

**Software Engineering**

**[IT301]**

**Practical File**

**Submitted By-**

**Kuldeep Dwivedi**

**A2305218477(CSE)**

**A close up of a sign

Description automatically generated**

**AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY**

**AMITY UNIVERSITY CAMPUS, SECTOR-125, NOIDA-201303**

**INDEX**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| SNo | Name of Experiment | Page No. | Date of Conduct | Date of Submission | Signature of Faculty |
| 1 | To create a Class Diagram in UML. | 1-2 | 22/12/2020 | 05/01/2021 |  |
| 2 | To create a Use Case Diagram in UML. | 3-4 | 05/01/2021 | 05/01/2021 |  |
| 3 | To create a State Diagram in UML. | 5-6 | 12/01/2021 | 19/01/2021 |  |
| 4 | To create an Object Diagram in UML. | 7-8 | 19/01/2021 | 02/02/2021 |  |
| 5 | To create an Activity Diagram in UML. | 9-10 | 02/02/2021 | 09/02/2021 |  |
| 6 | To create a Sequence Diagram in UML. | 11-12 | 09/02/2021 | 16/02/2021 |  |
| 7 | To create a Component Diagram in UML. | 13-14 | 16/02/2021 | 23/02/2021 |  |
| 8 | To create a Collaboration Diagram in UML. | 15-16 | 23/02/2021 | 02/03/2021 |  |
| Professional Skills Development Activities [PSDA] | | | | | |
| 9 | PSDA 1: Automatic Washing Machine | 18-22 | 02/03/2021 | 09/03/2021 |  |
| 10 | PSDA 2: Vending Machine | 23-27 | 16/03/2021 | 23/03/2021 |  |

**EXPERIMENT 1**

**AIM**: To create a Class Diagram in UML.

**THEORY**:

What is a Class Diagram?

In software engineering, 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.

A UML class diagram is made up of:

* A set of classes and
* A set of relationships between classes

Why use a Class Diagram?

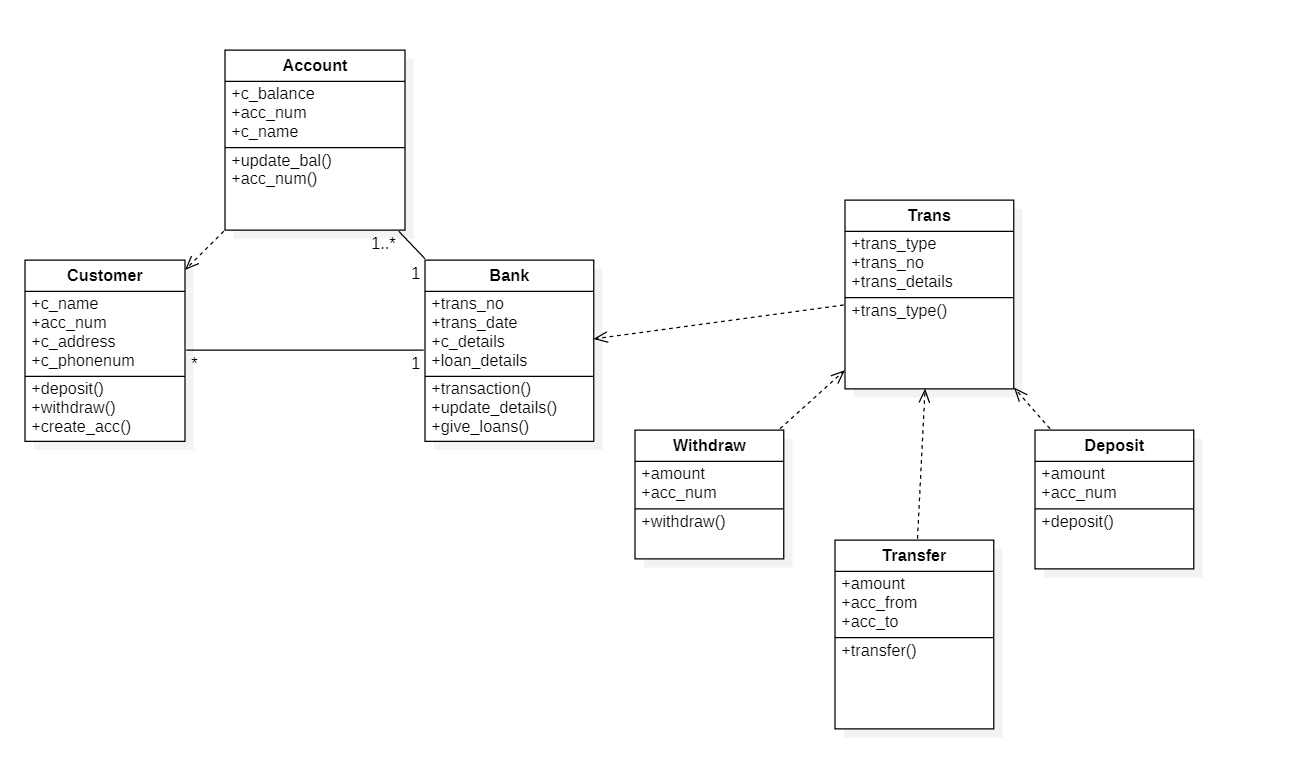
* Shows static structure of classifiers in a system
* Diagram provides a basic notation for other structure diagrams prescribed by UML
* Helpful for developers and other team members too
* Business Analysts can use class diagrams to model systems from a business perspective

How to draw a Class Diagram?

* The name of the class diagram should be meaningful to describe the aspect of the system.
* Each element and their relationships should be identified in advance.
* Responsibility (attributes and methods) of each class should be clearly identified
* For each class, a minimum number of properties should be specified, as unnecessary properties will make the diagram complicated.
* Use notes whenever required to describe some aspect of the diagram. At the end of the drawing it should be understandable to the developer/coder.
* Finally, before making the final version, the diagram should be drawn on plain paper and reworked as many times as possible to make it correct.

**SOFTWARE USED**: Star UML

**Class Diagram: Online Banking System**



**EXPERIMENT 2**

**AIM**: To create a Use Case Diagram in UML.

**THEORY**:

What is a Use Case Diagram?

A UML use case diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behavior (what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (i.e. use case diagram). A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

A use case diagram is usually simple. It does not show the detail of the use cases:

* It only summarizes some of the relationships between use cases, actors, and systems.
* It does not show the order in which steps are performed to achieve the goals of each use case.

Why use a Use Case Diagram?

* Specify the context of a system
* Capture the requirements of a system
* Validate a systems architecture
* Drive implementation and generate test cases

How to draw a Use Case Diagram?

When we are planning to draw a use case diagram, we should have the following items identified.

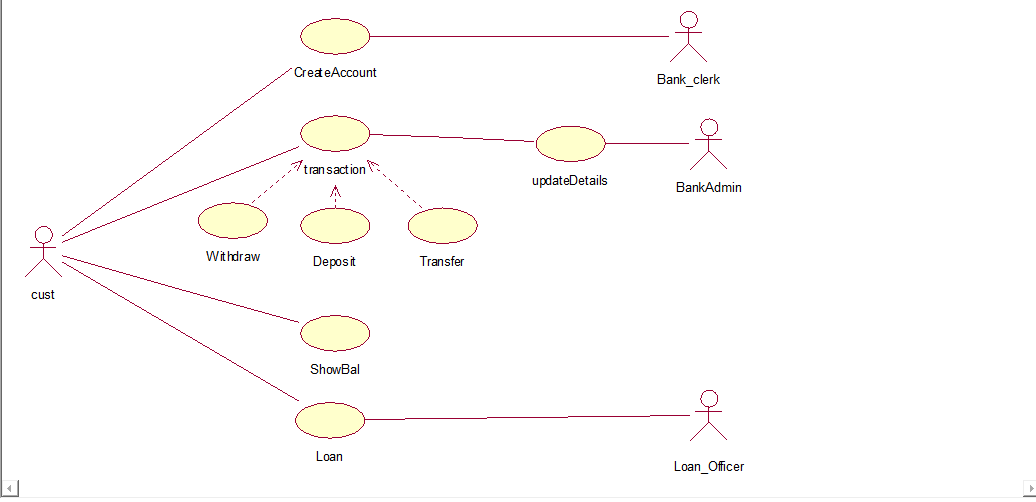
* Functionalities to be represented as use case
* Actors
* Relationships among the use cases and actors.

