EXP. NO: 6	STATE AND ACTIVITY DIAGRAM – STOCK INVENTORY MANAGEMENT

Aim:

To represent interactions in the Stock Inventory Management system using UML State and Activity Diagrams.

State Diagram:

A State Diagram (also known as a State Machine Diagram or Statechart Diagram) is a behavioral UML diagram that depicts the different states an object or system can be in, and the transitions between those states based on events or conditions. It is used to model the dynamic behavior of a system by showing how an object transitions from one state to another in response to external stimuli. This type of diagram is particularly useful for modeling the lifecycle of an object or system where its behavior depends on its state.

Purpose of a State Diagram:

1. **Modeling dynamic behavior**: Captures how an object or system changes its state in response to events over time.
2. **Understanding object lifecycles**: Provides a clear view of how an object's state evolves from creation to termination.
3. **System design**: Helps in designing state-dependent systems, such as embedded systems, user interfaces, or real-time systems.
4. **Clarifying complex logic**: Simplifies complex decision-making processes and control flow by visualizing state transitions.
5. **Testing and validation**: Aids in identifying all possible states and transitions, improving system testing and validation.
6. **Documenting object behavior**: Acts as a formal specification for how an object behaves throughout its lifecycle, useful in both design and maintenance.
7. **Supporting event-driven programming**: Useful for systems that operate based on events or conditions, helping developers understand when and why state changes occur.

Components of a State Diagram:

1. **State**
   - **Description**: Represented by a rounded rectangle.
   - **Explanation**: Represents a condition or situation in the lifecycle of an object during which it satisfies some condition, performs some activity, or waits for an event. A state can also include entry/exit actions or internal activities.
2. **Initial State (Start State)**
   - **Description**: Represented by a filled black circle.

- **Explanation**: Marks the beginning of the state machine. It shows where the object's state lifecycle starts.

3. **Final State**

- **Description**: Represented by a circle with a border and a filled black circle inside it.
- **Explanation**: Marks the end of the state machine. It indicates that the object or system has completed its lifecycle and will no longer transition to any other state.

4. **Transition**

- **Description**: Represented by a solid arrow connecting two states.
- **Explanation**: Represents the movement or change from one state to another in response to an event or condition. Transitions may also have triggers, guards (conditions that must be true for the transition to occur), and actions (activities performed during the transition).

5. **Event**

- **Description**: Labeled on a transition arrow.
- **Explanation**: Represents an external or internal occurrence that triggers a transition from one state to another. Events can be user inputs, signals, messages, or conditions that arise during the process.

6. **Guard Condition**

- **Description**: A condition placed on a transition, typically in square brackets (e.g., [condition]).
- **Explanation**: Specifies a condition that must be true for the transition to occur. If the guard condition is false, the transition does not happen, even if the event is triggered.

7. **Action**

- **Description**: Labeled on the transition line, often after a forward slash (/).
- **Explanation**: Represents an activity or operation that is executed as a result of a transition. Actions are performed immediately after the transition occurs.

8. **Entry/Exit Actions**

- **Description**: Represented by labels like entry or exit within a state.
- **Explanation**: Specifies actions that are executed when an object enters or exits a particular state. These actions can handle setup or cleanup activities associated with state transitions.

9. **Composite State (Nested State)**
   - **Description**: A state that contains sub-states, depicted by a state with a nested state diagram inside.
   - **Explanation**: Represents a state that is decomposed into multiple sub-states. Composite states are useful for modeling complex states where multiple states need to be managed as part of the object's lifecycle.

10. **Self-transition**
    - **Description**: An arrow that loops back to the same state.
    - **Explanation**: Represents a situation where an event causes the object to remain in the same state, but some action is performed during the transition.

11. **Choice Pseudostate**
    - **Description**: Represented by a diamond shape (similar to a decision node).
    - **Explanation**: Used to model multiple alternative transitions from a single state based on conditions. It evaluates conditions and directs the flow to different subsequent states.

12. **History State**
    - **Description**: Represented by an encircled 'H' inside the state.
    - **Explanation**: Represents a situation where the system remembers the last active sub-state when returning to a composite state. This is useful for processes that can resume from where they left off.

By using these components, a State Diagram effectively models how an object or system transitions between different states, capturing its behaviour and the triggers that drive those transitions. It's a powerful tool for designing event-driven systems and understanding how processes evolve over time.

State Diagram for Stock Inventory Management System

State Diagram – Warehouse Staff:

1. Initial State: Visit Site

- The process starts with a black dot, which represents the initial state.
- The warehouse staff visits the site and decides if they want to proceed with their tasks.

2. Login Process

- The next step directs the user to the Login page.
- Login verification occurs with the condition: if(login == true).
  - If the login is successful, the system grants access based on the user's role.
  - If unsuccessful, the process would typically stop, though it's not depicted here explicitly.

