# PRACTICAL-01

## Software Requirement Specification of ATM Management System

## 1. Introduction

1.1. **Purpose:** The purpose of this document is to provide a Software Requirements Specification (SRS) for an ATM Management System. This system is designed to manage the operations of Automated Teller Machines (ATMs) in a bank. The main objective of this system is to ensure efficient and secure transactions for bank customers through ATMs.

1.2. **Scope:** Using an ATM, customers can access their bank deposit or credit accounts in order to make a variety of financial transactions, most notably cash withdrawals and balance checking, as well as transferring credit to and from mobile phones. ATMs can also be used to withdraw cash in a foreign country.

1.3. **Overview**: ATM stands for automated teller machine. These are electronic banking outlets that allow people to complete transactions without going into a branch of their bank. Some ATMs are simple cash dispensers, while others allow a variety of transactions such as check deposits, balance transfers, and bill payments.

## 2. General description

The ATM (Automated Teller Machine) Management System is a software system designed to manage the operations of ATMs in a bank. The system's primary objective is to ensure efficient and secure transactions for bank customers through ATMs.

The system allows bank customers to carry out various transactions such as cash withdrawal, cash deposit, account balance inquiry, and funds transfer using ATMs. The system also provides ATM status monitoring and ATM cash management features to ensure the uninterrupted availability of cash at the ATMs.

The ATM Management System comprises two main components: the hardware component (ATMs) and the software component (ATM Management System). The ATMs are installed in various locations, allowing customers to carry out transactions. The ATM Management System is installed in the bank's central server, which manages all the ATM transactions.

## 3. Interface requirements

The user interface of an ATM (Automated Teller Machine) plays a crucial role in providing a seamless and user-friendly banking experience for customers. The following are some of the interface requirements of an ATM:

1.1. **Display Screen:** The ATM should have a high-resolution display screen that is easily readable in various lighting conditions. The display screen should be large enough to display all necessary information, such as transaction details, balance inquiries, and instructions.

1.2. **Keypad:** The ATM should have a keypad that is easy to use and responsive. The keypad should be durable and designed to withstand frequent use. The keys should be clearly labelled, and the layout should be intuitive for the customers to navigate.

1.3. **Touch Screen**: In addition to the keypad, the ATM may also have a touch screen option. The touch screen should be highly responsive and easy to use. It should be intuitive and provide clear instructions to guide customers through the transaction process.

1.4. **Audio Features:** The ATM should have an audio system that provides clear instructions and feedback to the customers. It should be audible even in noisy environments and easy to adjust the volume.

1.5. **Card Reader**: The ATM should have a reliable and efficient card reader that can read the magnetic stripe or chip on the customer's card. The card reader should be designed to prevent skimming and other forms of fraud.

1.6. **Receipt Printer:** The ATM should have a fast and reliable receipt printer that provides customers with a printed record of their transaction. The printer should be designed to minimize paper waste and ensure easy loading and replacement of paper rolls.

1.7. **Language Options:** The ATM should provide language options to cater to customers who speak different languages. The language options should be easy to navigate, and the ATM should be capable of displaying instructions and transaction details in multiple languages.

## 4. Performance requirements of ATM

The performance requirements of an ATM (Automated Teller Machine) are critical to ensure efficient and reliable transactions for customers. The following are some performance requirements of an ATM:

**4.1 Speed:** The ATM should perform transactions quickly and efficiently to minimize customer waiting times. For example, the cash dispensing process should be fast and accurate.

**4.3 Reliability:** The ATM should be reliable and available for use at all times. It should have a high uptime and be capable of handling multiple transactions simultaneously.

**4.3 Security:** The ATM should have robust security features that prevent unauthorized access to customer data and funds. It should be design to prevent fraud, such as skimming and card trapping.

**4.4 Capacity:** The ATM should be designed to handle large volumes of cash and transaction data. It should be capable of storing and dispensing a significant amount of cash, and its data storage capacity should be sufficient to handle transaction records.