We have to use the following guidelines to draw an efficient use case diagram

* The name of a use case is very important. The name should be chosen in such a way so that it can identify the functionalities performed.
* Give a suitable name for actors.
* Show relationships and dependencies clearly in the diagram.
* Do not try to include all types of relationships, as the main purpose of the diagram is to identify the requirements.
* Use notes whenever required to clarify some important points.

**SOFTWARE USED**: Rational Rose

**Use Case Diagram: Online Banking System**

****

**EXPERIMENT 3**

**AIM**: To create a State Diagram in UML.

**THEORY**:

What is a State Diagram?

UML State Machine Diagrams (or sometimes referred to as state diagrams, state machines or state charts) show the different states of an entity. State machine diagrams can also show how an entity responds to various events by changing from one state to another. State machine diagram is a UML diagram used to model the dynamic nature of a system.

Statechart diagrams provide us an efficient way to model the interactions or communication that occur within the external entities and a system. These diagrams are used to model the event-based system. A state of an object is controlled with the help of an event.

Why use a State Diagram?

* 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 lifetime.
* Define a state machine to model the states of an object.
* To design interactive systems that respond to either internal or external events.

How to draw a State Diagram?

Before drawing a Statechart diagram we should clarify the following points −

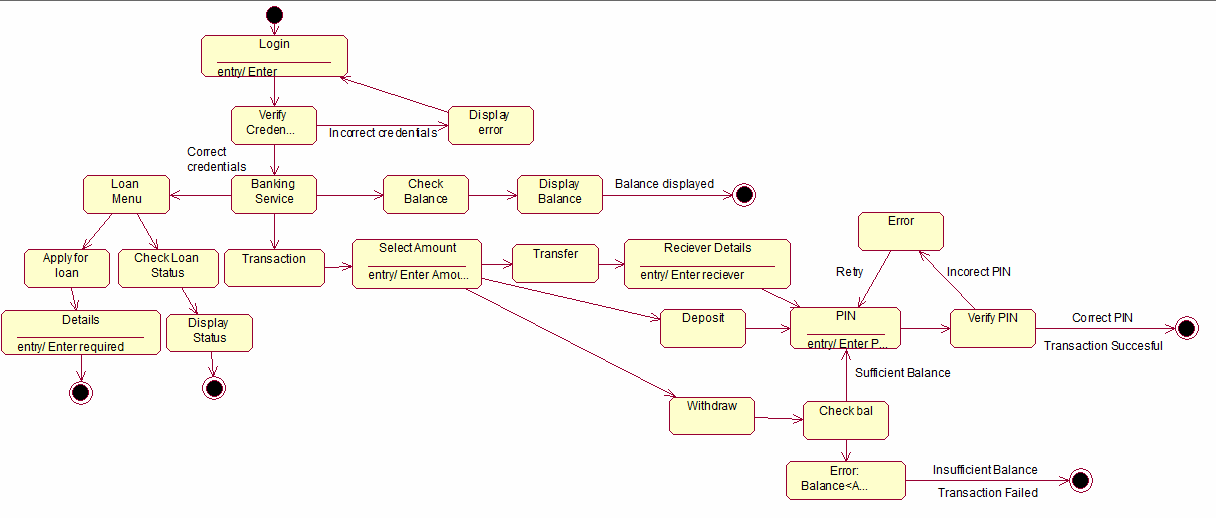
* Identify the important objects to be analyzed.
* Identify the states.
* Identify the events.

Following rules must be considered while drawing a state chart diagram:

* The name of a state transition must be unique.
* The name of a state must be easily understandable and describe the behavior of a state.
* If there are multiple objects, then only essential objects should be implemented.
* Proper names for each transition and an event must be given.

**SOFTWARE USED**: Rational Rose

**State Diagram: Online Banking System**

****

**EXPERIMENT 4**

**AIM**: To create an Object Diagram in UML.

**THEORY**:

What is an Object Diagram?

A static UML object diagram is an instance of a class diagram; it shows a snapshot of the detailed state of a system at a point in time, thus an object diagram encompasses objects and their relationships which may be considered a special case of a class diagram. Object diagrams are used to render a set of objects and their relationships as an instance.

Why use an Object Diagram?

* Forward and reverse engineering.
* Object relationships of a system
* Static view of an interaction.
* Understand object behaviour and their relationship from practical perspective

How to draw an Object Diagram?

Before drawing an object diagram, the following things should be remembered and understood clearly −

* Object diagrams consist of objects.
* The link in the object diagram is used to connect objects.
* Objects and links are the two elements used to construct an object diagram.

After this, the following things are to be decided before starting the construction of the diagram −

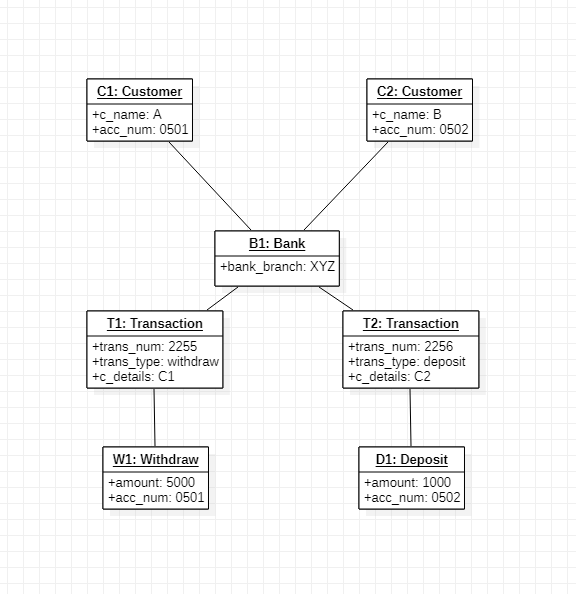
* The object diagram should have a meaningful name to indicate its purpose.
* The most important elements are to be identified.
* The association among objects should be clarified.
* Values of different elements need to be captured to include in the object diagram.
* Add proper notes at points where more clarity is required.

Steps of construction are as follows-

* First, analyze the system and decide which instances have important data and association.
* Second, consider only those instances, which will cover the functionality.
* Third, make some optimization as the number of instances are unlimited.

**SOFTWARE USED**: Star UML

**Object Diagram: Online Banking System**

****

**EXPERIMENT 5**

**AIM**: To create an Activity Diagram in UML.

**THEORY**:

What is an Activity Diagram?

An 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.

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

Activity Diagrams describe how activities are coordinated to provide a service which can be at different levels of abstraction.

Why use an Activity Diagram?

* Model the workflow in a graphical way, which is easily understandable.
* Model the execution flow between various entities of a system.
* Model the detailed information about any function or an algorithm which is used inside the system.
* Model business processes and their workflows.
* Capture the dynamic behavior of a system.
* Generate high-level flowcharts to represent the workflow of any application.
* Model high-level view of an object-oriented or a distributed system.

How to draw an Activity Diagram?

