**LAB 6**

**AIM: To draw activity diagram of a) ATM b)Amizone**

**THEORY:**

An **Activity Diagram** is a behavioral UML (Unified Modeling Language) diagram that represents the flow of activities

or processes in a system. It visually describes how tasks are performed in a step-by-step manner, making it useful for

modeling workflows, business processes, and system operations.

**Components of an Activity Diagram**

1. **Start Node** →Represents the beginning of the process (denoted by a filled black circle).

2. **Activity States** →Represent tasks or actions performed (depicted as rounded rectangles).

3. **Decision Nodes** →Represent choices or conditions (depicted as diamonds).

4. **Fork and Join Nodes** →Represent parallel processes:

o **Fork** (splits one flow into multiple parallel flows).

o **Join** (merges multiple parallel flows into one).

5. **Transitions (Arrows)** →Show the direction of process flow.

6. **End Node** →Represents the termination of the process (depicted as a black circle with a border).

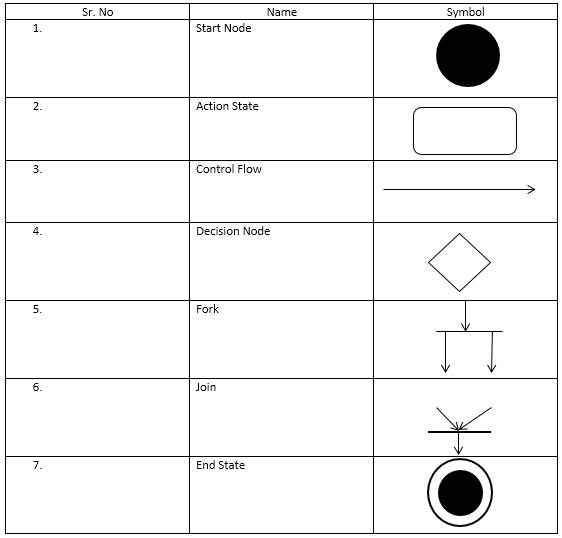
**Purpose of an Activity Diagram**

 **Models Workflows:** It represents sequential and parallel processes in business or software workflows.

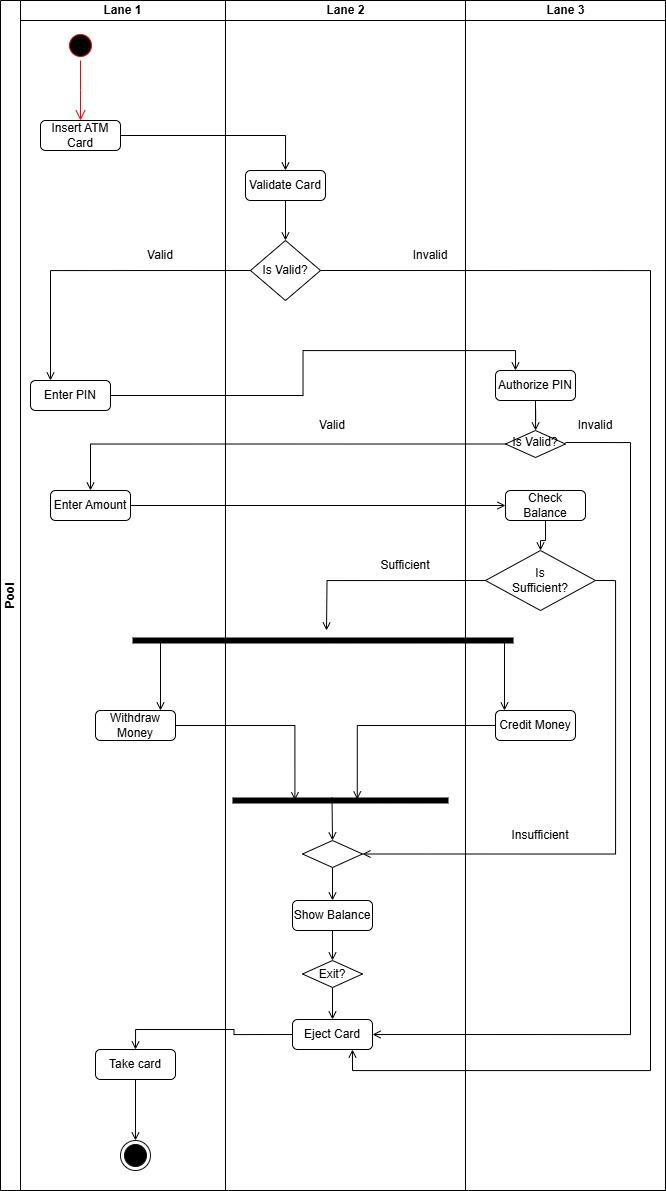
 **Clarifies System Behavior:** Helps in understanding how a system responds to different user interactions.