**4.5 Maintenance:** The ATM should be designed to minimize the need for maintenance and repair. It’s hardware and software should be easy to replace or upgrade, and its cash and receipt handling systems should be easy to access and maintain.

**4.6 Compatibility:** The ATM should be compatible with different types of cards, including debit and credit cards, and should be capable of handling transactions from different banks and financial institutions.

**4.7 Accessibility:** The ATM should be accessible to customers with disabilities, such as those who are visually impaired or have mobility challenges. It should comply with accessibility guidelines and be designed to accommodate a range of user needs.

## 5. Design Constraints

**5.1 Physical Space**: The ATM must be designed to fit within a limited physical space. This constraint may affect the size and shape of the ATM, as well as the location and orientation of its components.

**5.2 Power Consumption**: The ATM must be designed to operate within a specific power budget. This constraint may impact the choice of components and the design of the power supply system.

**5.3 Connectivity:** The ATM must be connected to a network or the internet to process transactions. This constraint may impact the choice of communication protocols and the design of the network infrastructure.

**5.4 Security:** The ATM must be designed to prevent unauthorized access to customer data and funds. This constraint may affect the choice of hardware and software components, as well as the design of the user interface.

**5.5 Cost:** The ATM must be designed to be cost-effective and affordable for the bank. This constraint may affect the choice of materials, components, and manufacturing processes.

**5.6 Regulatory Compliance:** The ATM must comply with relevant regulations and standards, such as those related to accessibility, security, and data protection. This constraint may affect the choice of components and the design of the user interface.

**5.7 User Experience:** The ATM must be designed to provide a positive user experience for customers. This constraint may affect the design of the user interface, as well as the choice of components such as the display screen, keypad, and audio system.

## 6. Non-functional attributes of ATM

Non-functional attributes of an ATM (Automated Teller Machine) are characteristics that do not relate to the primary function of the system but are still critical to its overall performance and user experience. The following are some non-functional attributes of an ATM:

6.1 **Security:** The security of an ATM is a critical non-functional attribute. It involves protecting the customer's sensitive data, such as personal and financial information, from unauthorized access and fraud.

6.2 **Availability:** The availability of an ATM refers to the percentage of time it is operational and available for customer use. High availability is critical to ensure customers can access their funds whenever needed.

6.3 **Performance:** The performance of an ATM includes factors such as transaction speed, system response time, and throughput. An ATM with fast transaction speeds and quick system response times can help reduce wait times and improve the customer experience.

6.4 **Usability:** The usability of an ATM is a non-functional attribute that relates to how easy it is for customers to interact with the system. It includes factors such as the design of the user interface, the ease of use of the keypad and touch screen, and the clarity of instructions.

6.5 **Maintainability:** Maintainability refers to the ease with which an ATM can be maintained and repaired. This attribute includes factors such as the accessibility of components and the ease of diagnosing and fixing problems.

6.6 **Scalability:** The scalability of an ATM is a non-functional attribute that refers to the ability of the system to handle increasing volumes of transactions over time. It involves factors such as hardware and software capacity, as well as the ability to integrate with other systems.

6.7 **Interoperability:** Interoperability refers to the ability of an ATM to interact with other systems, such as the bank's core banking system, payment networks, and other third-party systems.

## 7. Preliminary schedule and budget

The preliminary schedule and budget for an ATM project can vary depending on several factors, such as the scope of the project, the number of ATMs to be installed, and the level of customization required. However, here is a general overview of what a preliminary schedule and budget might look like for an ATM project:

### Preliminary Schedule:

1. Requirements gathering: 1-2 weeks
2. Design and development: 6-8 weeks
3. Hardware procurement and assembly: 2-4 weeks
4. Software installation and configuration: 1-2 weeks
5. Testing and quality assurance: 2-4 weeks
6. Deployment and installation: 1-2 weeks
7. Training and documentation: 1-2 weeks Total project timeline: 14-24 weeks

Preliminary Budget:

1. Hardware and components: \$10,000 - \$20,000 per ATM
2. Software development and licensing: \$5,000 - \$10,000 per ATM
3. Installation and setup: \$2,000 - \$5,000 per ATM
4. Testing and quality assurance: \$5,000 - \$10,000 per ATM
5. Training and documentation: \$2,000 - \$5,000 per ATM Total project budget: \$24,000 - \$50,000 per ATM

# PRACTICAL – 2

## Draw the use case diagram.

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses. Four important components of use-case diagram-
1. System, 2. Actors, 3. Use cases and 4. Relationships

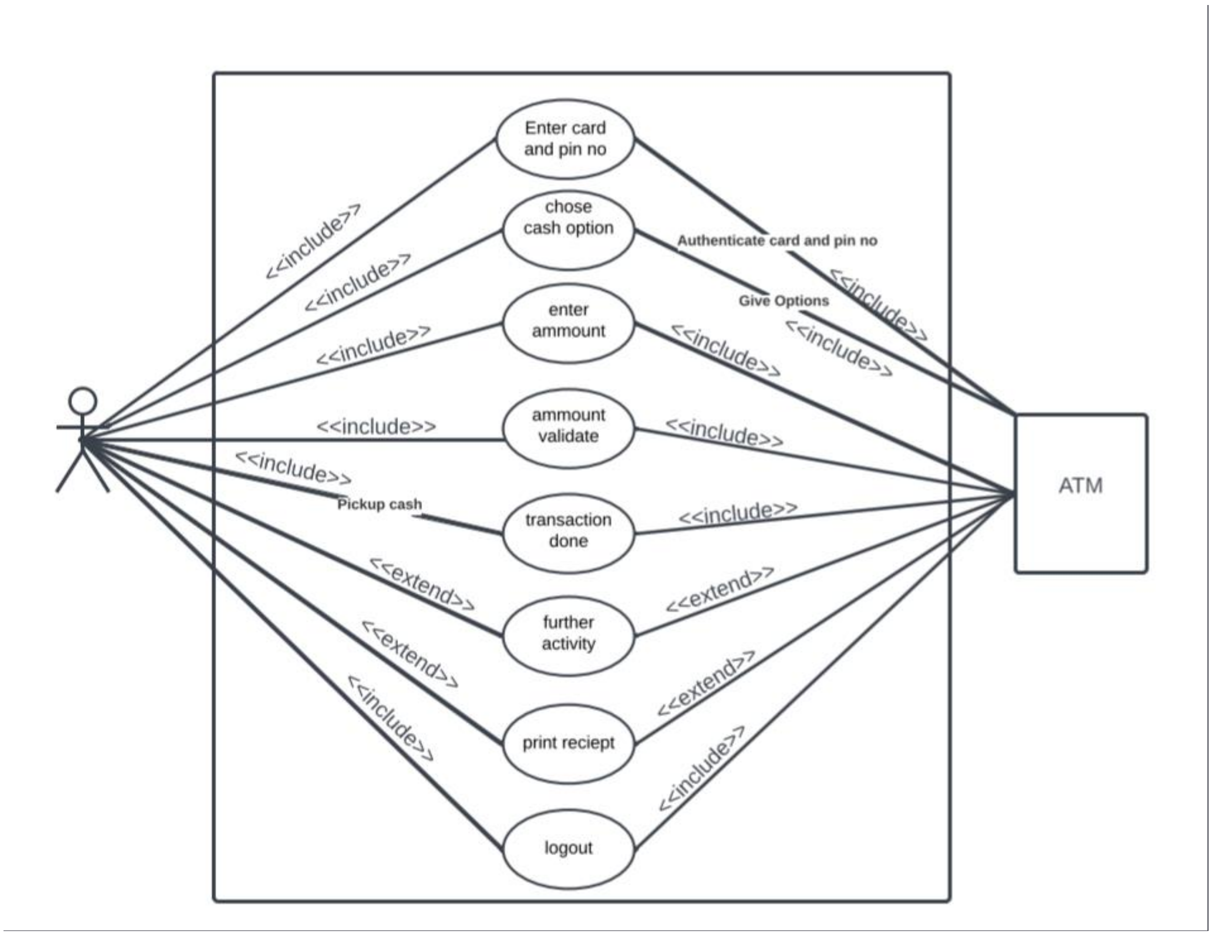


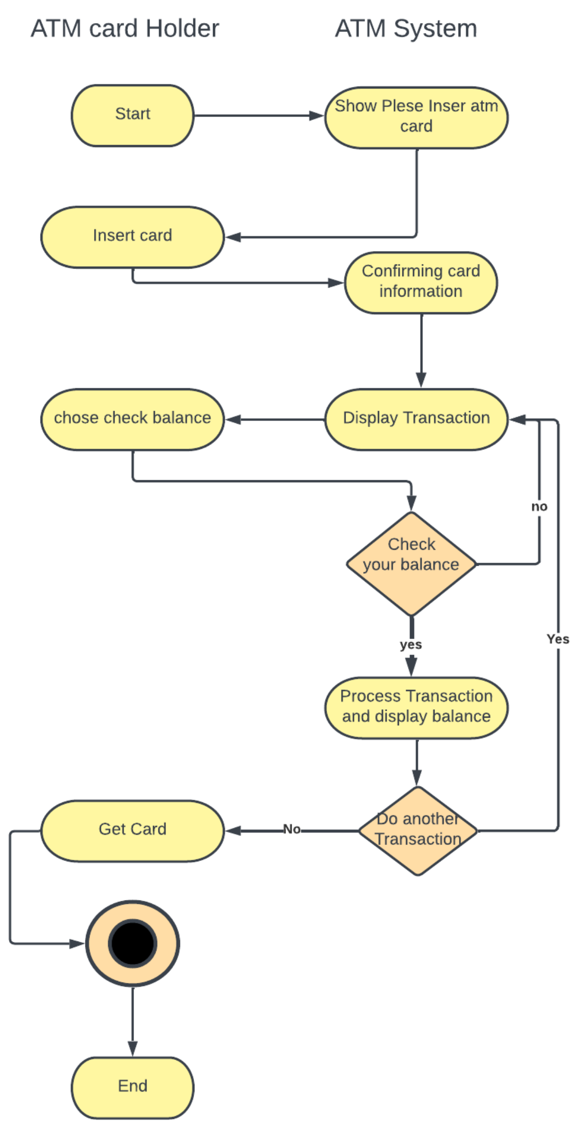
Fig. Usecase Diagram of ATM Management System

# PRACTICAL – 3

## Draw the activity diagram

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. Basic components of an activity diagram- 1. action 2. node 3.decision node 4.control flow 5.start node 6.end node

## Activity Diagram



# PRACTICAL – 4

## Draw the Class diagram.

Class diagrams are the blueprints of your system or subsystem. You can use class diagrams to model the objects that make up the system, to display the relationships between the objects, and to describe what those objects do and the services that they provide. Class diagrams are useful in many stages of system design.

# PRACTICAL – 5

## Draw the Sequence diagram.

A sequence diagram or system sequence diagram shows process interactions arranged in time sequence in the field of software engineering. It depicts the processes involved and the sequence of messages exchanged between the processes needed to carry out the functionality. A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process.