Before drawing an activity diagram, we should identify the following elements −

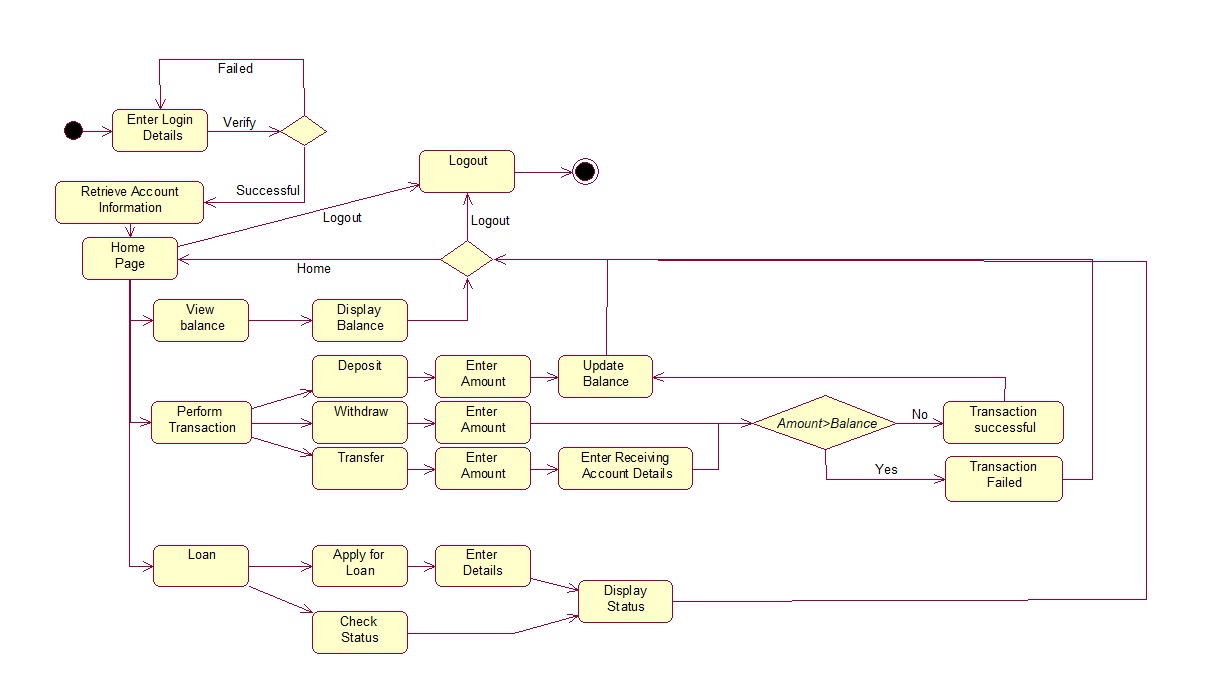
* Activities
* Association
* Conditions
* Constraints

Following rules must be followed while developing an activity diagram,

* All activities in the system should be named.
* Activity names should be meaningful.
* Constraints must be identified.
* Activity associations must be known.

**SOFTWARE USED**: Rational Rose

**Activity Diagram: Online Banking System**

****

**EXPERIMENT 6**

**AIM**: To create a Sequence Diagram in UML.

**THEORY**:

What is a Sequence Diagram?

UML Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focused and they show the order of the interaction visually by using the vertical axis of the diagram to represent time, what messages are sent and when.

Sequence Diagrams captures:

* the interaction that takes place in a collaboration that either realizes a use case or an operation (instance diagrams or generic diagrams)
* high-level interactions between user of the system and the system, between the system and other systems, or between subsystems (sometimes known as system sequence diagrams)

Why use a Sequence Diagram?

* Model high-level interaction between active objects in a system
* Model the interaction between object instances within a collaboration that realizes a use case
* Model the interaction between objects within a collaboration that realizes an operation
* Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction)

How to draw a Sequence Diagram?

Before drawing a Sequence diagram we should clarify the following points −

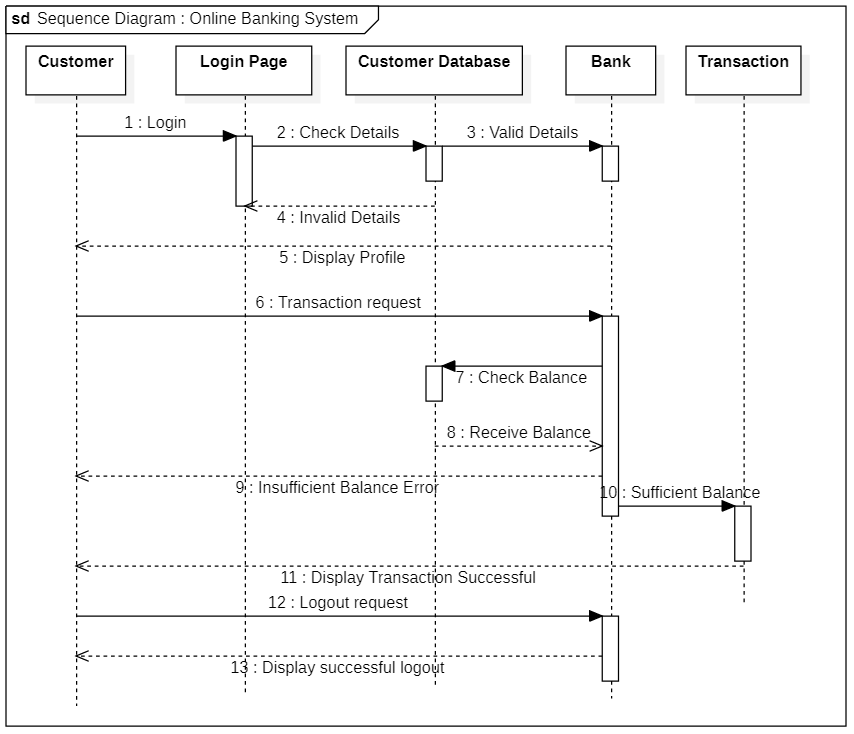
* Lifeline: A lifeline represents an individual participant in the Interaction.
* Activations: A thin rectangle on a lifeline) represents the period during which an element is performing an operation.
* Message: A message defines a particular communication between Lifelines of an Interaction.

Once we have identified the objects, it is then important to write a detailed description on what they do. From this description, it can be easily figured out the interactions that would occur between the objects.

From each of these steps, it can easily be specified what messages should be exchanged between the objects in the sequence diagram. Once it’s clear, we can go ahead and start drawing the sequence diagram.

**SOFTWARE USED**: Star UML

**Sequence Diagram: Online Banking System**

****

**EXPERIMENT 7**

**AIM**: To create a Component Diagram in UML.

**THEORY**:

What is a Component Diagram?

UML Component diagrams are used in modeling the physical aspects of object-oriented systems that are used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering. Component diagrams are essentially class diagrams that focus on a system's components that are often used to model the static implementation view of a system.

Why use a Component Diagram?

* Visualize the components of a system.
* Construct executables by using forward and reverse engineering.
* Describe the organization and relationships of the components.

How to draw a Component Diagram?

Initially, the system is designed using different UML diagrams and then when the artifacts are ready, component diagrams are used to get an idea of the implementation.

This diagram is very important as without it the application cannot be implemented efficiently. A well-prepared component diagram is also important for other aspects such as application performance, maintenance, etc.

Before drawing a component diagram, the following artifacts are to be identified clearly −

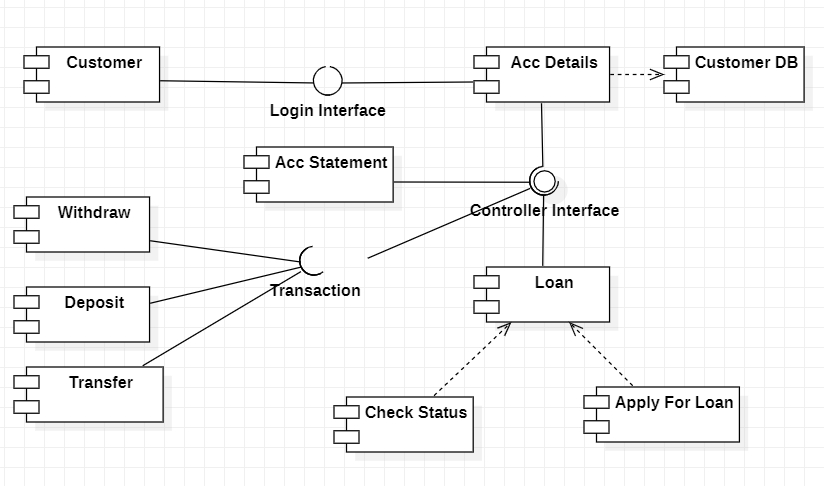
* Files used in the system.
* Libraries and other artifacts relevant to the application.
* Relationships among the artifacts.

After identifying the artifacts, the following points need to be kept in mind.

* Use a meaningful name to identify the component for which the diagram is to be drawn.
* Prepare a mental layout before producing the using tools.
* Use notes for clarifying important points.

**SOFTWARE USED**: Star UML

**Component Diagram: Online Banking System**

****

**EXPERIMENT 8**

**AIM**: To create a Collaboration Diagram in UML.

**THEORY**:

What is a Collaboration Diagram?

Collaboration diagrams are used to show how objects interact to perform the behavior of a particular use case, or a part of a use case. Along with sequence diagrams, collaboration is used by designers to define and clarify the roles of the objects that perform a particular flow of events of a use case. They are the primary source of information used to determine class responsibilities and interfaces.

Why use a Collaboration Diagram?