3. Role-Based Access

Once logged in, the system checks the role of the user and directs them to different modules:

- **[Role == Inventory Manager]**
  - The inventory manager is directed to the Stock Information Manager page.
  - They may manage stock levels, add or remove stock, and update stock records.
- **[Role == Cashier]**
  - The cashier is taken to the Billing and Stock Information page.
  - This page combines billing and stock data, likely allowing the cashier to manage sales and view stock availability.
- [**Role == Executive Manager]**
  - The executive manager is given access to the Dashboard.
  - The dashboard typically offers high-level metrics, reports, and performance data to monitor the warehouse operations.

4. Completing Work

- All roles converge at the Complete Work state after finishing their tasks.
  - This indicates that the workflow for any staff member ends at the same point, regardless of their role.

5. Logout

- The user can logout at the end of their session, represented by the arrow pointing back to the black dot, which marks the termination of the process.

6. Red Line (Synchronization Bar)

- The red line below the role-specific modules represents a synchronization bar (in UML).

- This ensures that, no matter which role the user has, all processes will converge at a common point before allowing the work to be marked complete.

State Diagram – Customer:

1. Initial State: Visit Site

- The process begins with a black dot, representing the initial state.
- The customer visits the site to explore available products.

2. Viewing Products

- After entering the site, the customer wants to view the products and selects this option.
- The system directs them to the "View the Products" page, where they can browse the inventory.

3. Making a Payment

- Once the customer is satisfied with their selection, they proceed to make the payment.
- This triggers the process for generating the bill for the transaction.

4. Receiving the Bill PDF

- After a successful payment, the system provides the customer with a PDF of the bill.
    - This ensures the customer has a formal receipt or proof of purchase.

5. End State: Logout/Exit

- The process ends with the red-circled black dot, indicating the final state.
- This shows that the customer’s interaction with the site has completed.

State Diagram – Administrator:

1. Initial State: Visit Site

- The process starts with a black dot, representing the initial state.
- The admin visits the site to manage various operations.

2. Login Process

- The admin decides to proceed and is directed to the Login page.
- The system verifies the credentials: if [login is successful].
- If the credentials are correct, the system shows available options for the admin.

3. Show Options (Admin Tasks)

After successful login, the system presents the admin with three options:

1. **Register Workers**
    - If the admin selects the option to add a worker, they are directed to the Register Workers page to input and register new staff members.
2. **Stock Details**
    - If the admin selects view stock details, they are directed to the Stock Details page, where they can check or manage inventory records.
3. **View Dashboard**
    - This option takes the admin to the Dashboard, where they can access high-level statistics and performance metrics.

4. Processing Tasks

- After completing tasks in any of the selected modules (registering workers, viewing stock details, or accessing the dashboard), the process moves to the synchronization bar (represented by the red line).
    - This ensures that, regardless of the task, all processes converge at the same point before the next decision.