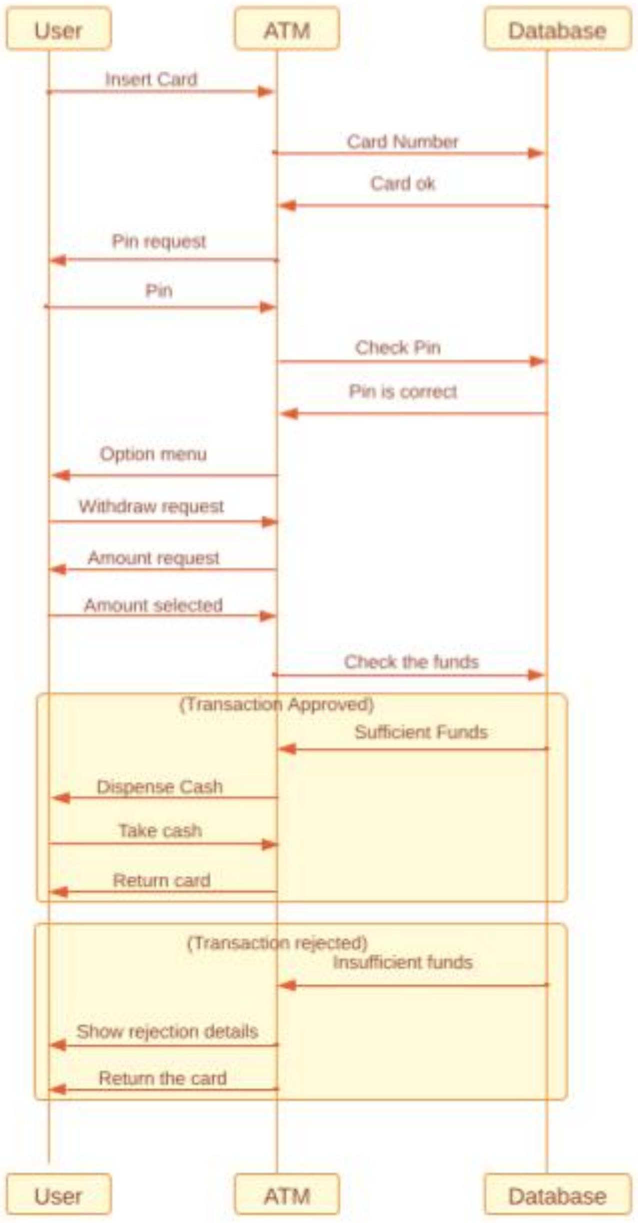


Fig. Sequence Diagram of ATM Management System

# PRACTICAL – 6

## COLLABORATION DIAGRAM

A collaboration diagram is a type of diagram used in software engineering to visualize the interactions between objects in a system. It is also known as a communication diagram or interaction diagram. Here are the main components of a collaboration diagram:

**Objects-** The entities that interact with the system, represented as rectangles with the object name.

**Links-** The lines that connect objects and represent the message exchanges between them.

**Messages-** The interactions between objects, represented as arrows with the message name and any parameters or arguments passed.

**Self-links-** The messages that an object sends to itself, represented as a loopback arrow.

**Lifelines-** The vertical dotted lines that represent the lifespan of an object.

Collaboration diagrams are useful for visualizing the interactions between objects in a system and can help to identify potential problems or areas for improvement. They can also be used to model the behaviour of a system at a high level, making it easier to understand and communicate the design to others.

