EXP. NO: 5	SEQUENCE AND COLLABORATION DIAGRAM – STOCK INVENTORY MANAGEMENT

Aim:

To represent interactions in the Stock Inventory Management system using UML Sequence and Collaboration Diagrams.

Sequence Diagram:

A UML Sequence Diagram is a type of interaction diagram that visually represents the flow of messages between objects or components in a system over time. It shows how objects interact in a specific sequence to achieve a particular functionality, emphasizing the order in which events occur. Each object involved in the interaction is represented by a lifeline, with messages exchanged between them depicted as horizontal arrows. These messages may include method calls, responses, or signals. The diagram helps to illustrate dynamic behavior in scenarios such as system workflows, business processes, or software features, making it useful for designing and understanding system interactions.

Purpose of Sequence Diagram:

1. **Visualizing interactions**: Displays how objects or components communicate in a specific sequence to fulfil a task.
2. **Understanding system behaviour**: Helps in analysing the dynamic behaviour of a system over time by showing message flows.
3. **Design validation**: Ensures that the interaction logic between objects aligns with the expected design.
4. **Identifying responsibilities**: Clarifies which objects are responsible for which actions or messages during a process.
5. **Improving communication**: Acts as a communication tool for developers, analysts, and stakeholders to better understand system interactions.
6. **Facilitating troubleshooting**: Assists in identifying potential issues or inefficiencies in message exchanges and process flows.
7. **Documenting use cases**: Provides a clear view of the interactions for specific use cases or scenarios, making it useful for system documentation.

Components of Sequence Diagram:

1. Lifeline

- **Description:** A vertical dashed line that represents the lifespan of an object or component during the interaction.
- **Explanation:** Each lifeline corresponds to a specific object or participant in the interaction. The name of the object is written at the top, with its timeline extending downwards to represent its existence throughout the interaction.

2. Actor

- **Description:** A stick figure or a labeled rectangle that represents an external user or system interacting with the system.
- **Explanation:** Actors initiate or receive interactions from the system. They can be

human users or other systems that are not part of the internal software.

3. Message

- **Description:** A horizontal arrow that shows communication between lifelines.
- **Explanation:** Represents the communication between objects, which can be in the form of method calls, signals, or information exchanges. There are different types of messages:
  - Synchronous message (solid arrow with filled head): The sender waits for the receiver to process the message.
  - Asynchronous message (solid arrow with open head): The sender does not wait for the receiver to process the message.

4. Activation bar (Execution specification)

- **Description:** A vertical rectangle drawn on top of a lifeline.
- **Explanation:** It represents the time period during which an object is performing an operation. It highlights the duration of the action initiated by a message.

5. Return Message

- **Description:** A dashed horizontal arrow pointing back to the sender.
- **Explanation:** Represents the return of control or data from the receiving object back to the sender after processing a synchronous message.

6. Fragment

- **Description:** A box that groups a part of the interaction flow.
- **Explanation:** It is used to represent complex behaviors like loops, conditions (alternatives), or parallel processing. Examples include:
  - Alt: Represents alternative flows based on conditions (if-else structure).
  - Loop: Denotes a repeated sequence of interactions (loops).
  - Opt: An optional interaction that occurs only if a condition is met.

7. Self-message

- **Description:** An arrow that loops from and to the same lifeline.
- **Explanation:** Represents an object calling its own method or performing an internal operation.

8. Combined Fragment

- **Description:** A frame that surrounds multiple messages.
- **Explanation:** Used to model control structures such as branching, looping, or parallel executions. It helps manage different sequences based on specific conditions or processes.

9. Destroy Message

- **Description:** A message that ends a lifeline with a cross at the bottom.
- **Explanation:** Represents the termination or destruction of an object at a specific point in the interaction. The object's lifeline ends after this.

Sequence Diagram for Stock Inventory Management System

Components:

1. Actor – User:

The user can be the Admin, Inventory Manager, Cashier or an Executive Manager. The user's interactions with the system are defined along the full lifeline of the user.

2. System Classes:

- **Registration and Login**
  Manages user authentication and authorization. This class handles the registration of new users by collecting and storing their credentials and user roles, and facilitates user login by verifying credentials with stored data. It also manages password encryption and security protocols.

