management System, you must determine first the possible features to identify the flow of the system. After that you can now create the blueprint or core of the system function.



Experiment No-06.

Aim:- Activity Diagram For Fast Food Billing System

When the flow of events is linear, a textual description of behavior is often sufficient to capture the system behavior. Activities diagram provide a visual documenting sequence of task making up a single activity. They especially are useful for activities governed by conditional logic, and flow of event running concurrently. We describe the basic system functionality with textual use cases, and employ activity diagrams for a visual representation of the corresponding sequence of task or flow of information.

Use Case 1. Place Order

Primary Actor: Customer
Description: Customer places an order from the available choices after indicating his language preference for the session.
Pre-conditions: System is connected to a power source, display is turned on and system is configured to accept the inputs.
Flow of Events:

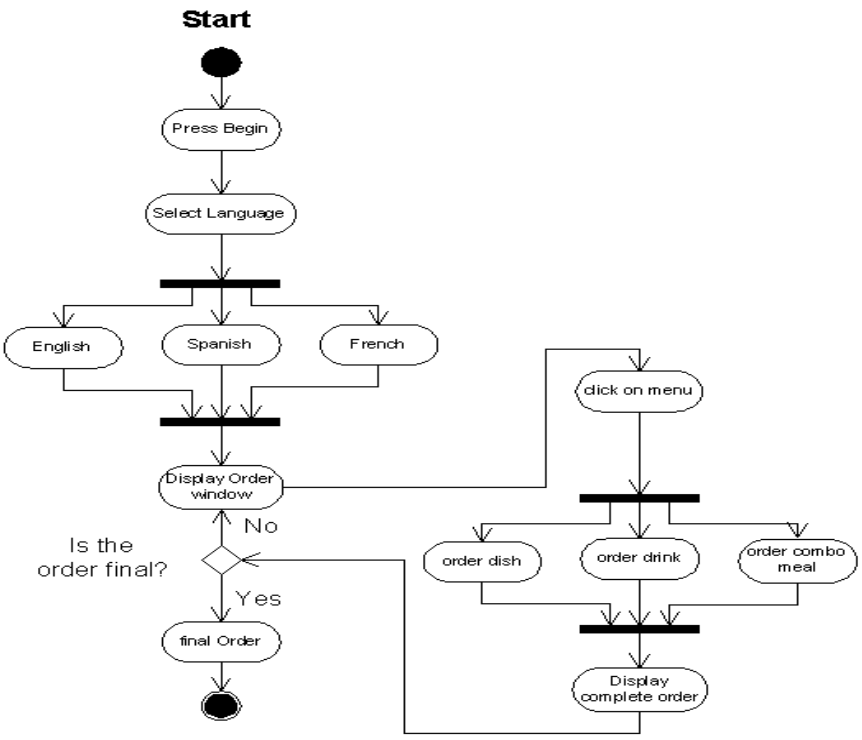
1. User selects his language preference for the session.
2. User selects from the menu.
3. User selects from the drinks menu
4. User selects from the combo deals
5. User confirms the order

Alternative Flow of Events:

6. User accidentally presses a wrong button and after realizing it he hits the backspace button.
7. User enters a wrong order and wants to go back to the main menu.

Post-condition: Order has been made that goes to the kitchen for processing.
Assumption: User is familiar with how to enter values through mouse and has a general idea why the inputs are being provided and what is expected out of system.

Activity diagram for this use case is given as below:



Use Case 2. Make Payment

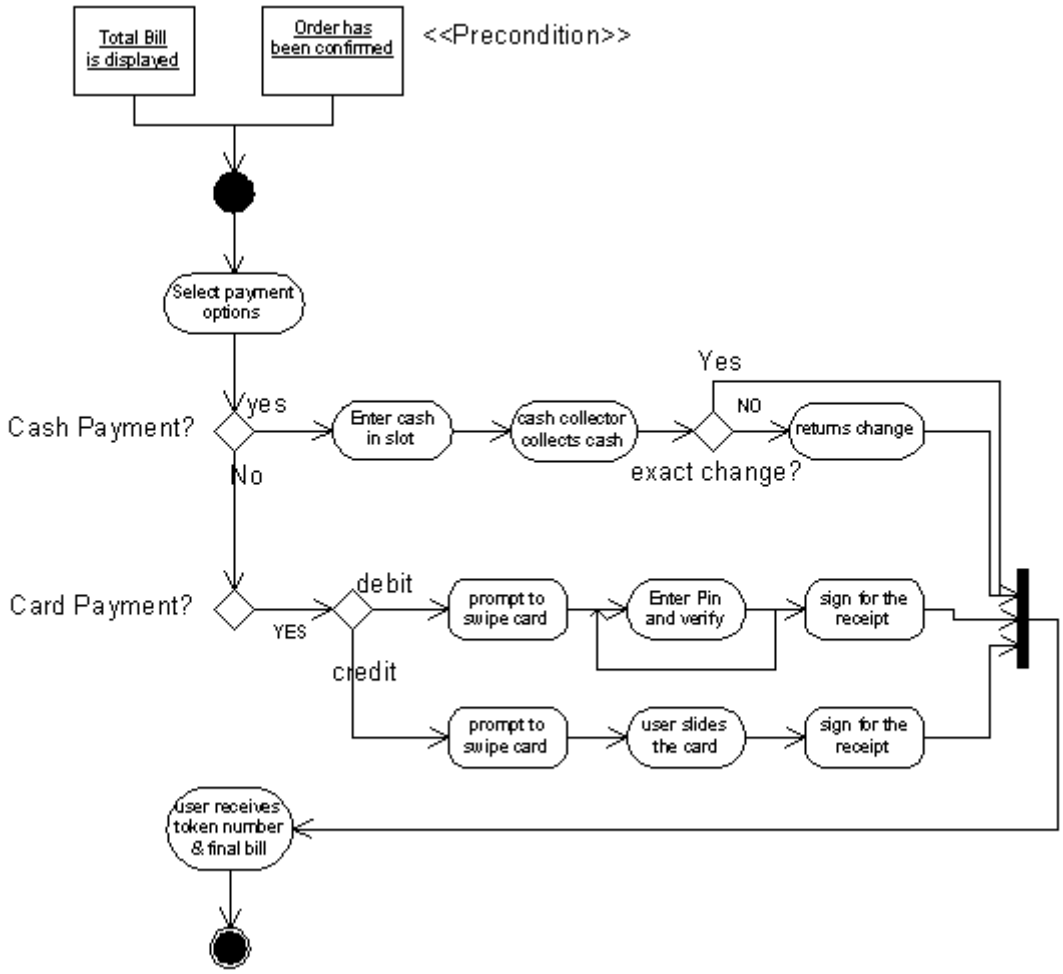
- Primary Actors: customer, Credit/Debit system, cash collector.
  Description: The user is asked for the mode of payment. The payment is accepted in terms of credit/debit card or is collected by cash collector. And the customer is given a token with their order number.
  Pre-condition: The order has been confirmed and the total bill has been displayed on the screen to the customer. Costumer decides to go ahead with the order. Flow of Events:
  1. User enters the mode of payment. (Credit/Debit/Cash)
  2. User makes the payment in cash
  3. Cash collector collects the money and gives back the change if required.
  4. User makes the payment by credit/debit card.
  5. User receives a token number and final bill.

  Alternative Flow of Events:

  6. User selects the mode of payment.

  Post-condition: Customer waits for the order to be processed.
  Assumption: User is familiar with how the system works and what is expected out of system.

Activity diagram for this use case is given as below:



Use Case 3. Update Menu.

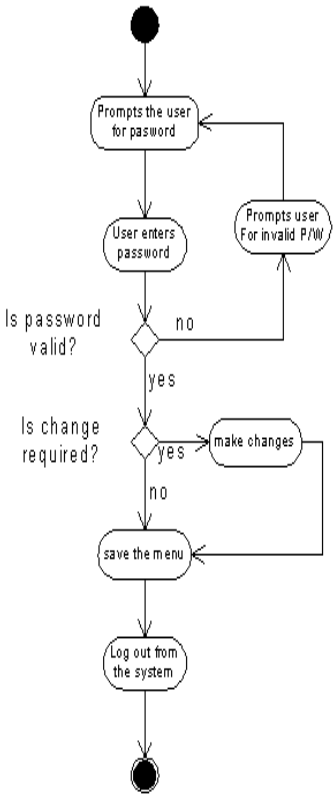
- Primary Actor: Store Manager.
  Description: The menu might change according to the inventories or add/delete items from menu and deals. The prices of each item might change for the period of time.
  Pre-condition: An order menu with their respective price already exists in the system in some particular format.
  Flow of Events:
  1. The Store manager enters the system with some password.
  2. The Store manager makes the required changes.
  3. The Store manager saves the changes and logs out.