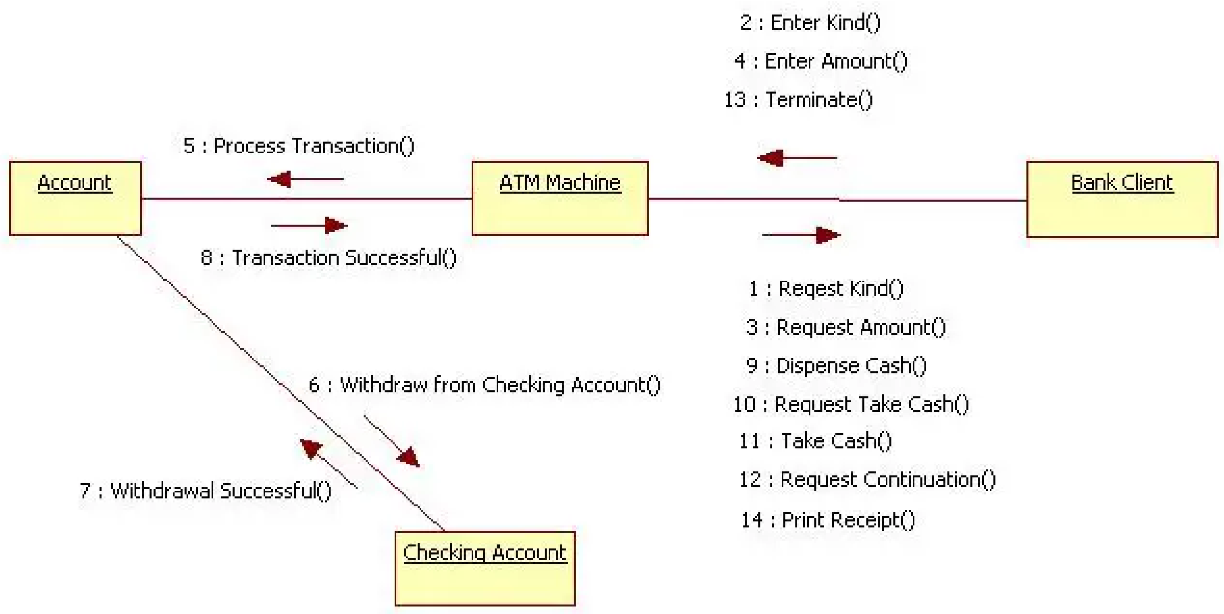


Fig. Collaboration diagram of ATM Management System

# PRACTICAL – 7

## STATE CHART DIAGRAM

A state chart diagram is a type of diagram used in software engineering to model the behaviour of objects in a system. It shows how an object transitions between different states based on events and conditions. The components of a state chart diagram include**:**

**States-** The states represent the different stages or conditions of the system or object. Each state is depicted as a rectangle with rounded corners. The name of the state is written inside the rectangle.

**Transitions-** The transitions represent the events or actions that cause the system or object to move from one state to another. They are depicted as arrows connecting the states. The label on the arrow describes the event or action that triggers the transition.

**Events-** The events are the stimuli that cause the system or object to change state. They can be internal or external to the system. Examples of events include user input, system messages, and timer expirations.

**Actions-** Actions are the behaviours or operations that occur when a state transition is triggered. They can be associated with the transition or with the state itself. Examples of actions include sending a message, updating a variable, or invoking a method.

**Initial state-** The initial state is the starting point of the system or object. It is represented by a solid circle.

**Final state-** The final state represents the end of the system or object's life cycle. It is represented by a solid circle with a dot inside.

**Guards-** Guards are conditions that must be satisfied before a transition can occur. They are represented as square brackets around the event label.