  The admin interacts with this class to register new employees. All other users interact with it to login and gain access to the system. Its lifeline only lasts until the user logs in or registers.

- **Stock In**
  Handles the process of adding new stock to the inventory. It records details such as product ID, quantity, supplier information, and date of stock entry. It updates the inventory database and logs the stock-in events for future reference.

  The inventory manager interacts with the stock in module to add items to the inventory database. It is also viewed by managers and executive managers to get data about the available stock.

- **Stock Out**
  Manages the removal or sale of stock from the inventory. This class updates inventory levels when products are sold or dispatched, recording details such as product ID, quantity, customer or recipient info, and date of the transaction.

  The inventory manager interacts with this class to add items to the stock out database, as a result of sales or other reasons. Additionally, the billing class interacts with this class to automatically add sold items to the stock out data during its lifeline.

- **Dashboard**
  Provides an overview of the system's current state and key performance indicators (KPIs). It aggregates data like total stock, recent stock movements, low stock warnings, sales reports, and user activities to present it in an easy-to-understand interface for users.

  This is mainly used by the admin or the executive manager to get an overview of stock movement and sales information. It retrieves data from stock-in, stock-out and billing classes during its lifeline.

- **Billing**
  Manages the billing process by generating invoices, tracking payments, and maintaining a record of sales transactions. This class calculates total costs, including taxes and discounts, and links sales data with stock out events to ensure inventory accuracy.

  This is used by the cashier during billing and sales. It generates a bill and sends sales information to the dashboard during its lifeline.

Interactions:

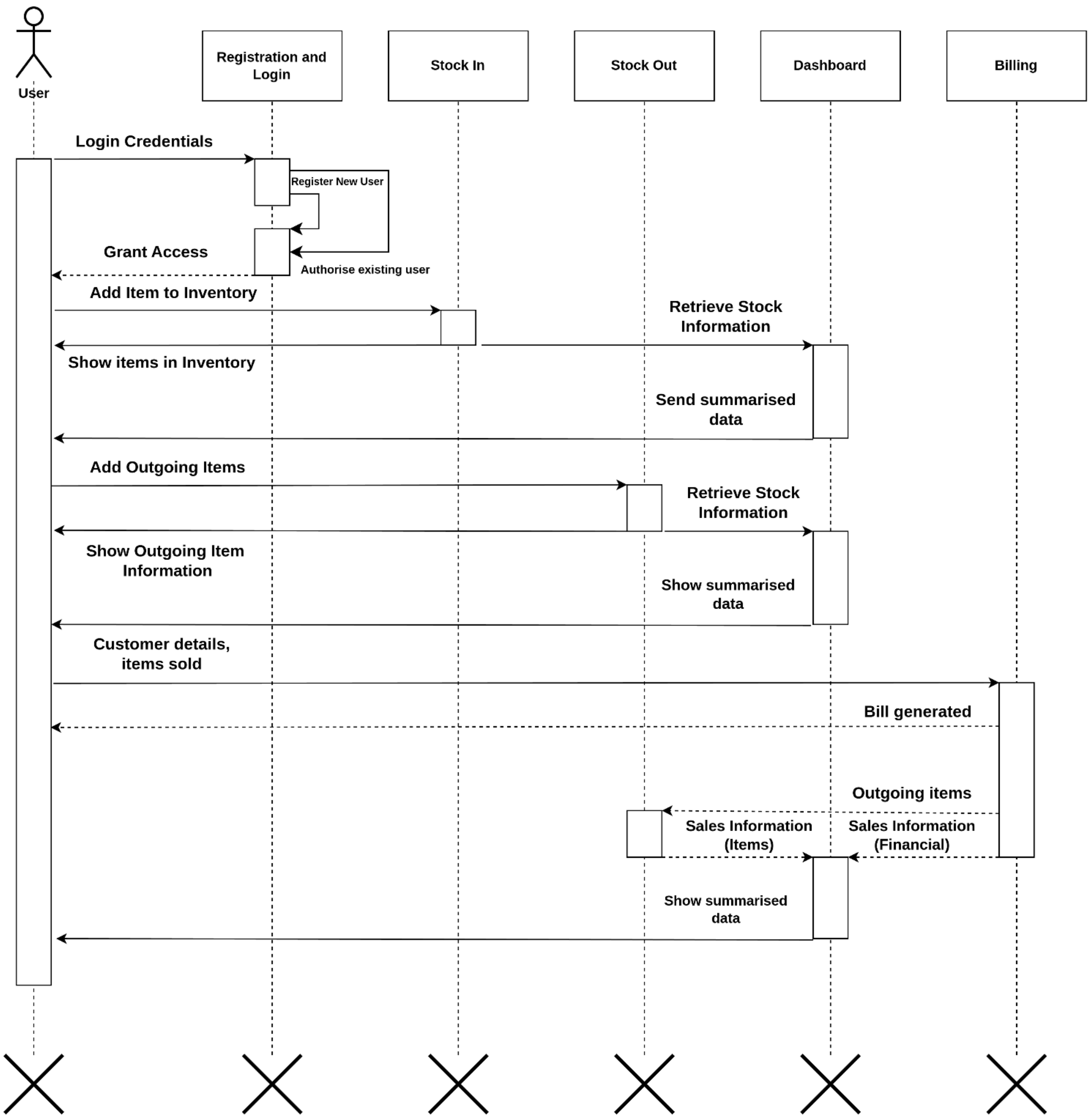
User Interactions:

- **Registration / Login:** To register a new user or login to the system. The login system grants access to the user.
- **Stock In:** To add stock to the inventory database and view available stock.
- **Stock Out:** To add stock to the stock out database and view sales and outgoing stock information.
- **Dashboard:** To view consolidated sales information, profits and other stock information.
- **Billing:** To bill items during sale and generate a bill for the customer.

Internal Interactions:

- **Stock In – Dashboard:** Stock In sends inventory information to the dashboard during its lifeline. The dashboard processes and displays a summary of that information.
- **Stock Out – Dashboard:** Stock Out sends outgoing stock information to the dashboard during its lifeline. The dashboard processes and displays a summary of that information.
- **Billing – Stock Out:** Billing module sends information of sold items so that they can be tracked in Stock Out. The Stock Out module removes the items from inventory and adds them to outgoing stock database for reports.
- **Billing – Dashboard:** Billing module sends sales information to the Dashboard during its lifeline. The dashboard processes this information and displays sales information such as profits and items sold.

UML Sequence Diagram – Stock Inventory Management



Collaboration Diagram:

A Collaboration Diagram (also known as a Communication Diagram) is a type of interaction diagram in UML that focuses on the structural organization of objects and the flow of messages between them to perform a specific task. Unlike sequence diagrams, which emphasize the order of interactions, collaboration diagrams highlight the relationships and interactions between objects by showing how messages are passed within a system. Objects are represented as boxes, and the communication between them is depicted by labeled arrows that indicate the messages exchanged. The diagram also numbers the interactions to show the sequence of events. Collaboration diagrams are particularly useful for understanding how different objects collaborate to achieve a common goal, offering a more structural view of the system's functionality.

Purpose of Collaboration Diagrams:

1. **Visualizing object interactions:** Illustrates how objects within a system collaborate to accomplish a specific task or process.
2. **Understanding system structure:** Emphasizes the structural relationships and links between objects, making it easier to understand the organization of components.
3. **Tracing message flow:** Shows the flow of messages between objects and the sequence of interactions, clarifying the communication pathways within the system.
4. **Supporting system design:** Helps designers and developers identify how different objects and components interact in a system, assisting in architectural and component design.
5. **Analysing responsibilities:** Clarifies which objects are responsible for initiating and responding to messages, helping in responsibility assignment.
6. **Optimizing collaboration:** Aids in detecting inefficiencies or redundancies in object interactions, allowing for optimization of communication processes.

Components of Collaboration Diagram:

1. Objects (Participants)

- **Description**: Represented by rectangular boxes, each object is an instance of a class that participates in the interaction.
- **Explanation**: These are the entities (objects) involved in the collaboration. Each object is labelled with its class name and sometimes its object name, showing its role in the interaction.

2. Links (Associations)

- **Description**: Solid lines that connect the objects.
- **Explanation**: Represent the relationships or communication paths between objects. A link shows that two objects can communicate with each other, even if the message is not currently being passed.