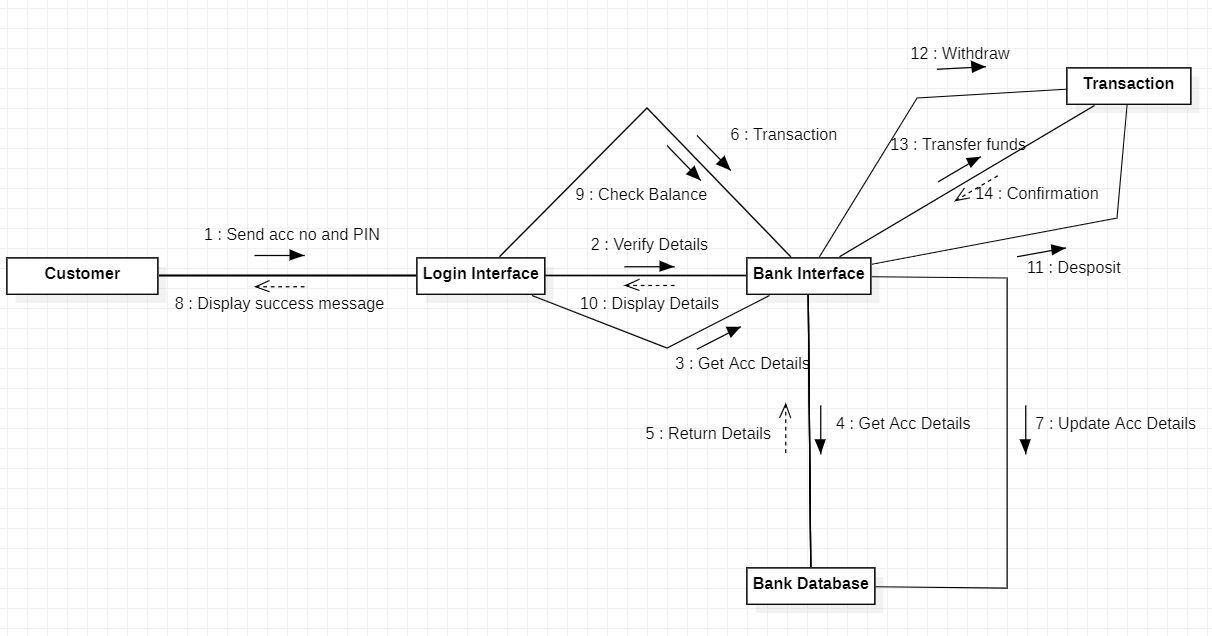
* To capture the dynamic behaviour of a system.
* To describe the message flow in the system.
* To describe the structural organization of the objects.
* To describe the interaction among objects.

How to draw a Collaboration Diagram?

* Determine the behavior for which the realization and implementation are specified.
* Discover the structural elements that are class roles, objects, and subsystems for performing the functionality of collaboration.
* Choose the context of an interaction: system, subsystem, use case, and operation.
* Think through alternative situations that may be involved.
* Implementation of a collaboration diagram at an instance level, if needed.
* A specification level diagram may be made in the instance level sequence diagram for summarizing alternative situations.

**SOFTWARE USED**: Star UML

**Collaboration Diagram: Online Banking System**

****

**Professional Skills**

**Development Activities**

**[PSDA]**

# 

# **CASE STUDY 1**

**AIM**: Design class diagram, object diagram, state diagram, activity diagram, use case diagram, sequence diagram, component diagram and collaboration diagram for automatic vending machine.

**THEORY**:

**Class diagram** is a static diagram.It describes the attributes and operations of a class and also the constraints imposed on the system. Class diagrams are the only UML diagrams, which can be mapped directly with object-oriented languages.

**Object diagrams** are derived from class diagrams. Object diagrams represent an instance of a class diagram. The basic concepts are similar for class diagrams and object diagrams The difference is that a class diagram represents an abstract model consisting of classes and their relationships whereas an object diagram represents an instance at a particular moment, which is concrete in nature.

**Statechart diagram** describes a state machine. It 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.

**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.Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

**Use case diagrams** are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.

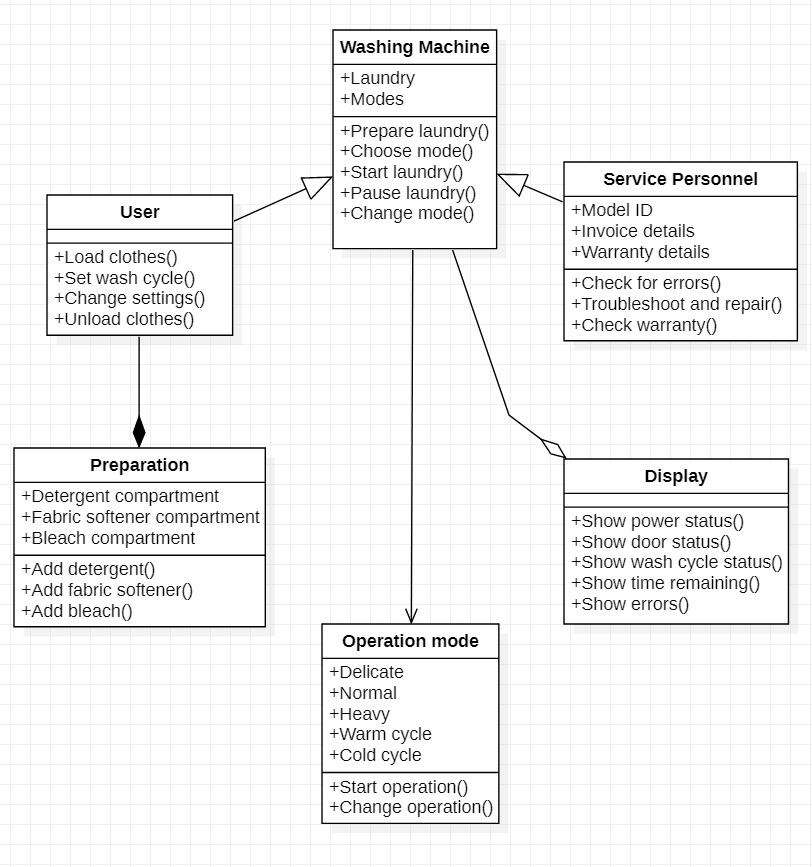
**Sequence Diagrams** are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of collaboration.Sequence Diagrams show elements as they interact over time and they are organized according to object (horizontally) and time (vertically).

**Component diagrams** are different in terms of nature and behavior.Component diagrams are used to model the physical aspects of a system. Physical aspects are the elements such as executables, libraries, files, documents, etc. which reside in a node.Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems.

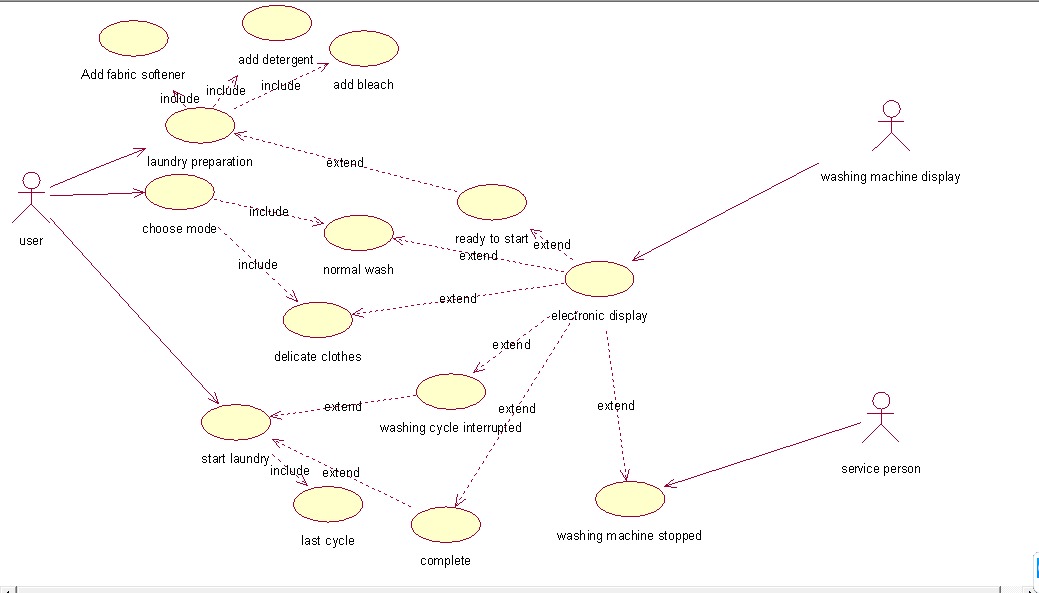
**Collaboration diagram** is used to show the relationship between the objects in a system.Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming.The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