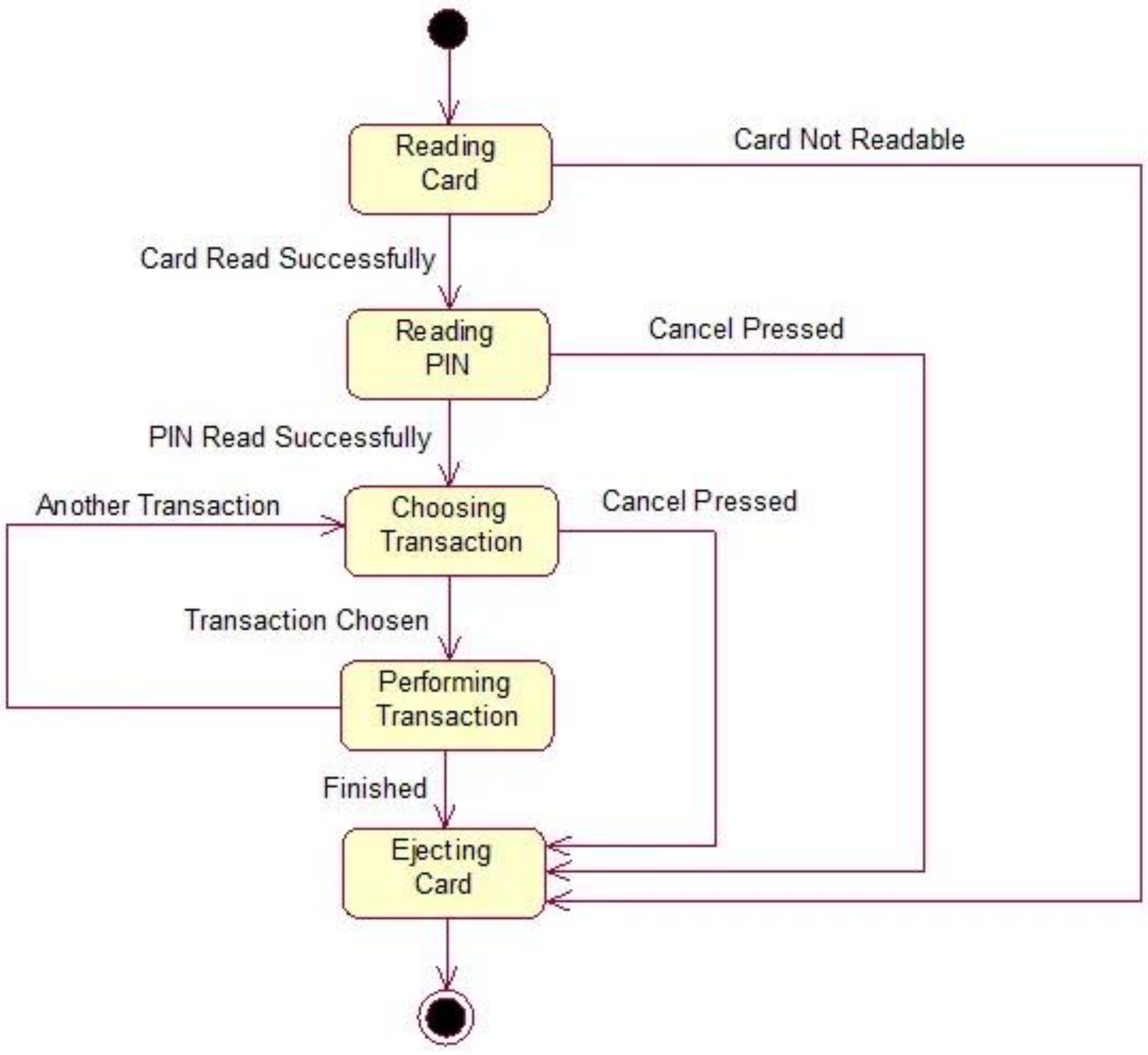


Fig. State Chart Diagram of ATM Management System

# EXPERIMENT NO –8

## COMPONENT DIAGRAM

A component diagram in UML (Unified Modelling Language) is a type of structural diagram that shows the components of a system and their relationships. The components are the building blocks of the system, and the diagram shows how they interact and collaborate to provide the system's functionality. The main components of a component diagram include:

**Component-** A component is a modular, self-contained, and reusable unit of software that can be deployed and executed independently. It can be a physical component, such as a server or a database, or a logical component, such as a module or a class. It is represented as a rectangle with the component name inside.

**Interface-** An interface is a contract between a component and its environment. It specifies the operations that the component can perform and the messages that it can send and receive. It is represented as a circle with the interface name inside.

**Port-** A port is a point of interaction between a component and its environment. It is a connection point where messages can be sent and received. It is represented as a small square on the edge of a component or an interface.

**Connector-** A connector is a link that connects two components or interfaces. It represents the communication or interaction between them. It is represented as a line connecting the ports of the components or interfaces.

**Relationship-** A relationship is a connection between two components that indicates their dependency or association. The different types of relationships in a component diagram include inheritance, aggregation, composition, and realization.

**Dependency-** A dependency is a relationship between two components that indicates that one component depends on the other. It is represented as a dashed arrow pointing from the dependent component to the independent component.