5. Decision Point: Want to Continue?

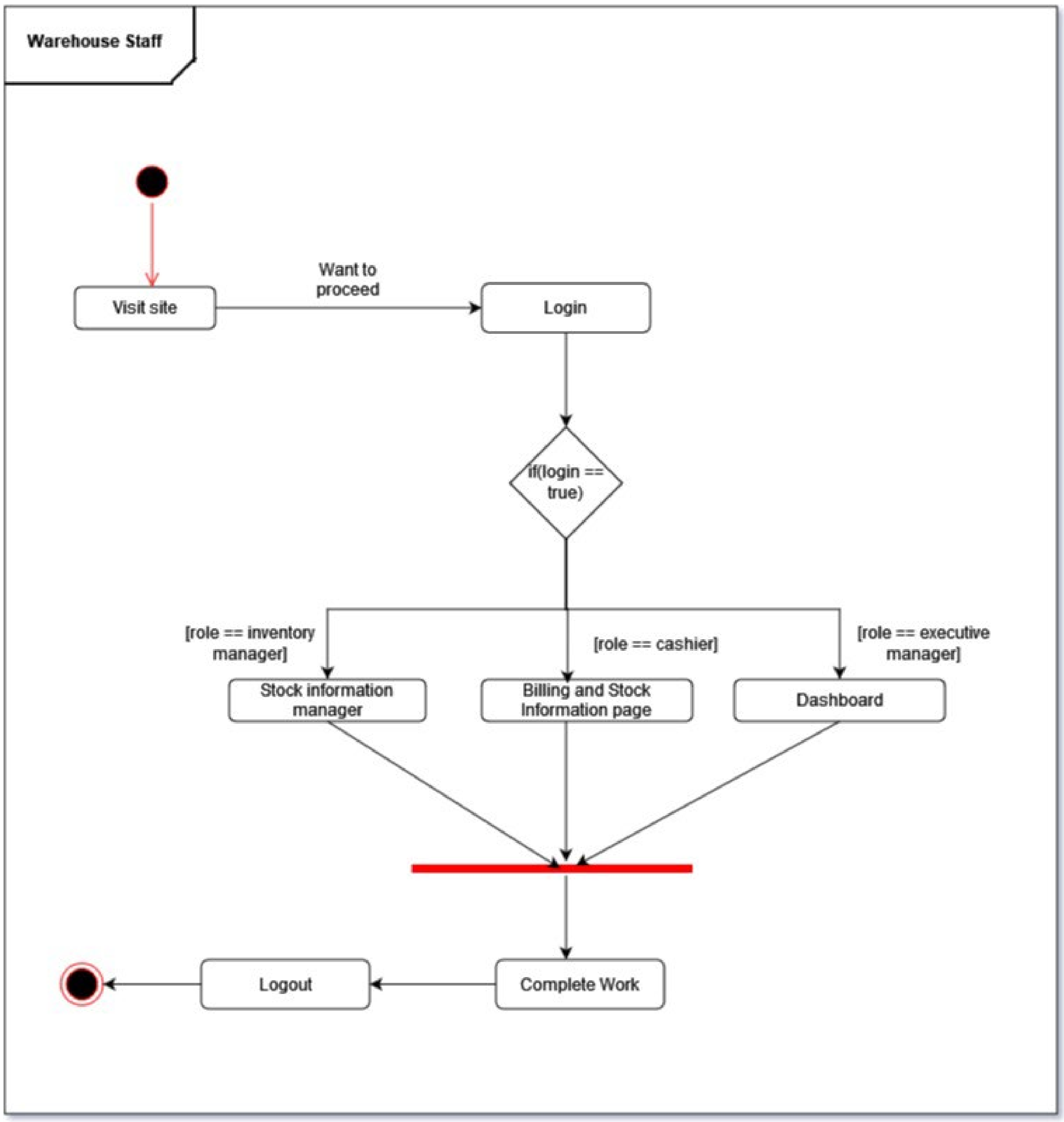
- After processing the task, the system asks the admin: "Want to continue?"
    - If True, the system loops back to Show Options, allowing the admin to perform another task.
    - If False, the admin proceeds to Logout.

6. End State: Logout

- When the admin logs out, the process ends, represented by the red-circled black dot as the final state.

