EX NO:12	FOREIGN TRADING SYSTEM
DATE:27/1/26	

AIM:

To draw the diagrams [usecase, activity, sequence, collaboration, class, statechart, component, deployment, package] for the Foreign trading system.

SOFTWARE REQUIREMENTS SPECIFICATION:

	SOFTWARE REQUIREMENTS SPECIFICATION
1	Hardware Requirements
2	Software Requirements
3	Problem Analysis and Project Plan
4	Project Description
5	Reference

1.0 **HARDWARE REQUIREMENTS**:

Intel Pentium Processor I3/I5

1.1 SOFTWARE REQUIREMENTS:

Rational rose / Argo UML

1.2 PROJECT ANALYSIS AND PROJECT PLANNING

The initial requirements to develop the project about the mechanism of the Foreign Trading System are bought from the trader. The requirements are analyzed and refined which enables the analyst (administrator) to efficiently use the Foreign Trading System. The complete project analysis is developed after the whole project analysis explaining about the scope and the project statement is prepared.

1.3 PROJECT DESCRIPTION:

This software is designed to maintain the details about the trading system that exists between the foreign countries. These details are hold by the trading management system. The details to the system are provided by the customer and the supplier.

1.4 REFERENCES:

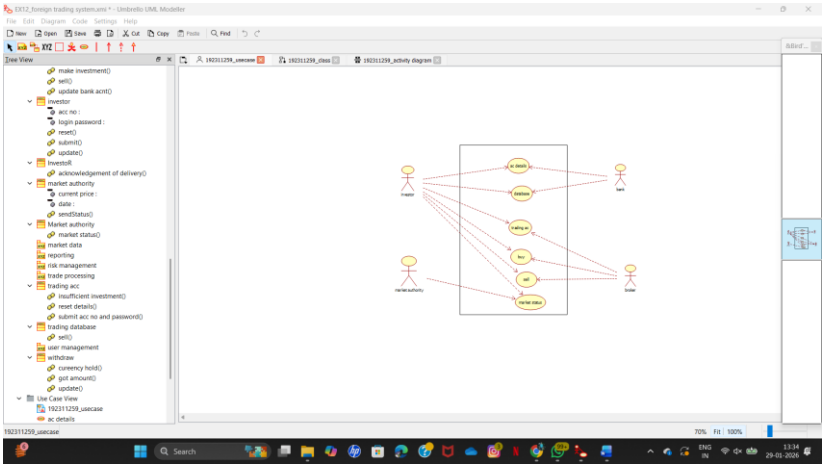
IEEE Software Requirement Specification format.

USE CASE DIAGRAM:

This diagram will contain the actors, use cases which are given below

Actors: Customer, Supplier, Custom officer

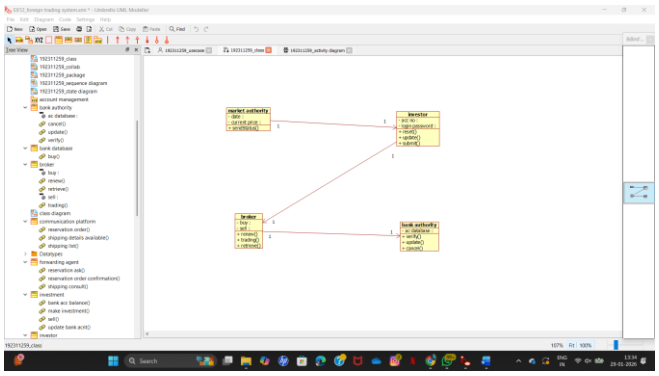
Use case: Order of product, Quantity, Specify the amount



CLASS DIAGRAM:

This diagram consists of the following classes, attributes and their operations.

CLASSES	ATTRIBUTES	OPERATIONS
Trading management system	Verify product	Transport()
Customer	Quality	Payment()
Supplier	Product supply	Money transfer()

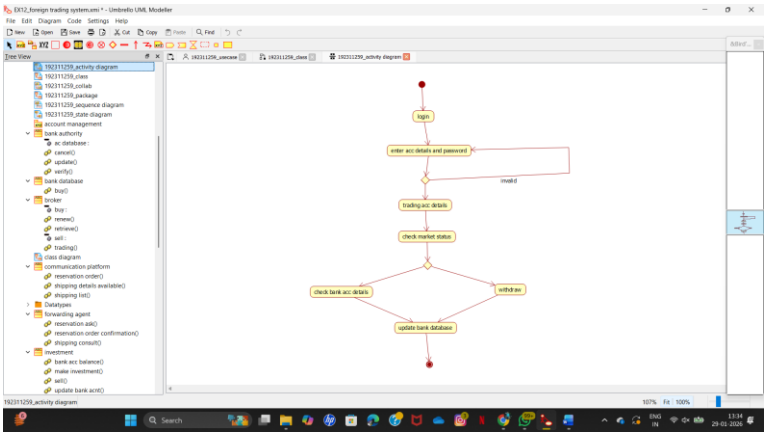


ACTIVITY DIAGRAM:

This diagram will have the activities as Start point ,End point, Decision boxes as given below:

Activities: Order of the product, Specify amount, Payment, Money transfer

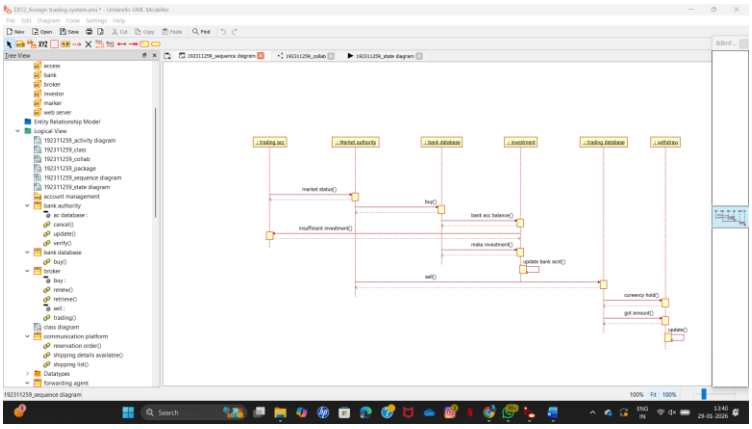
Decision box: Check for availability



SEQUENCE DIAGRAM:

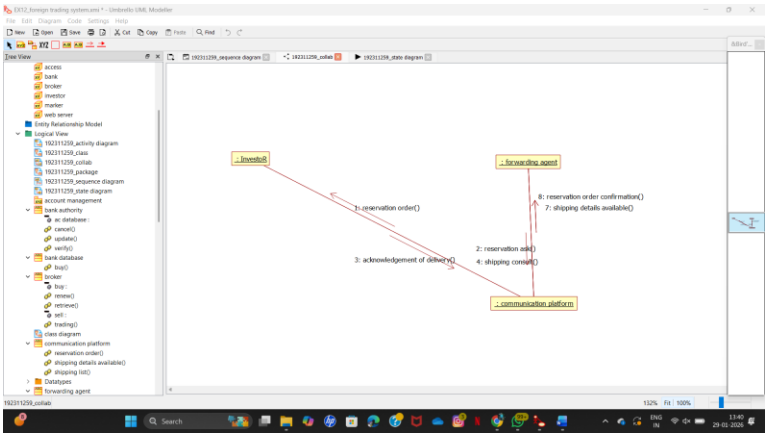
This diagram consists of the objects, messages and return messages.

Object: Customer, Supplier, Trading management system



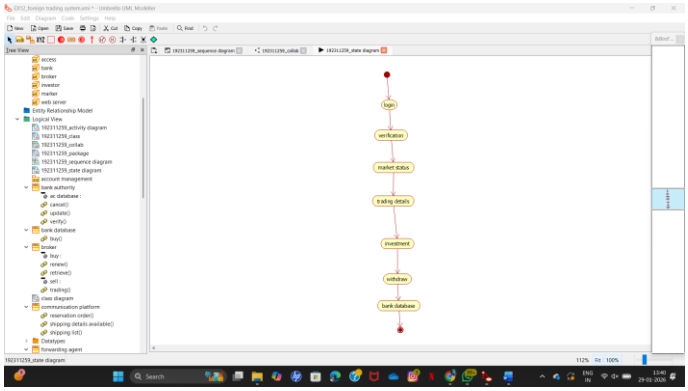
COLLABORATION DIAGRAM:

This diagram contains the objects and actors. This will be obtained by the completion of the sequence diagram and pressing the F5 key.



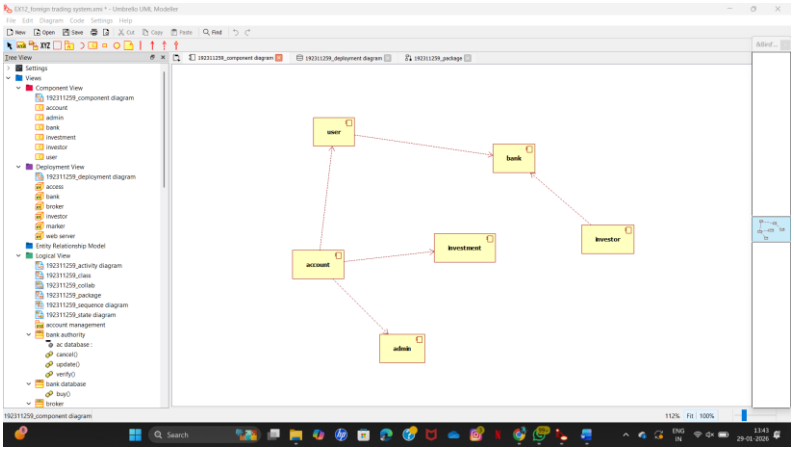
STATECHART DIAGRAM:

It is a technique to describe the behavior of the system. It describes all the possible states that a particular object gets into the object oriented technique. State diagrams are drawn for a single class to show to the lifetime behaviour of a single objects



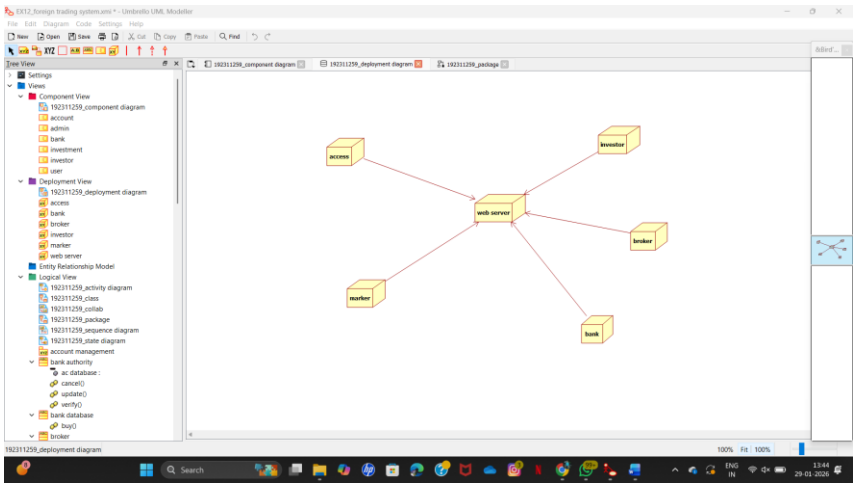
COMPONENT DIAGRAM:

The component diagram is represented by figure dependency and it is a graph of design of figure dependency. The component diagram's main purpose is to show the structural relationships between the components of a system. It is represented by a boxed figure. Dependencies are represented by a communication association.



DEPLOYMENT DIAGRAM:

A deployment diagram in the unified modeling language serves to model the physical deployment of artifacts on deployment targets. Deployment diagrams show "the allocation of artifacts to nodes according to the Deployments defined between them. It is represented by a 3- dimensional box. Dependencies are represented by communication association



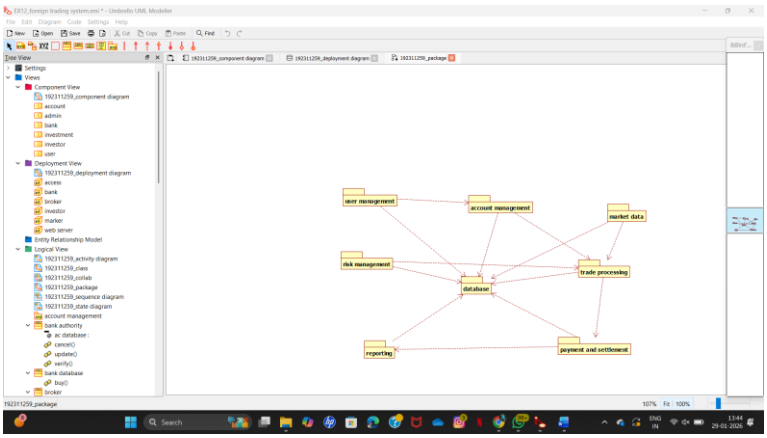
PACKAGE DIAGRAM:

A package diagram in unified modeling language that depicts the dependencies between the packages that make up a model. A Package Diagram (PD) shows a grouping of elements in the OO model, and is a Cradle extension to UML. PDs can be used to show groups of classes in Class Diagrams (CDs), groups of components or processes in Component Diagrams (CPDs), or groups of processors in Deployment Diagrams (DPDs).

There are three types of layers. They are

1. User interface layer
2. Domain layer

3. Technical services layer



PROGRAM CODING:

TRADING MANAGEMENT SYSTEM:

```
public class trading management system

{

public integer verify product;

        public integer

        amount; public void

        transport()

        {

        }

        public void money transfer()

        {

        }

}
```

CUSTOMER:

```
public class customer

{
```

```
        Public integer
        order product;
        Public integer
        amount; Public
        void payment()
        {
        }
        Public void delivery()
        {
        }
}
```

SUPPLIER:

```
Public class supplier
{
        Public integer supply;
        Public void available product()
        }
```

RESULT:

Thus the diagrams [Usecase, Activity, Sequence, Collaboration, Class, Statechart, Component, Deployment, package] for foreign trading systems have been designed, executed and output is verified.