  Alternative Flow of Events:

  4. Some of the menu might not need any change.
  5. User might enter invalid password and need to go back.

  Post-condition: A menu list will be displayed when the user enters the system.
  Assumption: The Store manager is given the rights and privileges to enter the system and make the required changes.

Activity diagram for this use case is given below:



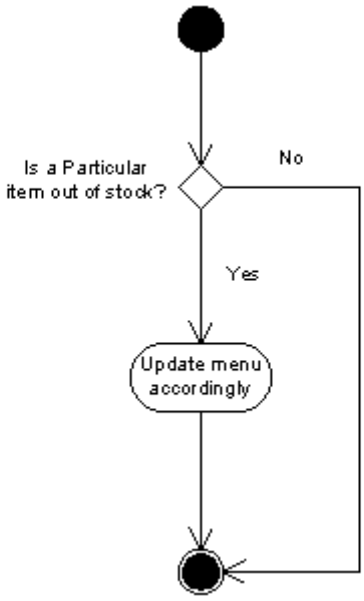
Update Menu - Activity diagram

Use Case 4. Monitor Inventory.

- Primary Actor: Food preparation person, Store Manager Description: This use case triggers when an item goes out of stock.
  Pre-condition: None
  Flow of Events:
  1. Food preparation person/Store manager notices an item out of stock
  2. Updates the menu accordingly.

  Post-condition: A new and updated menu list will be displayed.
  Assumption: The Store manager is given the rights and privileges to enter the system and make the required changes.

Activity diagram for this use case is given below:

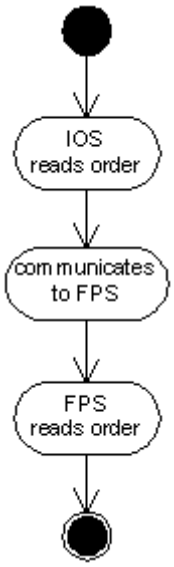


Use Case 5. Read Order.

- Primary Actor: Food preparation person, Internal Order system.
  Description: Internal order system reads the order once the customer confirms his order and then he communicates the order to the food preparation person. Pre-condition: User confirms the order.
  Flow of Events:
  1. Internal order system reads the order
  2. Communicates the order to the food preparation person

  Post-condition: The final order is being processed in the kitchen.
  Assumption: Food preparation person is available to take the order and know the sequence of processing the orders.

Activity diagram for this use case is given below:



Experiment No.- 07

Aim:- State Chart Diagram for fast food billing system

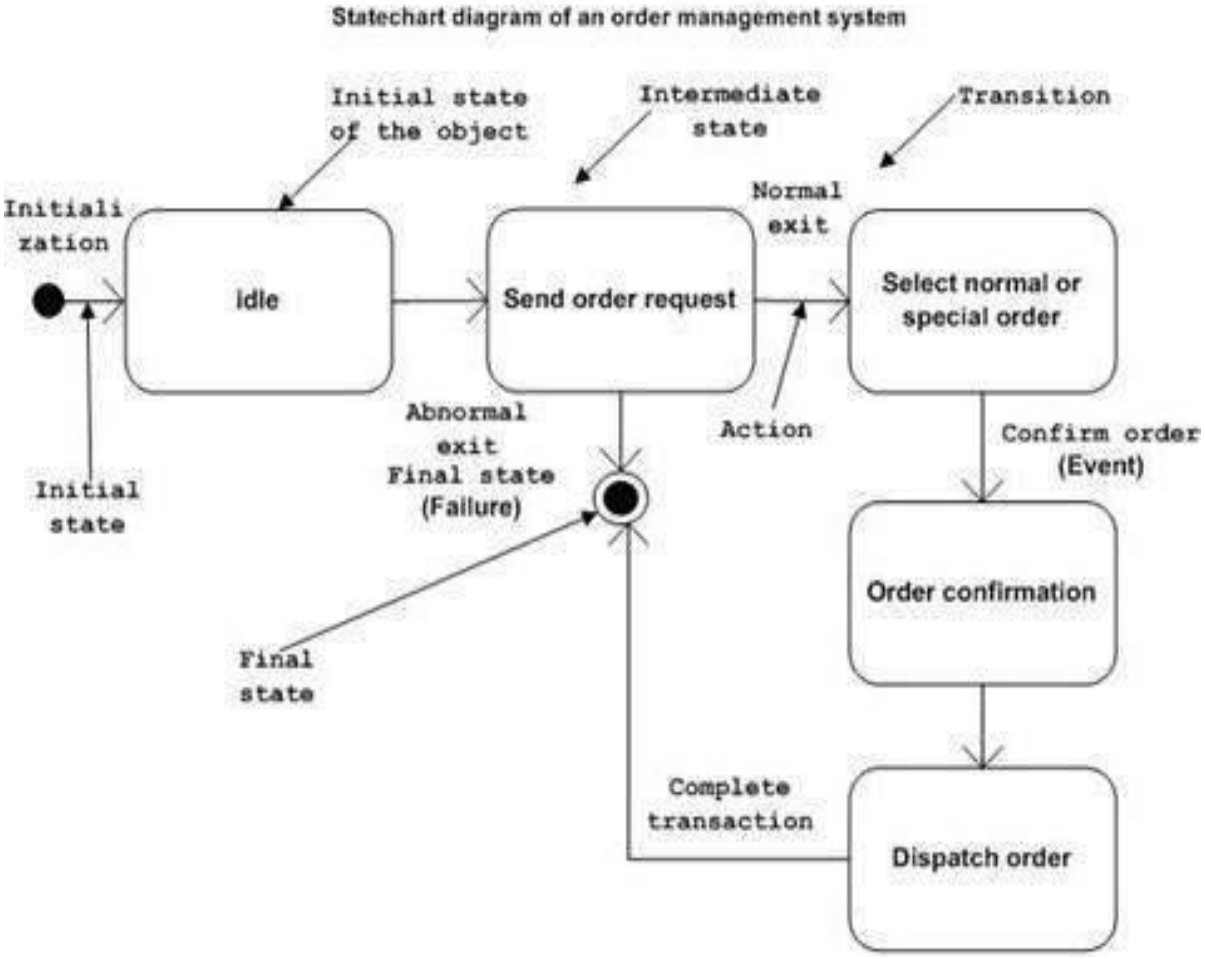
Statechart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. Statechart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

Statechart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of Statechart diagram is to model lifetime of an object from creation to termination.

Statechart diagrams are also used for forward and reverse engineering of a system. However, the main purpose is to model the reactive system.

Following are the main purposes of using Statechart diagrams −

- To model the dynamic aspect of a system.
- To model the life time of a reactive system.
- To describe different states of an object during its life time.
- Define a state machine to model the states of an object.