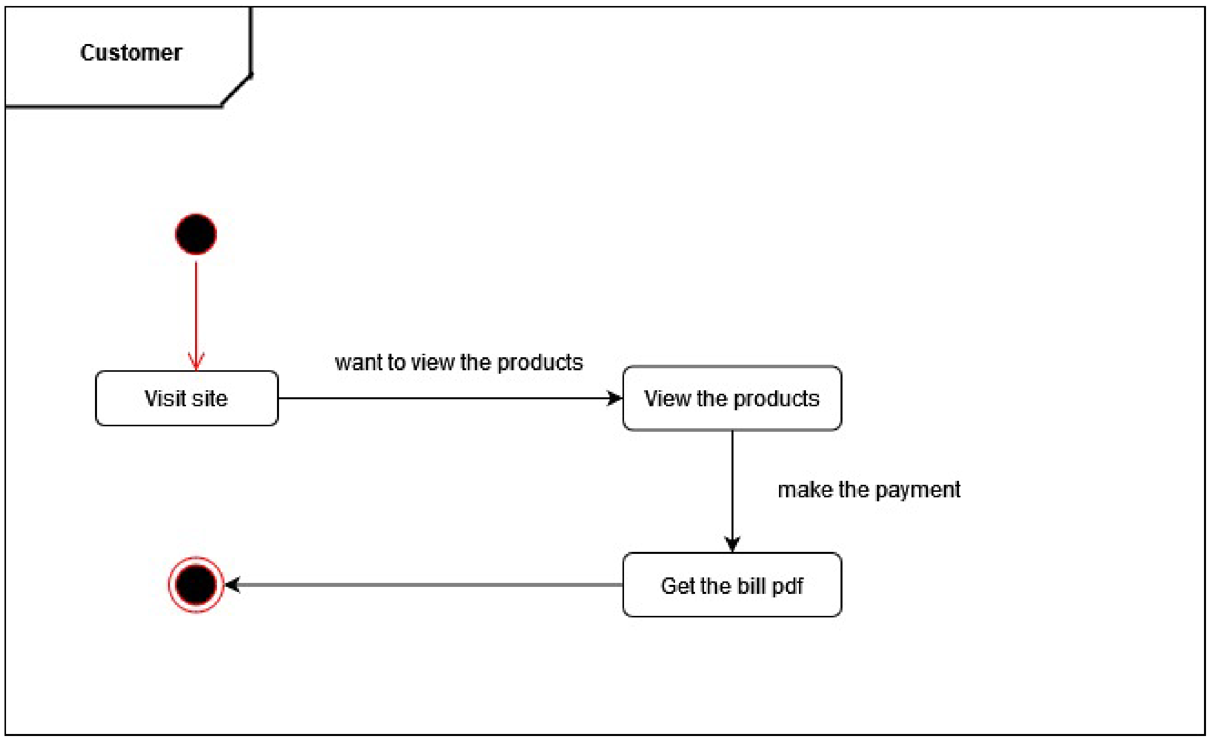
UML State Diagrams

1. Warehouse Staff

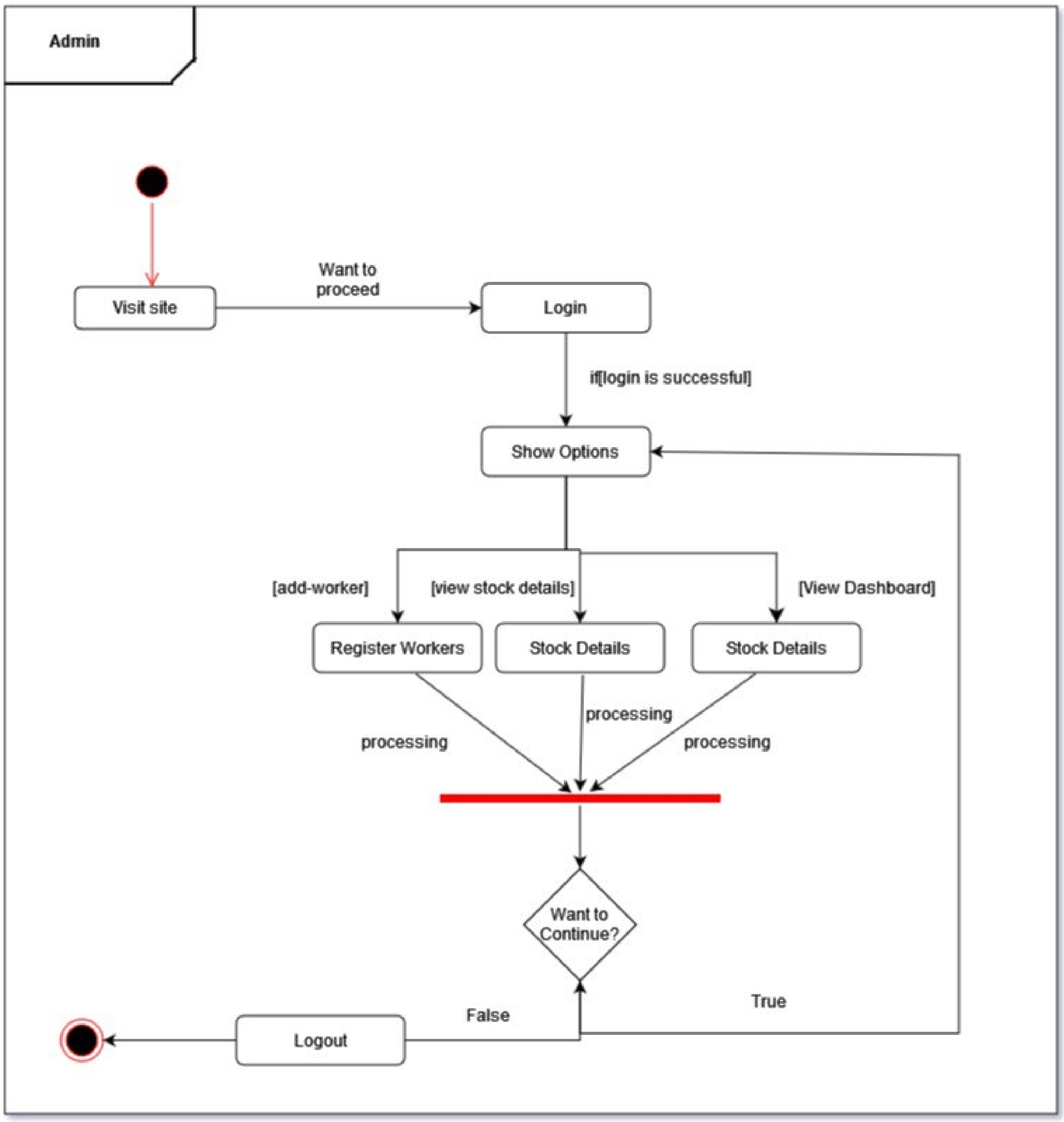


Warehouse Staff
Visit site
Want to proceed
Login
if(login == true)
[role == inventory manager]
[role == cashier]
[role == executive manager]
Stock information manager
Billing and Stock Information page
Dashboard
Complete Work
Logout

2. Customer


Customer
want to view the products
Visit site
View the products
make the payment
Get the bill pdf

3. Administrator:



Activity Diagram:

A UML Activity Diagram is a type of behavioral diagram within the Unified Modeling Language (UML) used to represent the flow of activities, actions, and decisions in a system or process. It models both sequential and concurrent activities, making it ideal for illustrating how a particular process or workflow unfolds. It can be used to document both high-level business processes and detailed logic flows in software systems.