**SOFTWARE USED**: Rational Rose, Star UML

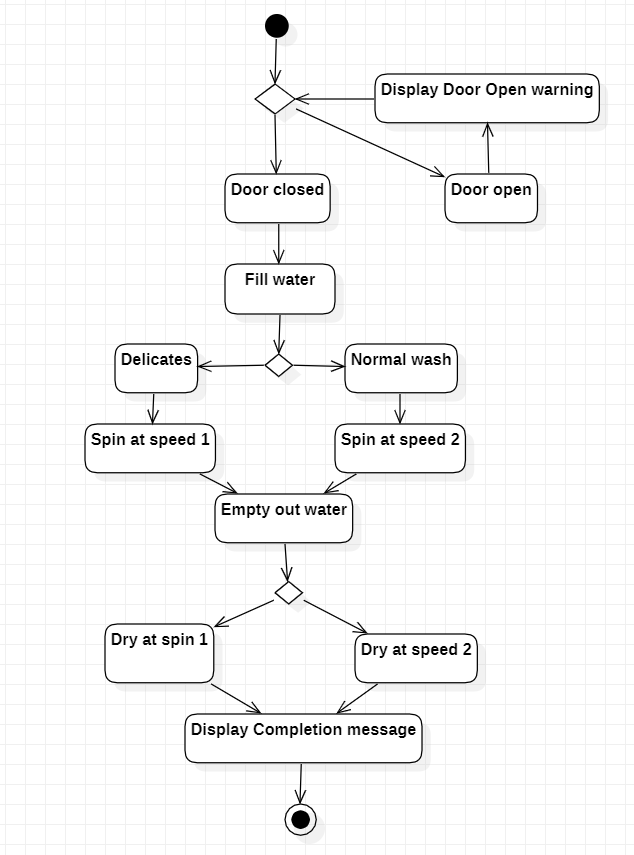
1. **Class Diagram**



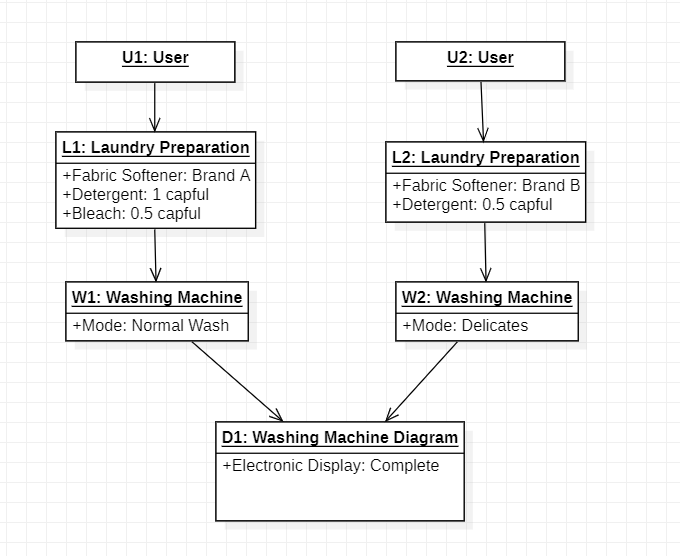
1. **Use Case Diagram**



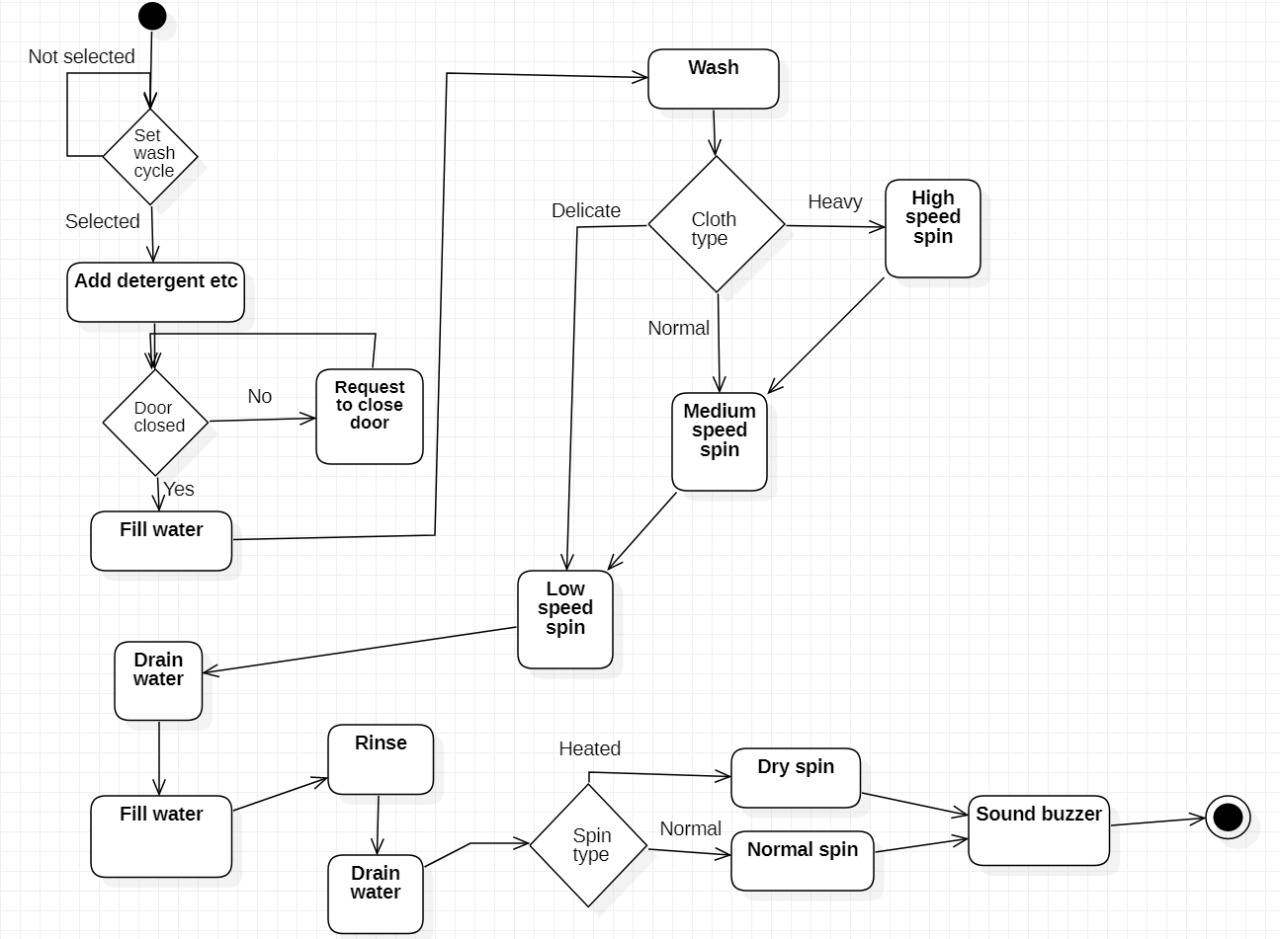
1. **State Diagram**



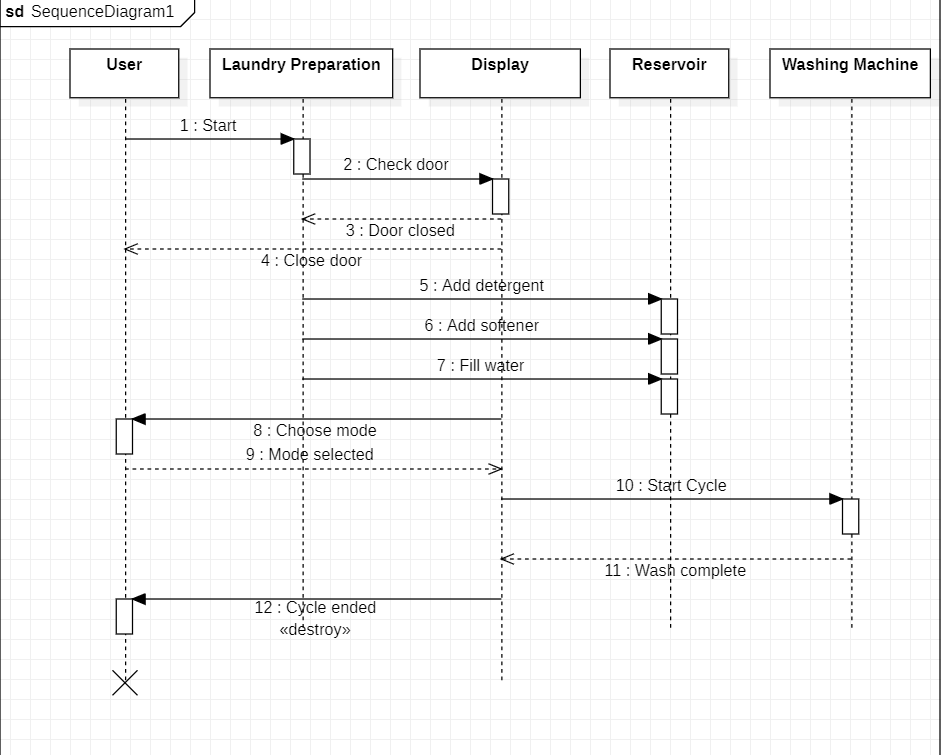
1. **Object Diagram**



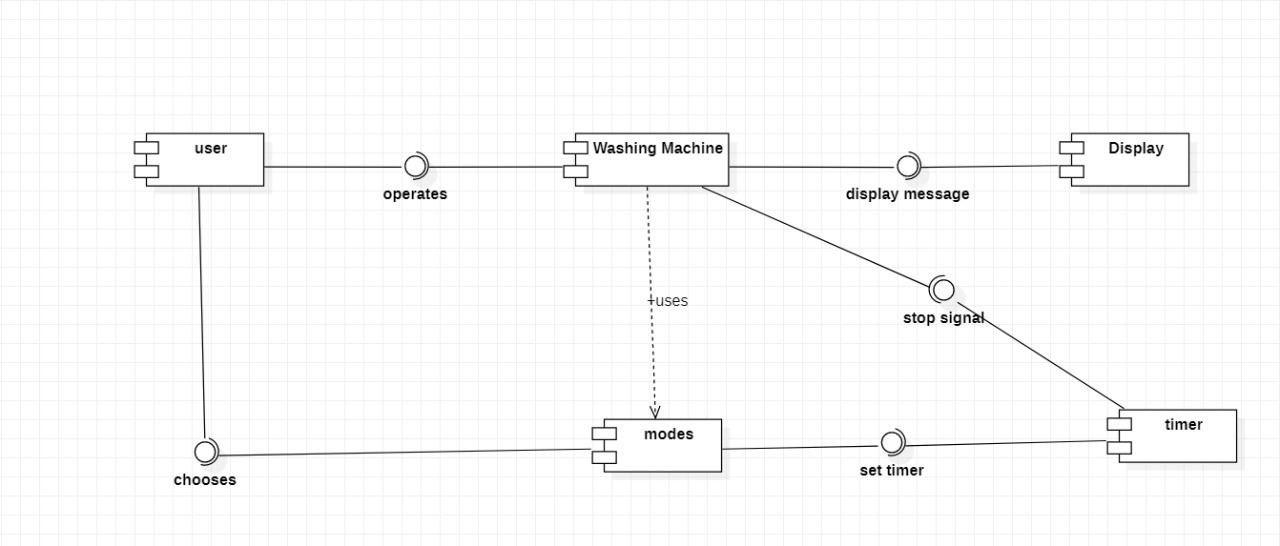
1. **Activity Diagram**



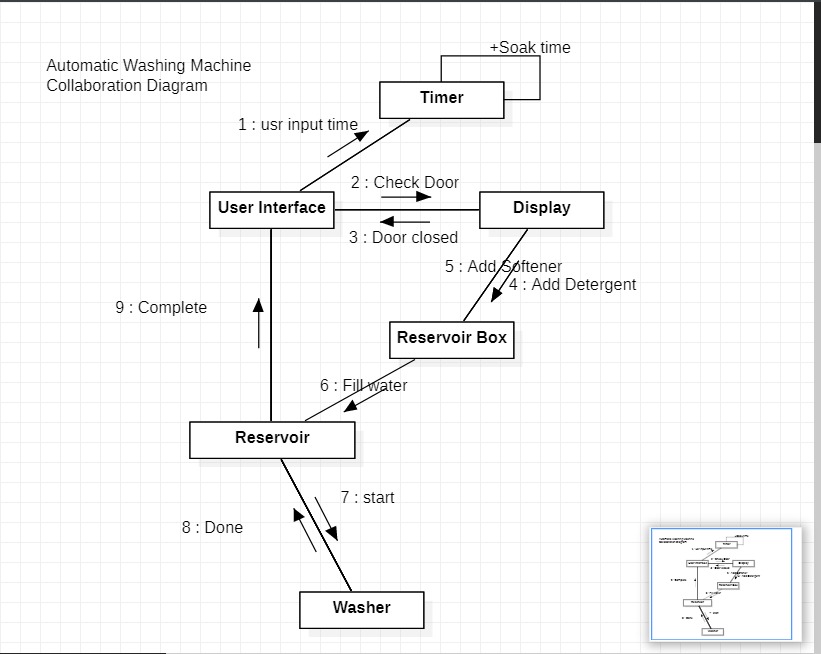
1. **Sequence Diagram**



1. **Component Diagram**

****

1. **Collaboration Diagram**

****

**CASE STUDY 2**

**AIM**: Design class diagram, object diagram, state diagram, activity diagram, use case diagram, sequence diagram, component diagram and collaboration diagram for vending machine.

**THEORY**:

**Class diagram** is a static diagram.It describes the attributes and operations of a class and also the constraints imposed on the system. Class diagrams are the only UML diagrams, which can be mapped directly with object-oriented languages.