3. Messages

- **Description**: Labeled arrows attached to the links between objects, usually numbered sequentially.
- **Explanation**: Represent the communication between objects in the form of method calls, signals, or data transfer. The messages are numbered to indicate the sequence of the interaction. Each message label often includes the method or action being invoked.

4. Sequence Numbers

- **Description**: Numbers or decimal numbers attached to the message arrows.
- **Explanation**: Indicate the order in which the messages are passed between objects. For example, "1" would be the first message sent, and "1.1" would be a sub-message (nested interaction) within the first message sequence.

5. Object Roles

- **Description**: The labels that appear next to objects, often showing an object's role in the interaction.
- **Explanation**: These labels specify what role or responsibility the object plays in the interaction. For example, one object might serve as a "sender" while another is a "receiver."

6. Self-Messages

- **Description**: Arrows that loop back to the same object.
- **Explanation**: Represent an object invoking a method or performing an internal operation on itself during the collaboration.

7. Conditional Messages

- **Description**: Messages that are only sent when certain conditions are met.
- **Explanation**: The condition for sending the message is typically written inside brackets (e.g., [condition]), representing a guard condition that must be true for the message to be sent.

8. Multiplicity

- **Description**: Indicates how many instances of an object or link exist.
- **Explanation**: Represented by numbers at the ends of links, multiplicity shows whether the interaction involves one-to-one, one-to-many, or many-to-many relationships between objects.

These components together help visualize how objects are linked and how they interact through the exchange of messages to fulfil a specific use case or functionality within a system. The focus is on understanding the structure of object collaboration rather than just the sequence of events.

Collaboration Diagram for Stock Inventory Management System

Components:

1. Actors:

- Admin / Manager: Has access to all objects
- Executive Manager: Has access to the Dashboard
- Inventory Manager: Has access to the Stock In and Stock Out Objects
- Cashier: Has access to Billing
- Customer: Receives bill from Billing
- User: Generalisation of all other actors, represented as such for the purpose of registration and login.

2. Objects:

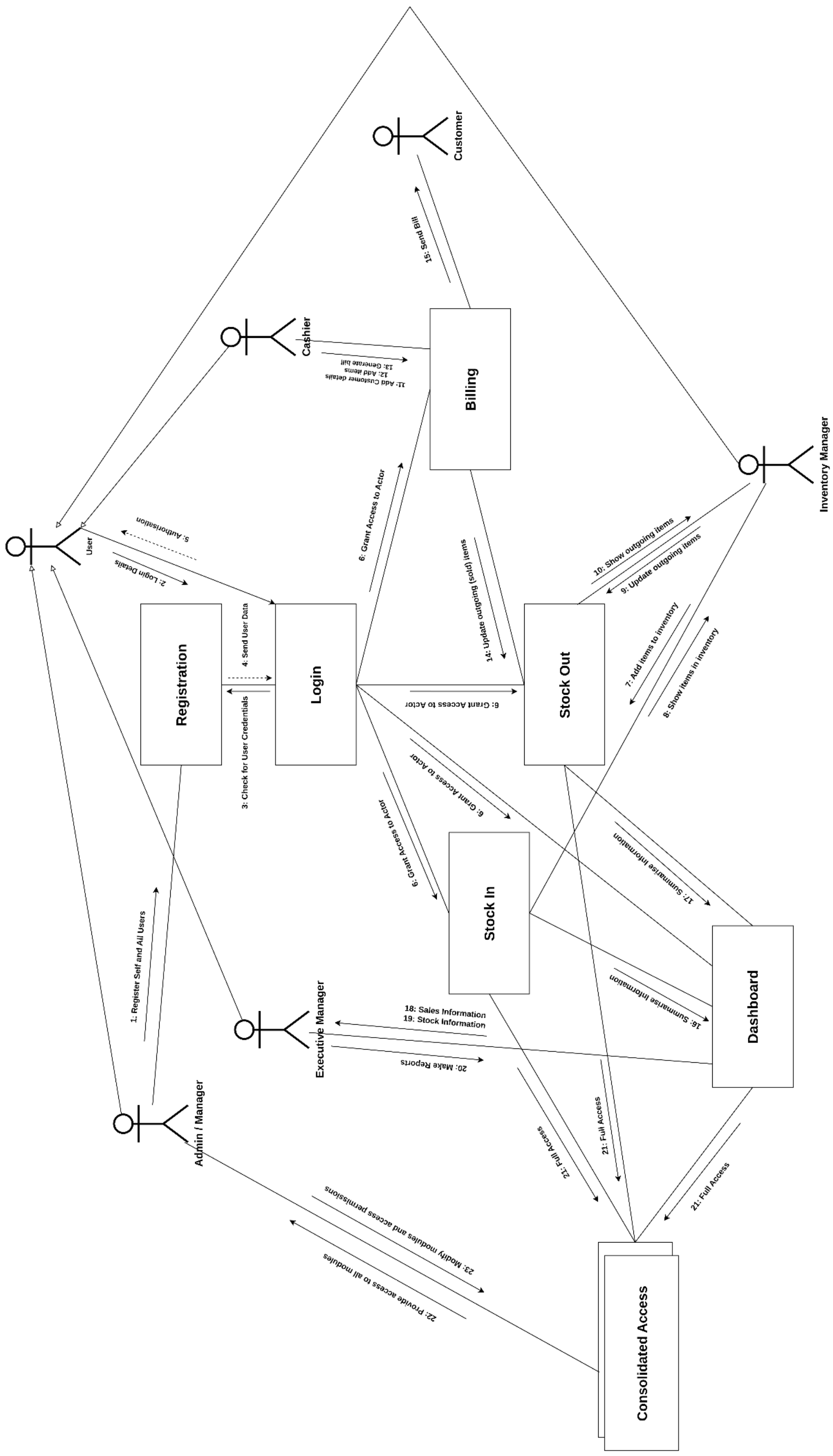
- **Registration:** Allows registration of new users (actors)
- **Login:** Authorises users
- **Stock In:** Manages inventory database and incoming items
- **Stock Out:** Manages outgoing items and sold items
- **Dashboard:** Provides summary of information obtained from Stock In, Stock Out and Billing
- **Billing:** Allows cashier to handle sales and generates bills and sales reports
- **Consolidated Access:** Gives access to Stock In, Stock Out and Dashboard to administrators and managers