Purpose of Activity Diagram:

The main purpose of an activity diagram is to describe the dynamic behavior of a system by focusing on how the control flows from one activity to another. It is especially helpful in scenarios where multiple processes or decisions are involved. Some key objectives include:

- **Visualizing workflows:** It offers a clear picture of business processes or software operations, ensuring everyone understands the sequence of tasks.

- **Analyzing complex processes:** By breaking down activities into smaller steps, it helps identify inefficiencies, bottlenecks, or errors.

- **Designing and documenting processes:** Developers use it to design workflows, and business analysts rely on it for process documentation.

- **Showing decision points and parallel operations:** It provides insight into conditional flows and parallel activities, demonstrating how different parts of a system can operate simultaneously or sequentially.

- **Facilitating communication:** Activity diagrams bridge the gap between non-technical and technical stakeholders, providing a common understanding of workflows.

Components of Activity Diagram:

Below are the essential components that make up an activity diagram:

1. **Initial Node (Start Point)**
    - Represented by a black circle, this node marks the starting point of the process or activity flow.

2. **Activity (Action) Nodes**
    - These are represented as rectangles with rounded corners. Each activity node shows a specific task or action to be performed within the workflow. Multiple actions can be linked to form the sequence of steps.

3. **Control Flow (Arrows)**
    - Arrows indicate the flow of control from one activity to another, showing the path of execution.

4. **Decision Node**
    - Represented by a diamond shape, this node models conditional logic where the flow can branch out based on a decision (e.g., a yes/no condition). Example: "Is the payment successful?" — If yes, proceed; if no, return to retry.

5. **Merge Node**
    - Another diamond-shaped node, which merges multiple alternative flows back into a single flow. It represents the convergence of paths after a decision point.

6. **Fork Node**
    - A thick horizontal or vertical bar that splits the flow into parallel activities. Each outgoing branch can execute independently and simultaneously.

7. **Join Node**
    - A thick bar used to synchronize multiple parallel flows back into a single path. This ensures that all concurrent activities are completed before moving to the next step.

8. **Swimlanes (Optional)**
    - Swimlanes are vertical or horizontal partitions used to group activities by responsible actors (e.g., different roles like admin, manager, or customer). They show who performs which tasks, helping to clarify role-specific responsibilities.

9. **Object Flow (Optional)**
    - Dotted arrows represent the flow of objects or data between activities (e.g., passing a document or product between two steps).

10. **Final Node (End Point)**
    - Represented by a black circle with an outer border, the final node indicates the completion of the process or activity.

11. **Synchronization Bar (Optional)**
    - This bar is used when activities in parallel need to sync before moving forward, ensuring all paths are completed before continuing.

Activity Diagrams for Stock Inventory Management System

Activity Diagram - Dashboard:

1. Start:

- The process begins at the start node (black circle). This represents the initialization of the stock inventory management system.

2. Input Username and Password:

- The user is prompted to enter their Username and Password.
- These inputs are sent for verification.

3. Verification of Credentials:

- The system checks the provided username and password against the stored credentials in the database. This ensures that the user attempting to log in exists and their credentials are correct.

4. Authorization Check:

- The system performs an authorization check to determine whether the user has the necessary permissions to access the system.
- If the user is not authorized, the process terminates here, and the user is denied access.
- If the user is authorized, the process continues.

5. Dashboard Access:

- Once authorized, the user gains access to the Dashboard. This is a control panel where the user can view important inventory information, track stock levels, or perform other system-related tasks.
- The diagram shows repeated access to the Dashboard, possibly indicating multiple dashboard views or different dashboard pages that the user can navigate through.

6. Generate the Report:

- The user is presented with the option to Generate a Report. This step may involve creating a stock report, inventory summary, or any other relevant data reports based on the system's available data.

7. Decision: Continue or Logout:

- After generating the report, the user is given a choice through a decision node:
    - If the user wants to continue using the system, they are taken back to the dashboard or can repeat tasks.
    - If the user chooses not to continue, the system proceeds to logout.

Logout:

- If the user chooses to stop, the system logs the user out, and the process terminates.

Activity Diagram – Customer and System Server:

Customer Side (Left Side):

1. **Start**:
   - The black circle at the top signifies the start of the process. The customer begins interacting with the system.

2. **View the Stock List**:
   - The customer is shown the available stock items in the system, and they can browse through this list.

3. **Add the Stocks to the Cart**:
   - The customer selects the desired stock items and adds them to the shopping cart for purchase.

4. **Payment Mode Selection**:
   - After adding items to the cart, the customer chooses a Payment Mode. This step involves selecting between cash or credit card as the payment method.

5. **Payment Methods**:
   - **Credit Card**:
     - If the customer selects credit card as the payment mode, the card is read by the system.
     - The system checks the card's validity. If the card is not valid, the transaction terminates.
     - If the card is valid, the system proceeds with the payment and moves to the next step.
   - **Cash**:
     - If the customer selects cash, they are prompted to enter the payment amount.