**Object diagrams** are derived from class diagrams. Object diagrams represent an instance of a class diagram. The basic concepts are similar for class diagrams and object diagrams The difference is that a class diagram represents an abstract model consisting of classes and their relationships whereas an object diagram represents an instance at a particular moment, which is concrete in nature.

**Statechart diagram** describes a state machine. It 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.

**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.Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

**Use case diagrams** are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.

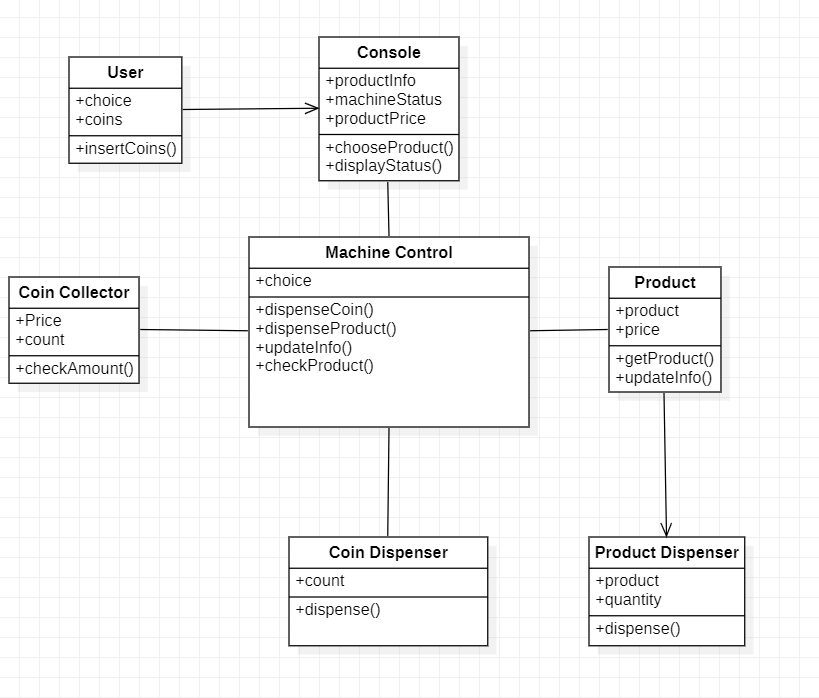
**Sequence Diagrams** are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of collaboration.Sequence Diagrams show elements as they interact over time and they are organized according to object (horizontally) and time (vertically).