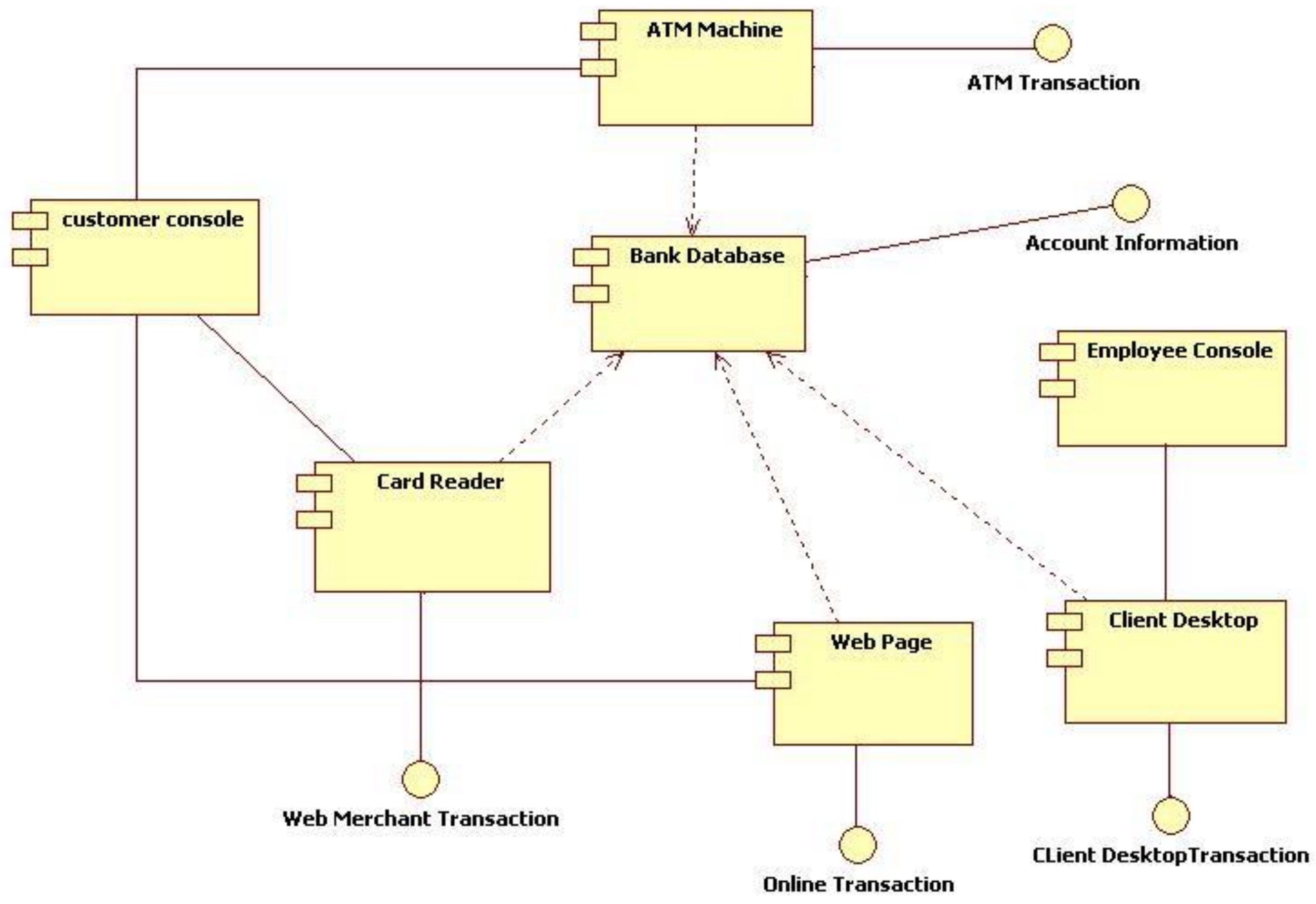


Fig. Component Diagram of ATM Management System