6. **Transaction Completion Check**:
   - The system checks whether the transaction is completed. This could involve ensuring payment was successful or verifying that all the necessary steps were followed.
   - If the transaction is completed successfully, the process moves forward. If not, it loops back for correction or retry.

Server Side (Right Side):

1. **Calculate the Bill**:
    - As the customer adds stock items to the cart, the server calculates the total bill for the items in the cart.

2. **Display the Bill**:
    - The system displays the calculated bill to the customer for review.

3. **Validate Card** (If Credit Card is Chosen):
    - If the customer chooses credit card payment, the system reads the card details and validates them. If the card is valid, the transaction continues; otherwise, the process halts.

4. **Update the Stock Data in the Database**:
    - Once the transaction is confirmed as complete (either through cash or a valid card payment), the server updates the stock data in the database to reflect the sold items, ensuring that the inventory remains accurate.

5. **Update the UI**:
    - After the stock data is updated in the database, the system updates the User Interface (UI), possibly reflecting the updated stock levels or notifying the customer of the successful transaction.

6. **End**:
    - The process ends after updating the UI, signified by the black circle at the bottom.

Activity Diagram – Stock In & Stock Out:

1. Start:
- The black circle at the top represents the start of the "Stock In & Stock Out" process.

2. Enter Username and Password:
- The user is required to enter both a **Username** and **Password**.
- These credentials are necessary to access the system.

3. Verification:
- The system verifies the entered username and password.
- If the credentials are incorrect, the process will not proceed. The diagram doesn't explicitly show the failure path, but it implicitly loops back to the login step.

4. View the Stock Details:
- Once the user is authenticated, they are allowed to view the stock details.
- This step provides an overview of the available stock items in the system.

5. Search for the Stock:
- The user initiates a search for a specific stock item to perform "Stock In" (adding inventory) or "Stock Out" (removing inventory).

6. Existence Check:
- The system checks whether the searched stock item **exists** in the database.
- If the stock item does **not exist**, the process terminates or loops back for another search.
- If the stock item **exists**, the process continues.

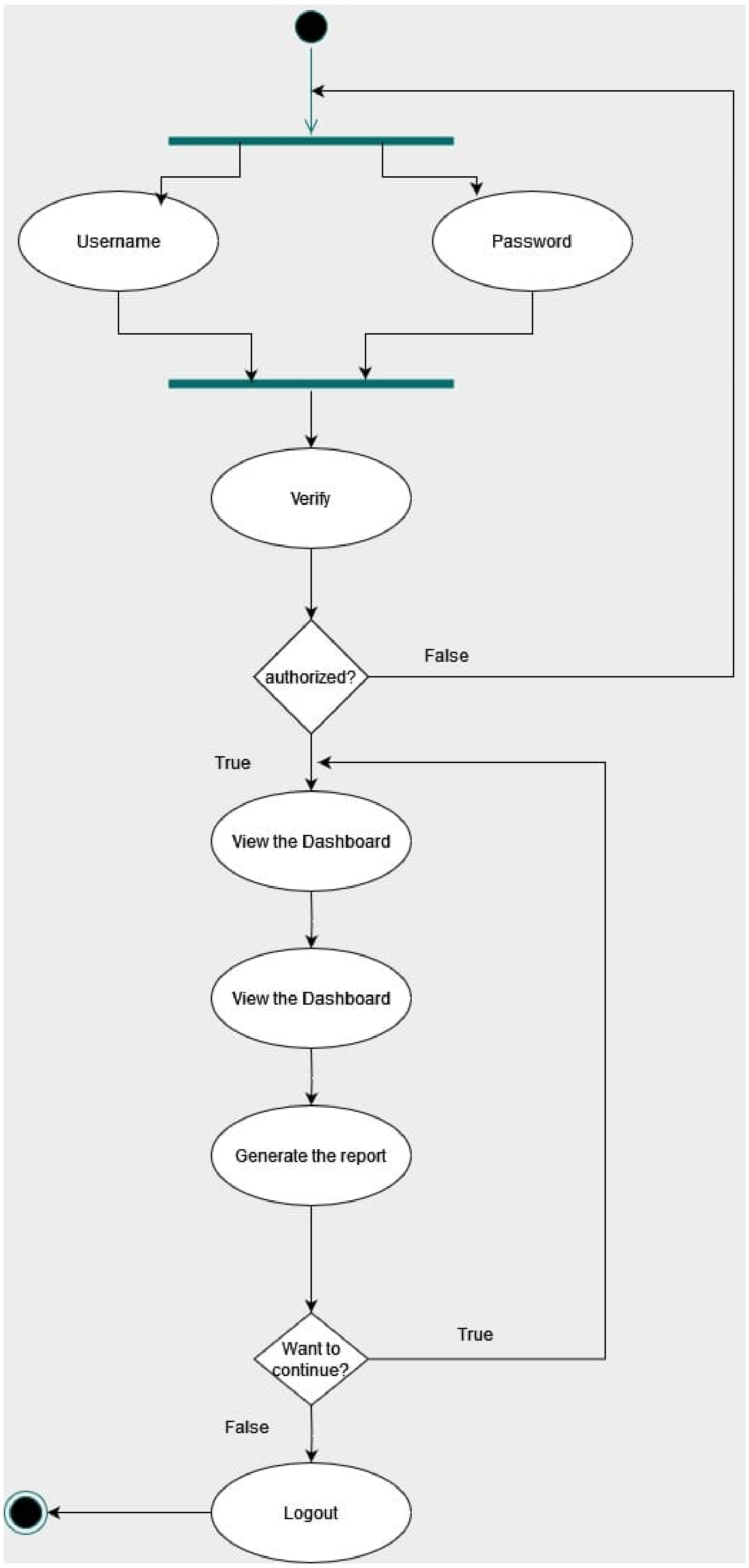
7. Update Stock Details:
- The system allows the user to update the stock details, specifically:
  - **Stock In**: Adding inventory to the system.
  - **Stock Out**: Removing inventory from the system.
- This step updates the stock quantities in the database to reflect current stock levels.