3. Message Flow:

1. Register Self and All Users: The primary administrator registers himself and other users to the system.
2. Login Details: User logs in to the system by passing their credentials.
3. Check for User Credentials: The login object checks with the registration database for the user.
4. Send user data: The registration object confirms existence of user.
5. Authorisation: User is allowed to access the system

6. Grant Access to Actor: The login system grants access to other objects depending on the user's privileges.
7. Add Items to Inventory: Inventory Manager adds items to the inventory database using Stock In object.
8. Show Items in Inventory: Stock In object shows information about available items.
9. Update outgoing items: Inventory Manager updates outgoing items using the Stock Out object.
10. Show outgoing items: Stock Out object shows Inventory Manager outgoing items.
11. Add customer details: Cashier enters customer details through the Billing object.
12. Add items: Cashier adds items to be sold using the Billing object.
13. Generate bill: Cashier generates bill of sale using the Billing object
14. Update outgoing items (sold): Billing object updates outgoing items by sending a message to the Stock Out object.
15. Send Bill: Billing object sends Customer their bill.
16. and 17. Summarise Information: Stock In and Stock Out objects send inventory information to the Dashboard.
18. Sales Information: Dashboard shows sales information to Executive Manager.
19. Stock Information: Dashboard shows stock flow information to Executive Manager.
20. Make reports: Executive Manager can generate reports using the Dashboard.
21. Full Access: Stock In, Stock Out, and Dashboard objects are consolidated and allowed access to by the "Consolidated Access" object. It gives administrators complete access to these modules.
22. Provide access to all modules: The Consolidated Access object provides access to the above-mentioned object for administrators.
23. Modify modules and access permissions: The administrator manages modules and access to modules.

UML Collaboration Diagram – Stock Inventory Management System



Description	Allotted Marks	Obtained Marks
Preparation	**20**	
Design / Implementation	**20**	
Viva	**15**	
Output	**10**	
Record	**10**	
Total	**75**	

RESULT:

Thus, the development of Sequence Diagram and Collaboration Diagram for Stock Inventory Management System has been completed successfully.