 **Identifies Bottlenecks:** Highlights inefficiencies or delays in processes.

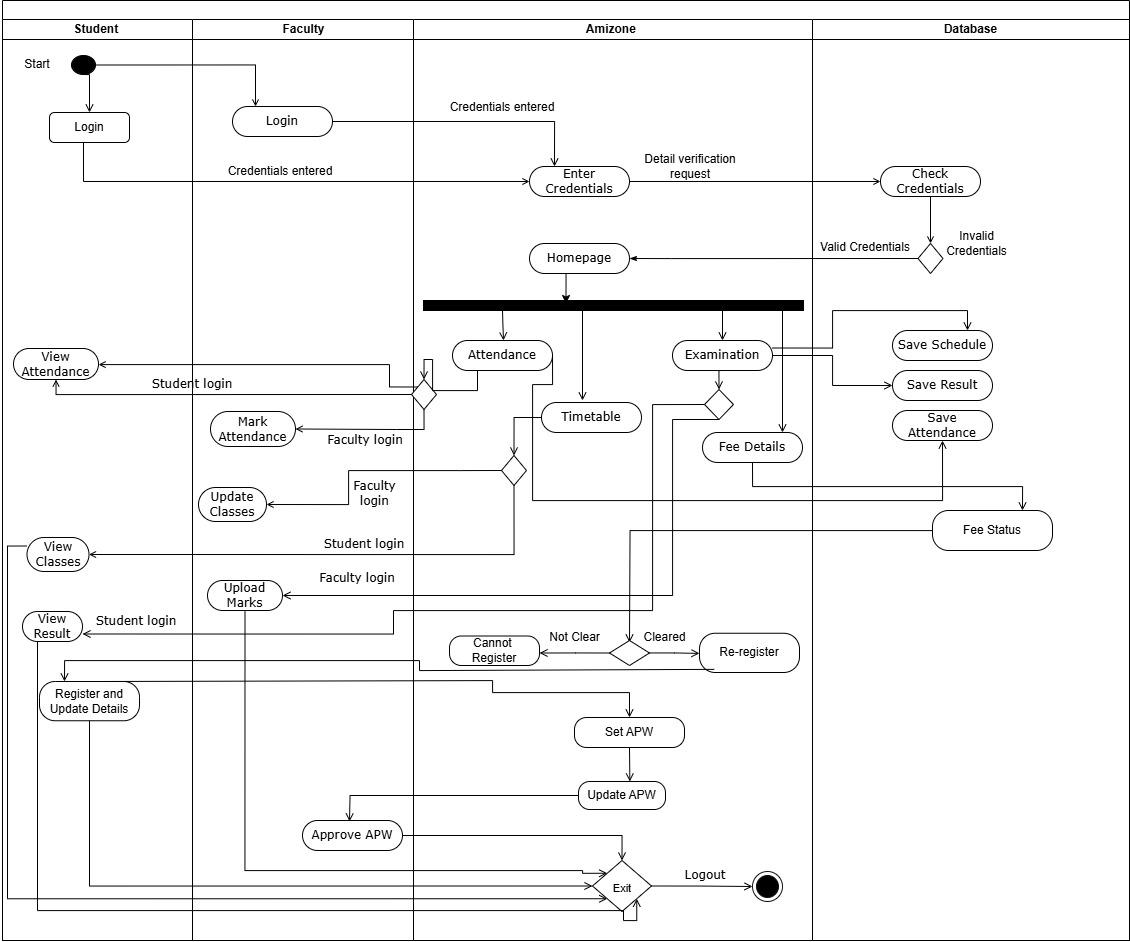
 **Aids in Software Design:** Helps developers visualize system operations before implementation.