8. Decision: Continue?
- The system asks the user if they wish to continue with additional stock management tasks.
- If the user selects **true**, the process loops back, allowing the user to perform more operations.
- If the user selects **false**, the process moves to logout.

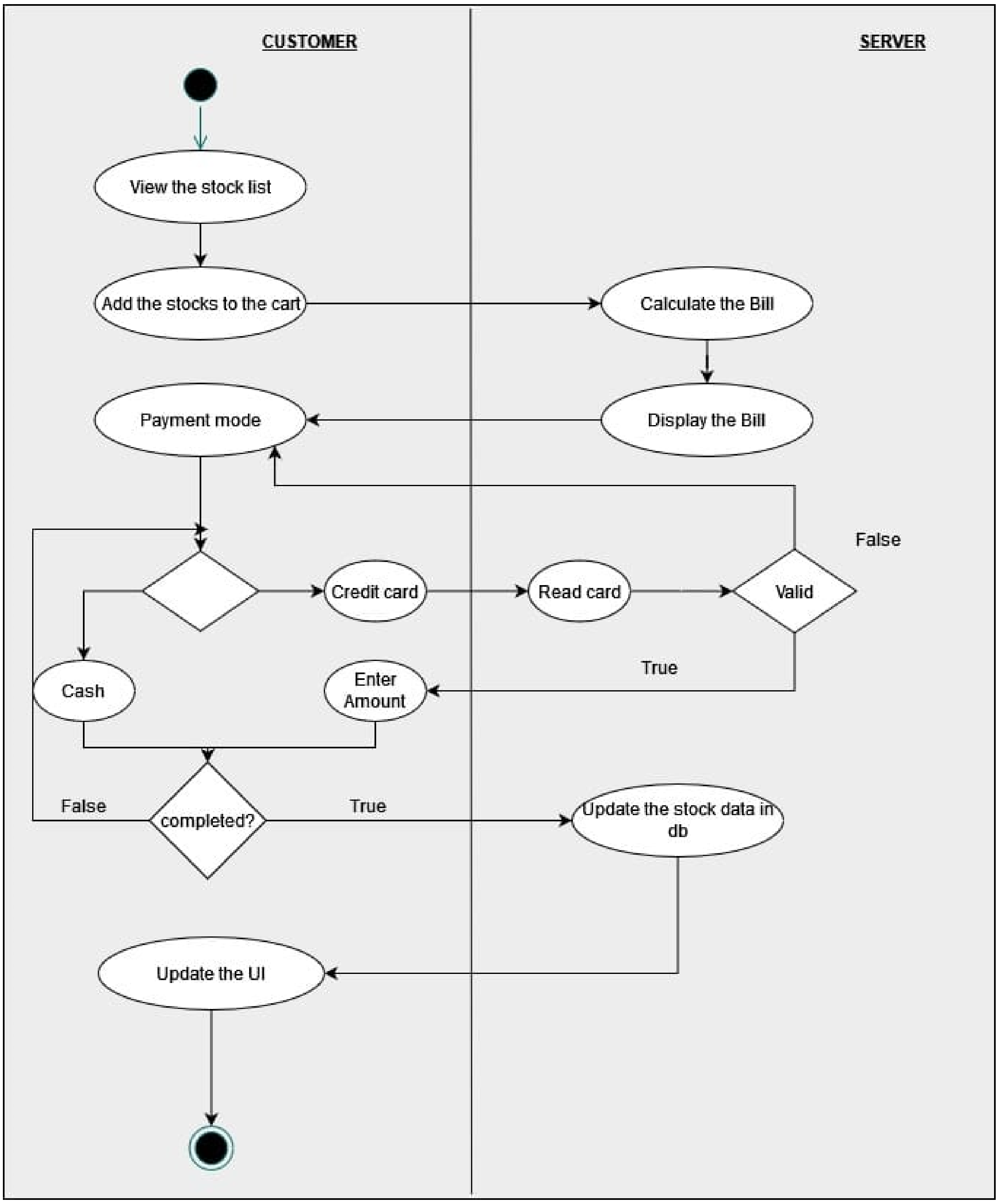
9. Logout:
- The user logs out of the system.
- The process ends, as indicated by the black circle at the bottom.