**Component diagrams** are different in terms of nature and behavior.Component diagrams are used to model the physical aspects of a system. Physical aspects are the elements such as executables, libraries, files, documents, etc. which reside in a node.Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems.

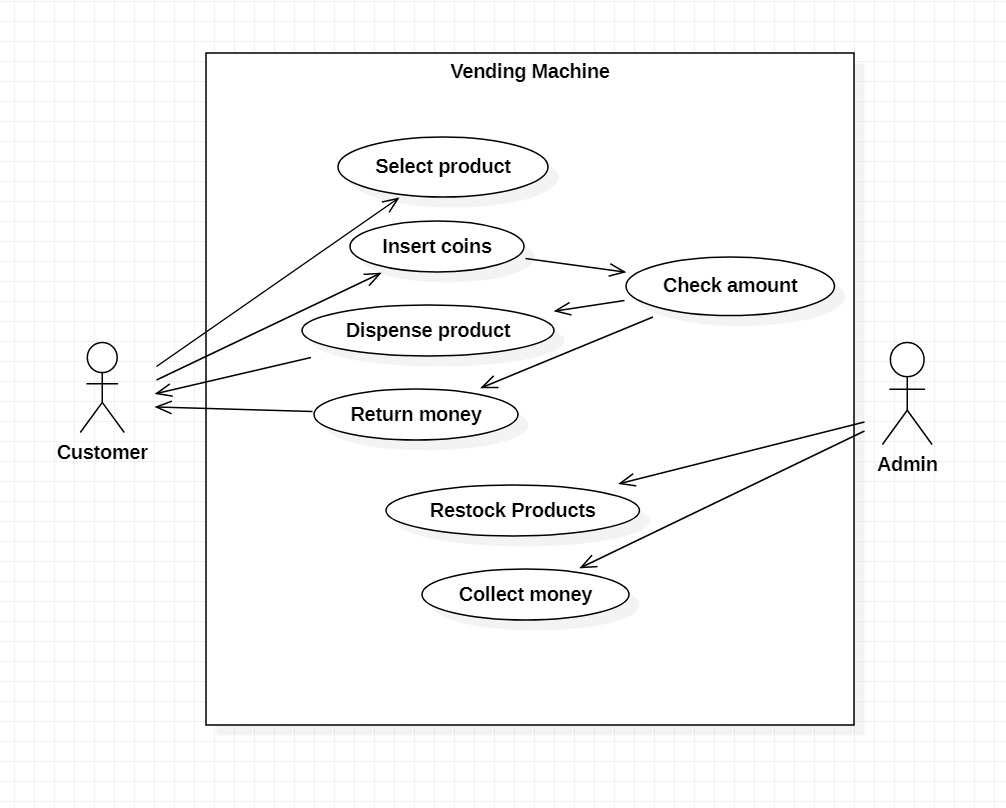
**Collaboration diagram** is used to show the relationship between the objects in a system.Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming.The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