a) Activity diagram of ATM



b) Activity diagram of Amizone



**LAB 7**

**AIM: To draw component diagram of ATM**

**THEORY:**

A **Component Diagram** is a structural UML (Unified Modeling Language) diagram that represents the **physical components** of a system and their relationships. It shows how different parts of a system interact to **form a complete application**. Component diagrams are especially useful in **software architecture design**, helping to understand the dependencies among various components.

**Key Elements of a Component Diagram**

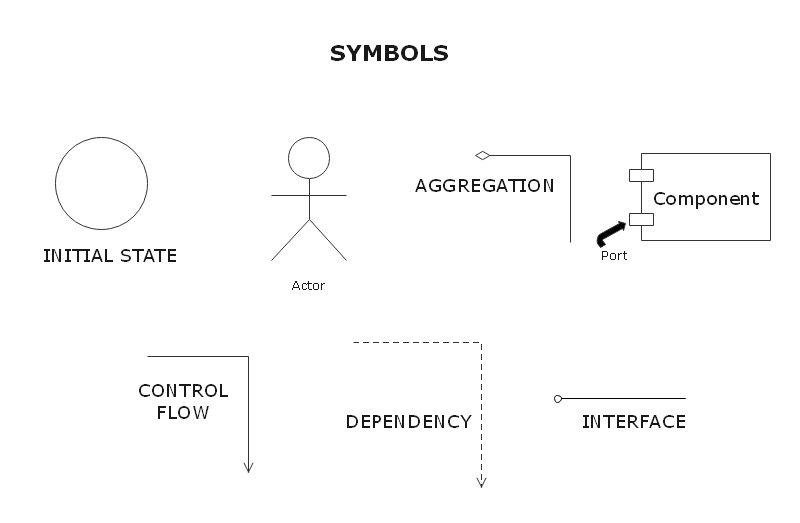
|  |  |
| --- | --- |
| 1.  2.  3.  4.  5. | **Component** →Represents a modular part of the system (depicted as a rectangle with a small box-like symbol).  **Interface** →Defines how components communicate (shown as a circle or lollipop symbol). **Dependency** →Represents relationships between components (dashed arrows).  **Ports** →Small rectangles on component edges, representing interaction points.  **Subsystems** →Logical grouping of related components. |

**Purpose of a Component Diagram**

A component diagram is essential for **visualizing the high-level structure of complex software systems**. It helps in understanding **component interactions, module dependencies, and system organization**. Developers and architects use these diagrams for **planning, analyzing, and optimizing software architecture**, ensuring modularity and scalability.

**Applications of Component Diagrams**

|  |  |
| --- | --- |
|        | **Software Architecture Design** – Helps in designing modular and maintainable software systems by clearly defining components and their dependencies.  **System Implementation Planning** – Used during the development phase to break down complex systems into smaller, manageable modules.  **Dependency Management** – Identifies how components rely on each other, helping to prevent tight coupling.  **Enterprise Applications** – Commonly used in large-scale applications such as **banking systems, e-commerce platforms, and university management systems** to represent interactions between different modules like authentication, databases, and user interfaces. |



Component diagram of ATM