UML Activity Diagrams

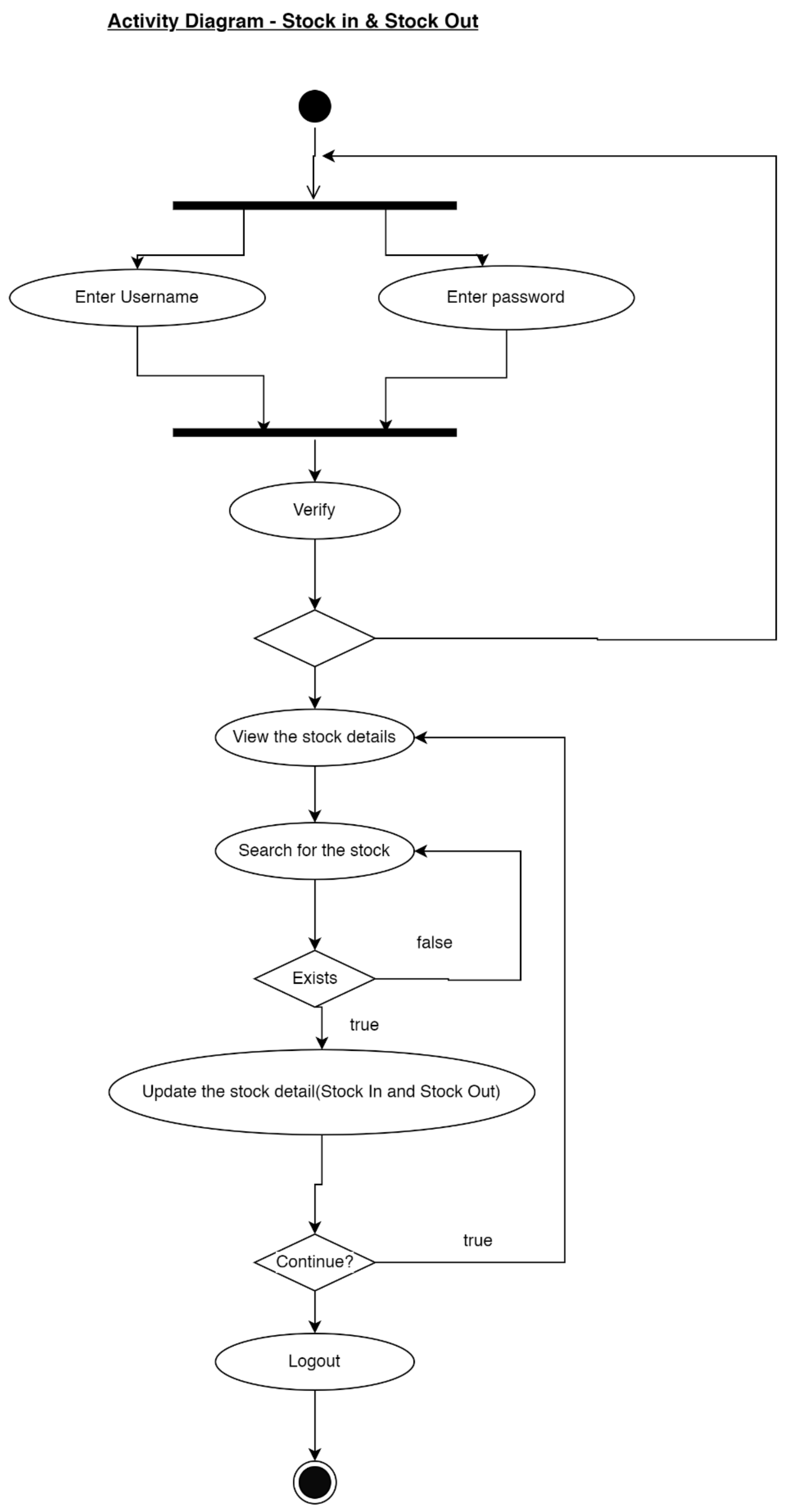
1. Dashboard:



2. Customer – Server:



3. Stock Management:



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

RESULT:

Thus, the development of State Diagrams and Activity Diagrams for Stock Inventory Management System has been completed successfully.