**SOFTWARE USED**: Rational Rose, Star UML

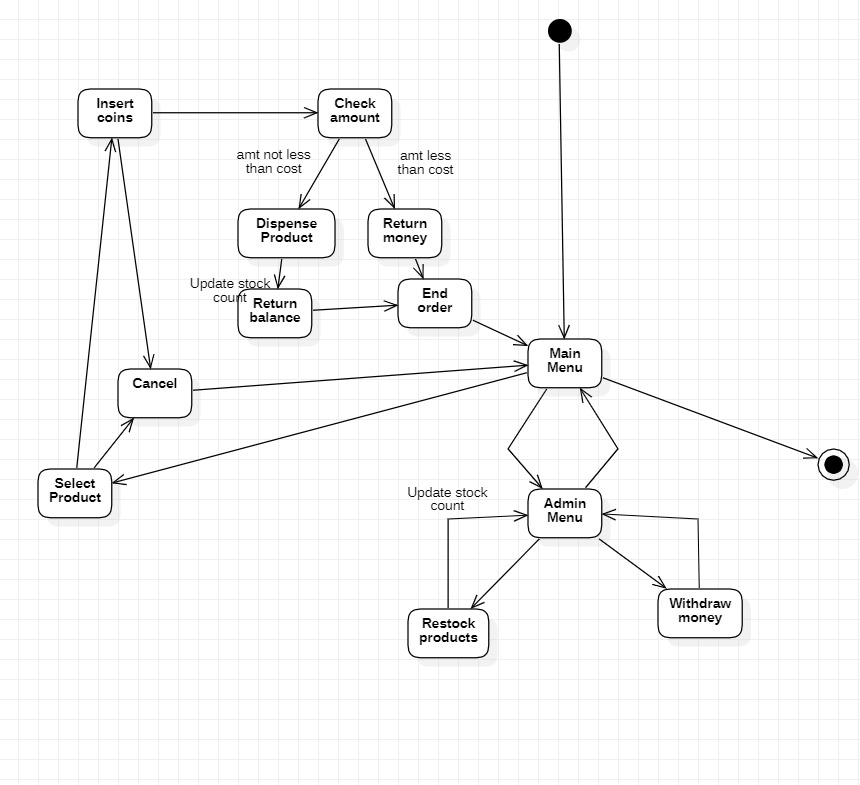
1. **Class Diagram**



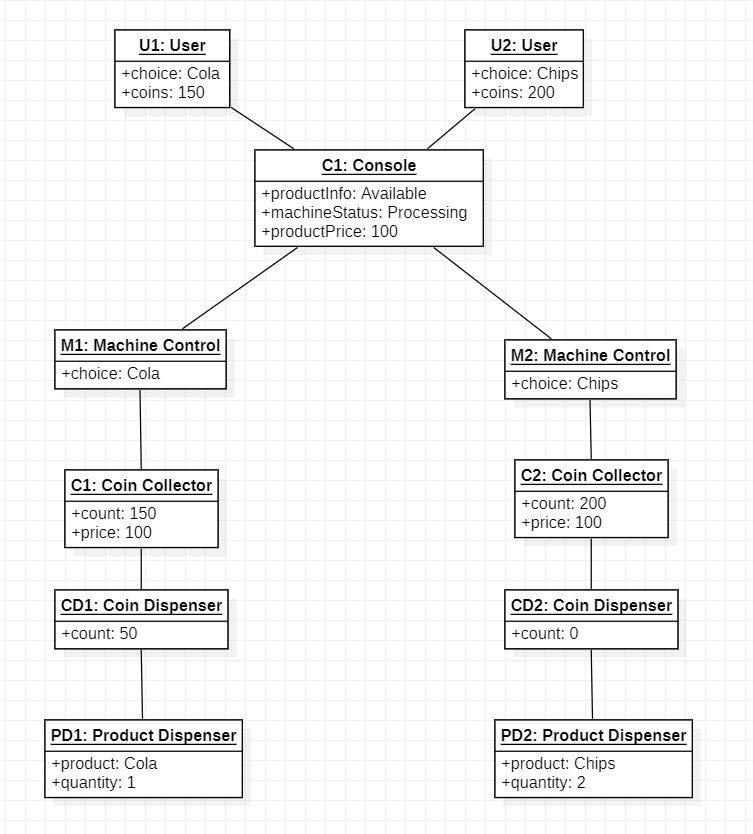
1. **Use Case Diagram**



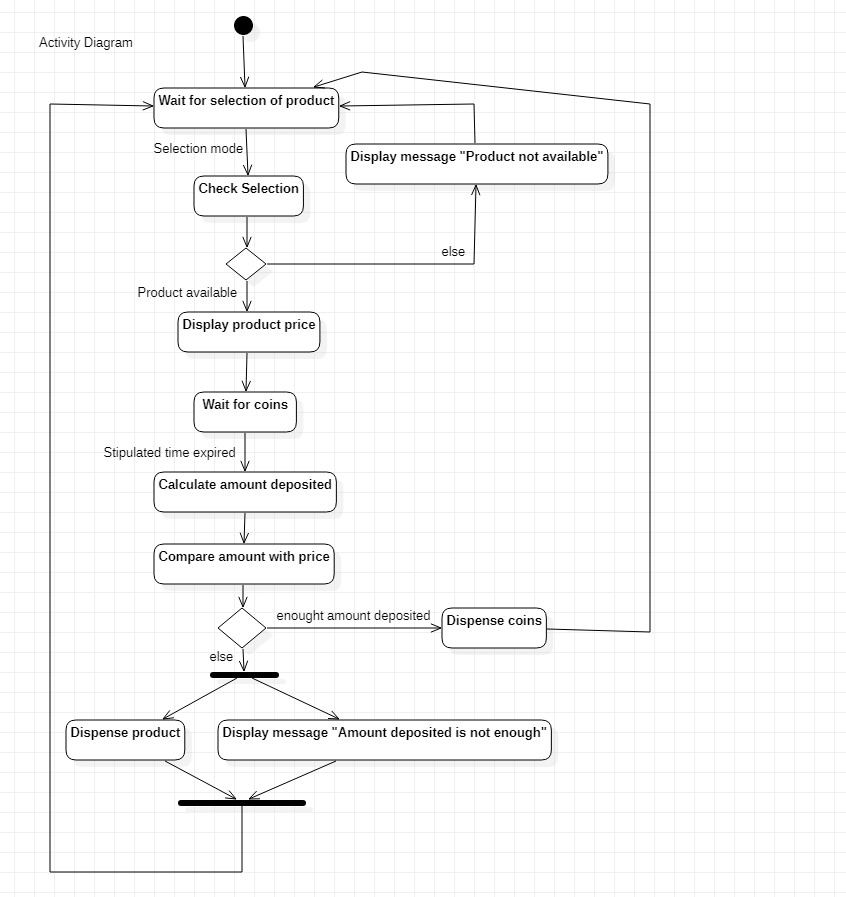
1. **State Diagram**



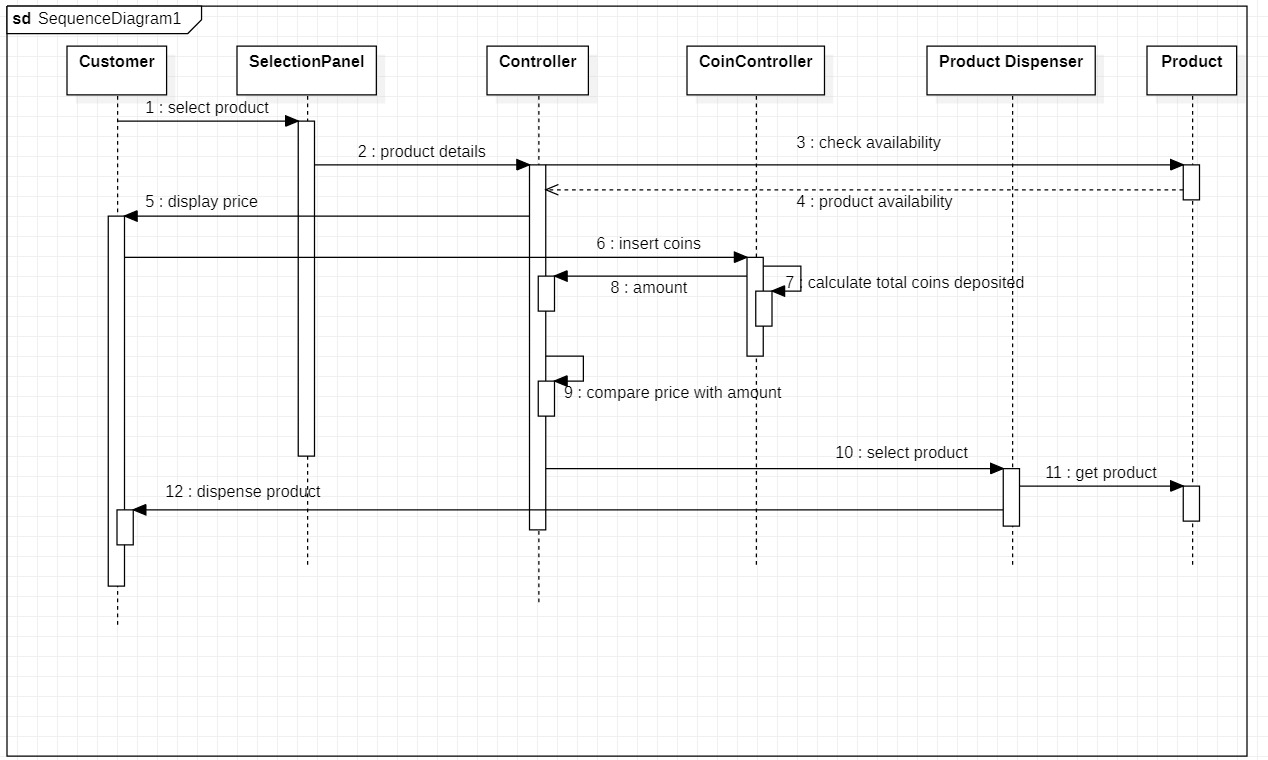
1. **Object Diagram**



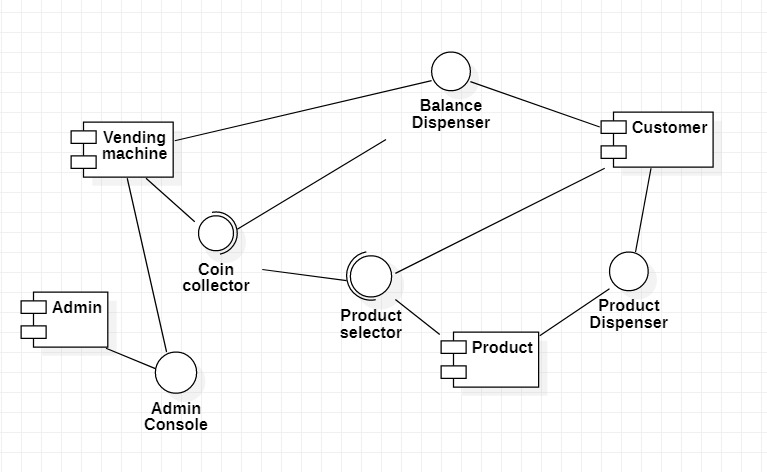
1. **Activity Diagram**



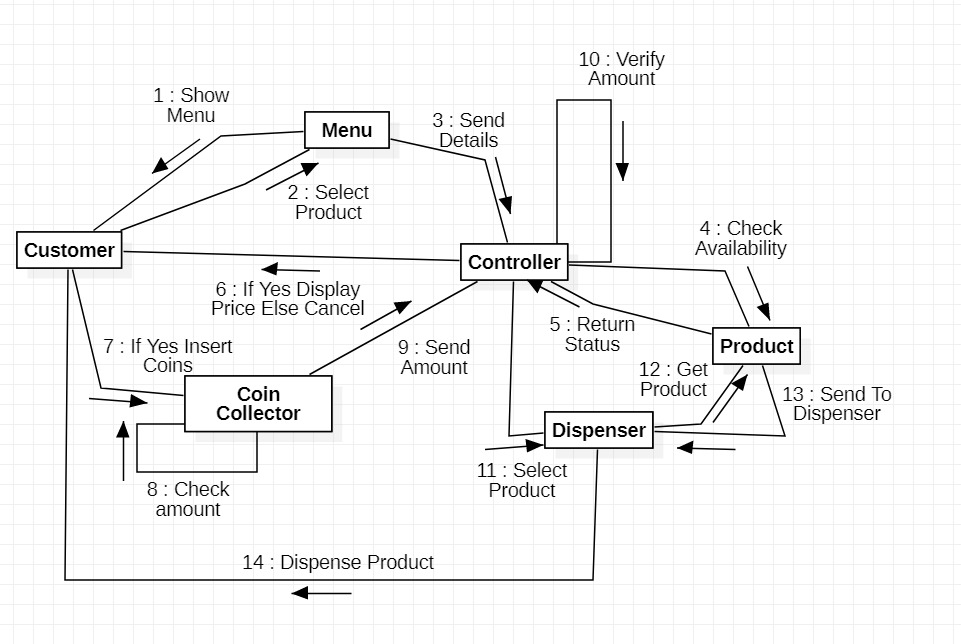
1. **Sequence Diagram**



1. **Component Diagram**

****

1. **Collaboration Diagram**

****