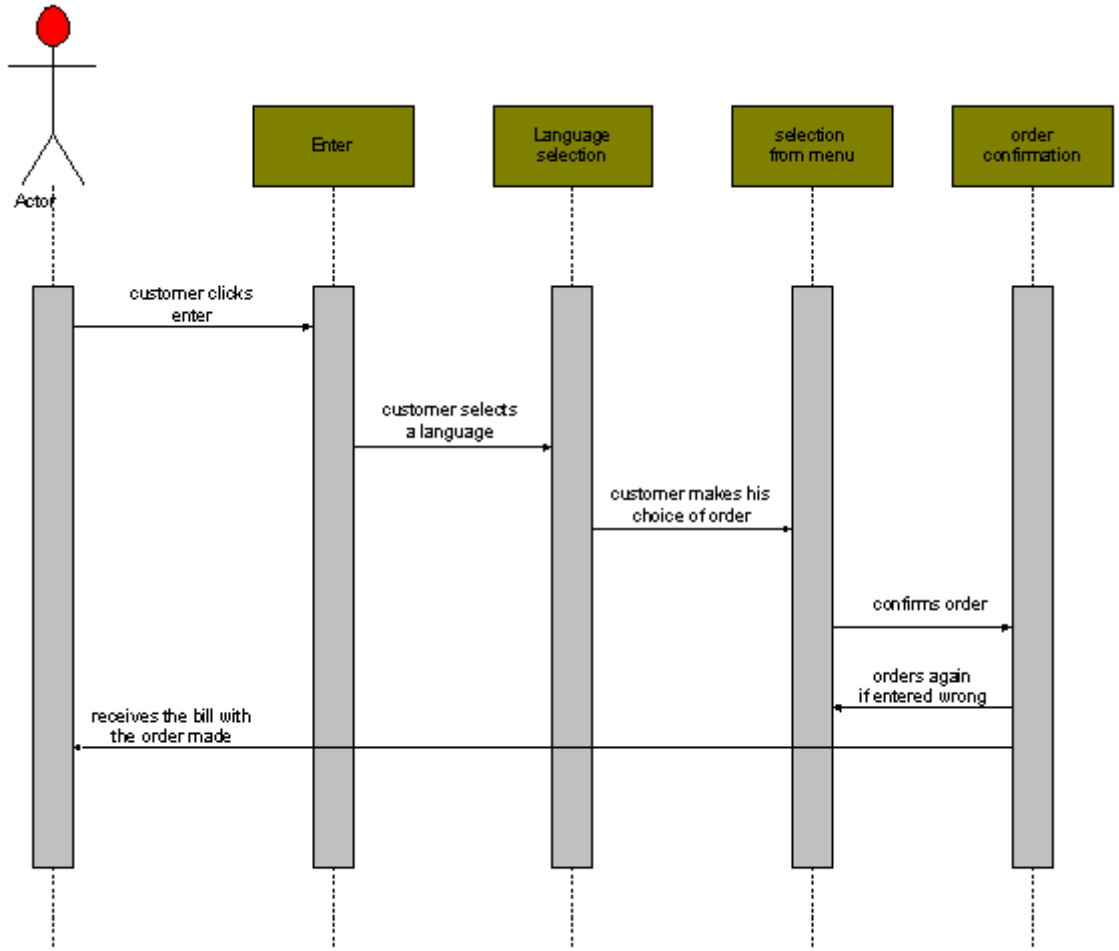
Experiment No.- 08

Aim:- Sequence Diagram for fast food billing system

Sequence diagram provides a graphical representation for how a task is accomplished by passing a sequence of messages among objects. These interactions define behavior as implemented by the fragments of the system structure.

Our system can be divided into two different sequence diagram as shown below:

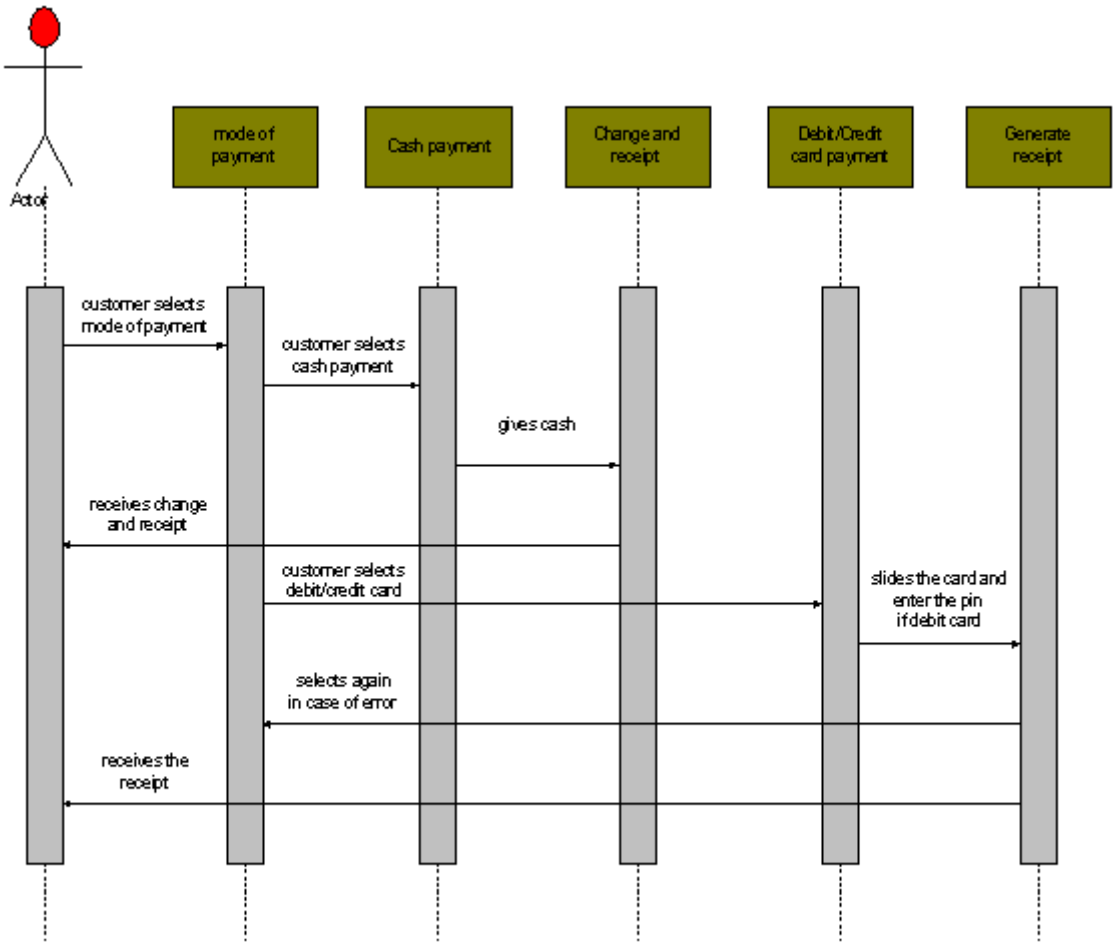
Sequence Diagrams for Placing an Order



Sequence Diagram for placing the Order

As seen in the diagram the customer enters the system by clicking begin button and selects his language choice. He then clicks on the menu button to see the menu items and then makes a selection. He then confirms his order by clicking on the confirm order button.

Sequence Diagrams for Making the Payment



Sequence Diagram for making the Payment

Customer has been prompted for the mode of payment. If he selects cash then he has to give cash to the cash collector and receive change, if any. He also receives a receipt from the cash collector. If the customer selects debit/credit card payment mode, he is asked again to select from debit or credit. If he selects debit card then he is asked a pin number or else he is asked to slide the card. After checking for the validity of the card the payment is made and the customer receives a receipt

Experiment No.- 09

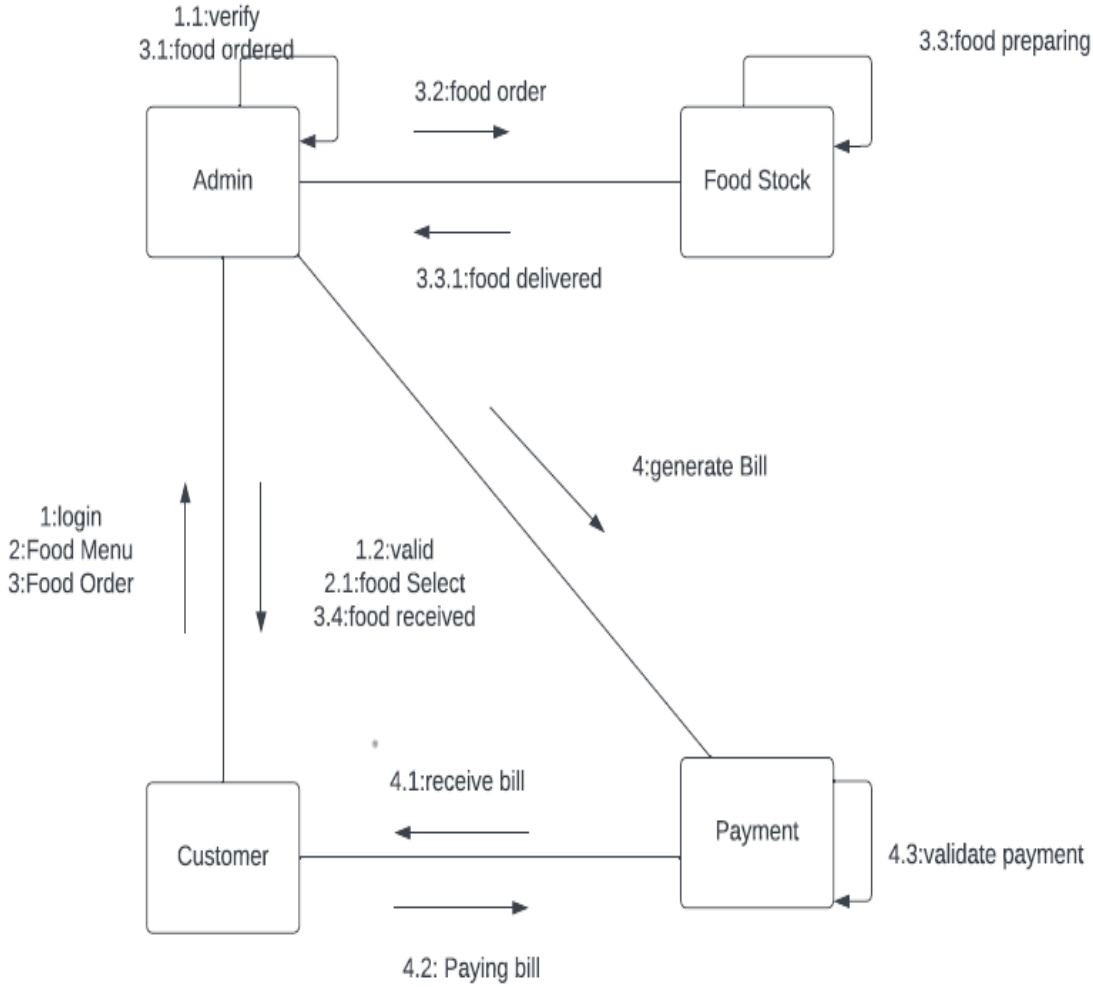
Aim:- Collboration Diagram for fast food billing system

A collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). These diagrams can be used to portray the dynamic behavior of a particular use case and define the role of each object.Collaboration diagrams are created by first identifying the structural elements required to carry out the functionality of an interaction. A model is then built using the relationships between those elements. Several vendors offer software for creating and editing collaboration diagrams.

Notations of a collaboration diagramA collaboration diagram resembles a flowchart that portrays the roles, functionality and behavior of individual objects as well as the overall operation of the system in real time. The four major components of a collaboration diagram are:

1. Objects- Objects are shown as rectangles with naming labels inside. The naming label follows the convention of object name: class name. If an object has a property or state that specifically influences the collaboration, this should also be noted.
2. Actors- Actors are instances that invoke the interaction in the diagram. Each actor has a name and a role, with one actor initiating the entire use case.
3. Links- Links connect objects with actors and are depicted using a solid line between two elements. Each link is an instance where messages can be sent.
4. messages- Messages between objects are shown as a labeled arrow placed near a link. These messages are communications between objects that convey information about the activity and can include the sequence number.

The most important objects are placed in the center of the diagram, with all other participating objects branching off. After all objects are placed, links and messages should be added in between.



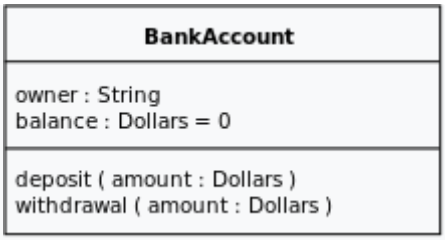
Collaboration Diagram of fast food billing system

Experiment No.- 10

Aim:- Class Diagram for fast food billing system

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling, translating the models into programming code. Class diagrams can also be used for data modeling.[1] 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:

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



Classes of Fast Food Corner Portal Class Diagram:

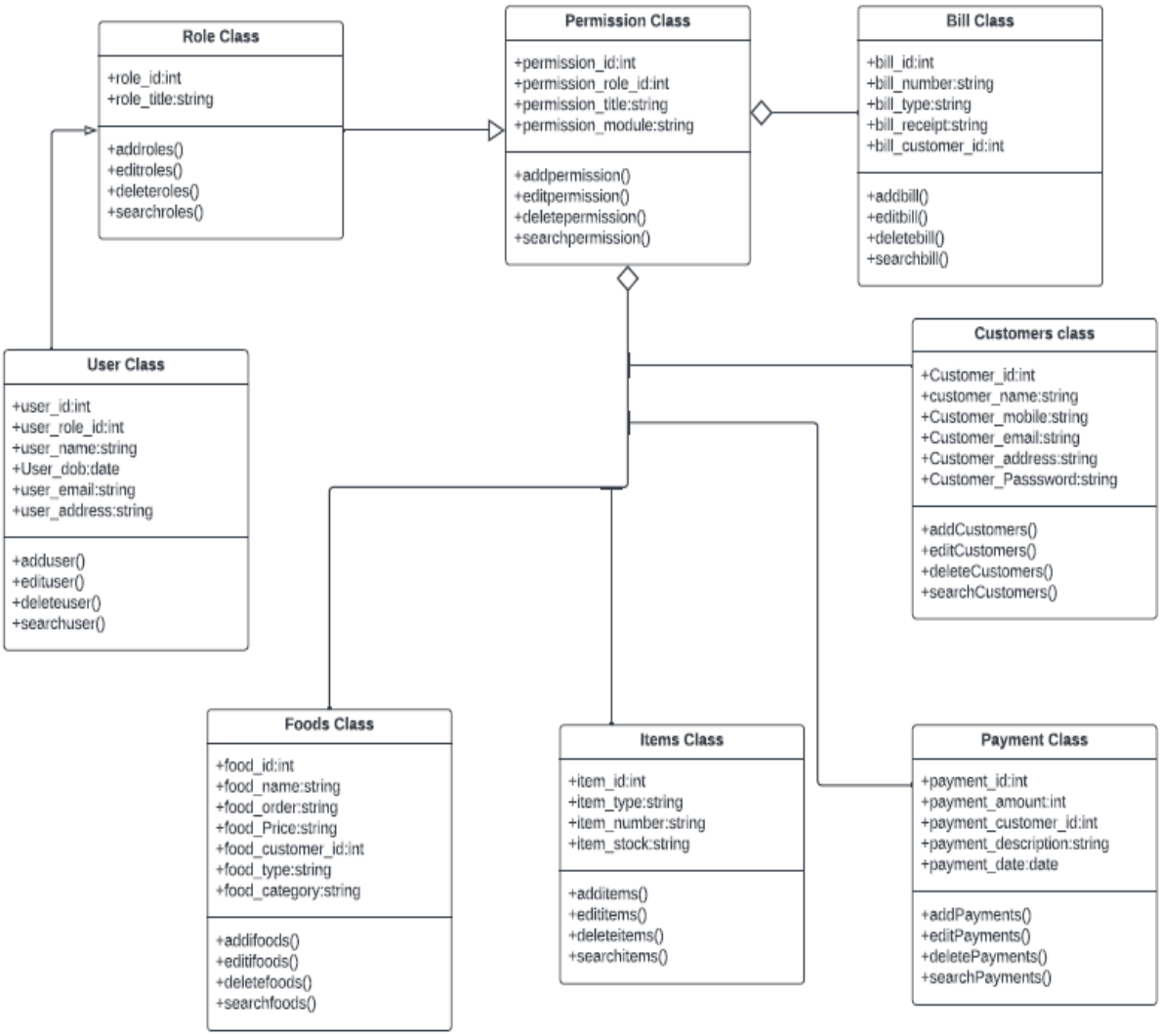
- **Foods Class** : Manage all the operations of Foods
- **Items Class** : Manage all the operations of Items
- **Payments Class** : Manage all the operations of Payments
- **Receipts Class** : Manage all the operations of Receipts
- **Customers Class** : Manage all the operations of Customers
- **Bill Class** : Manage all the operations of Bill

Classes and their attributes of Fast Food Corner Portal Class Diagram:

- **Foods Attributes** : food_id, food_customer_id, food_order, food_category, food_price, food_name, food_type, food_description
- **Items Attributes** : item_id, item_stocks, item_number, item_type, item_description
- **Payments Attributes** : payment_id, payment_customer_id, payment_date, payment_amount, payment_description
- **Receipts Attributes** : receipt_type, receipt_date, receipt_description, receipt_maintenance bill_number
- **Customers Attributes** : customer_id, customer_name, customer_mobile, customer_email, customer_username, customer_password, customer_address
- **Bill Attributes** : bill_id, bill_customer_id, bill_number, bill_type, bill_receipt, bill_description

Classes and their methods of Fast Food Corner Portal Class Diagram:

- **Foods Methods** : addFoods(), editFoods(), deleteFoods(), updateFoods(), saveFoods(), searchFoods()
- **Items Methods** : addItems(), editItems(), deleteItems(), updateItems(), saveItems(), searchItems()
- **Payments Methods** : addPayments(), editPayments(), deletePayments(), updatePayments(), savePayments(), searchPayments()
- **Receipts Methods** : addReceipts(), editReceipts(), deleteReceipts(), updateReceipts(), saveReceipts(), searchReceipts()
- **Customers Methods** : addCustomers(), editCustomers(), deleteCustomers(), updateCustomers(), saveCustomers(), searchCustomers()
- **Bill Methods** : addBill(), editBill(), deleteBill(), updateBill(), saveBill(), searchBill()



Class Diagram of fast food Billing System