Section 1 OBJECT ORIENTED ANALYSIS AND DESIGN LAB

1.0 Introduction

Object-oriented analysis and design (OOAD) is one of the very important courses as it helps in developing a conceptual understanding of the Analysis and Design of the system. While the study of OOAD, one needs to study a lot of problems and analyse them. Object oriented (OO) approach is though very challenging but it is very powerful approach for system development. Basic concepts like class, object, inheritance, polymorphism etc. help in designing of the system. A class in the system serve as a model for which we create objects. The interaction of objects with each other makes the system functional. Systems developed using the OO approach provide systems reusability, reliability, robustness, and extensibility. This section is based on OOAD laboratory exercises of course MCS-219. The main objective of this lab manual is to make different Unified Modeling Language (UML) diagrams. Some of the tools/software that may be used to draw UML diagrams are AgroUML, StarUML, BOUML, Dia, Visual Paradigm, Edraw-Max, Smart Draw, MS-Visio, Rational UML and IBM Rational.

1.1 OBJECTIVES

After completing this lab section, you should be able to:

- Draw different UML diagrams for specified systems,
- Implement the UML diagrams in code, and
- Do mapping of classes into database tables.

1.2 UML DIAGRAMS

UML is ready to use, expressive visual modelling language. It provides extensibility and is independent of any particular programming language. UML is used for designing Object Oriented Systems. UML is used as a documentation language. It is used for modelling the activities of project planning and related activities.

UML diagrams are used to graphically represent the related artifacts and elements such as actors, roles, classes, actions etc. of the system. UML includes the following nine diagrams which are used to represent different aspects of the systems model. Details about UML diagrams and their components can study in the MCS-219 course.

Nine UML Diagrams:

1. Class Diagram
2. Object Diagram
3. Use Case Diagram
4. Sequence Diagram
5. Collaboration Diagram
6. Statechart Diagram
7. Activity Diagram
8. Component Diagram

9. Deployment Diagram

1.3TOOLS FOR DRAWING UML DIAGRAMS

To draw UML diagrams, many tools are available. You may draw UML diagrams using simple drawing tools of MS-Words or Paint software. You may also draw UML diagrams using specialised software tools designed for this purpose. Here is the description of some tools/software which may be used for UML diagrams. `

ArgoUML: It is an open-source UML modeling tool which includes support for all standard UML diagrams, including statechart, activity diagrams, class diagrams, use-case diagrams, collaboration designs, deployment, and sequence diagrams, among others. You may run it on any Java platform, and it is available in other languages also. ArgoUML is written entirely in Java and uses the Java Foundation Classes (JFC). You can download ArgoUML (download from https://argouml.en.softonic.com/) on your computer and use it. Once downloaded, you can run the setup file and install the program on your Windows PC. You may even create a desktop shortcut so that you can launch the application anytime without any hassle. Since the application is written in Java, you may run it on any machine that has JRE (Java Runtime Environment) installed.

StarUML: It is an open-source software to develop a flexible, extensible, featureful, and freely-available UML platform. The aim of the StarUML software is to build a software modeling tool not only for UML diagrams but also for DFD and ERD. StarUML is a multi-lingual project and not tied to a specific programming language, so any programming languages can be used to develop StarUML. Examples C/C++, Java, Visual Basic, Delphi, JScript, VBScript, C# and VB.NET etc. If you want to add more and more functionalities to your developing software then this tool must have the well-defined plug-in platform and fulfil your requirements. The key features of StarUML include the followings:

- Multi-platform support (MacOS, Windows and Linux)
- UML 2.x standard compliant
- Entity-Relationship Diagram (ERD)
- Data-flow diagram (DFD)
- Flowchart diagram
- Open APIs
- Various third-party extensions

BOUML: It is UMKL diagram designer software. It runs under Windows, Linux and MacOS X operating systems. You may download BOUML from https://www.bouml.fr/ . It can be used for the automatic generation of code from the UML class diagrams and rebuild them from existing source code using the languages: C++, Java, PHP and MYSQL. It can also generate code for Python and IDL.

Visual Paradigm: It is a UML CASE Tool supporting UML 2. It is a powerful visual modeling tool that helps you to build and manage your diagrams and model elements. You may download it from https://www.visual-paradigm.com/solution/freeumltool/ .

Using any of the above software, you can draw the UML diagram given in the exercise list of this section. In this lab manual will see the demonstration of some UML diagrams drawing using StarUML software. Again it is to mention that the objective of doing the exercises in this section is to learn about UML diagrams thereof; you may use any tool/software at your convenience to draw the UML diagrams.

1.4 SOME UML DIAGRAMS

Now let us see the drawing of some UML diagrams using StarUML. To practice, you can download StarUML software from the download page http://staruml.io/download and install on your computer.

Creating Diagrams using StarUML

You can start StarUML from the Windows Start Menu. First, you will see after starting the main Project window, as shown in figure 1.1

Main Menu: The **main menu** is at the top of the screen. Most StarUML functions are accessible through the main menu.

Toolbars: The yare right below the main menu. They contain frequently used menu items. The centre part of the screen is used for creating a diagram.

Model Explorer: It is located in the upper right corner of the screen. This corner shows the model elements in hierarchical structures.

Editor: It is located in the lower right corner of the screen, which is used for editing the properties of model elements. The position of all the sections depends on the installed version of the software.

Figure 1.1: StarUML—Main Window

Your diagrams are created in the form of Project. It is a basic management unit which is a top-level element stored as a single file (.mdj). You can press Ctrl+N or select File-> New for creating a modeling project. If you already have some model files (.mdj), you can open it in StarUML by pressing Ctrl+O or select File-> Open and then select a file in the Open Dialog box. You can save the working project by pressing Ctrl+S or selecting File -> Save. If you want to save your model file as another file, press Ctrl+Shift+S or select the File -> Save As option from the menu bar.

Class Diagram

A class Diagram is a static structure diagram used to describe the structure of a system by showing the classes in the system, attributes in the classes, operations (methods) and the relationships among them.

For creating class diagram, you can select Model -> Add Diagram->| Class Diagram in the Menu Bar or select class option on one of the tools in the Toolbox. You can click in Toolbox (see figure 1.1) for an element and then click at the place within the diagram area where you want the object to be created. An instance of the type of object you selected will then appear at that point on the diagram. You can change the default name as per your requirement (see figure-1.1) and add attributes, operations to the class. In a similar manner, you can place all elements on the diagram area and create a relationship between them. The figure-1.2 shows the Student Registration class diagram. You have already studied the class diagram in the course MCS-219.

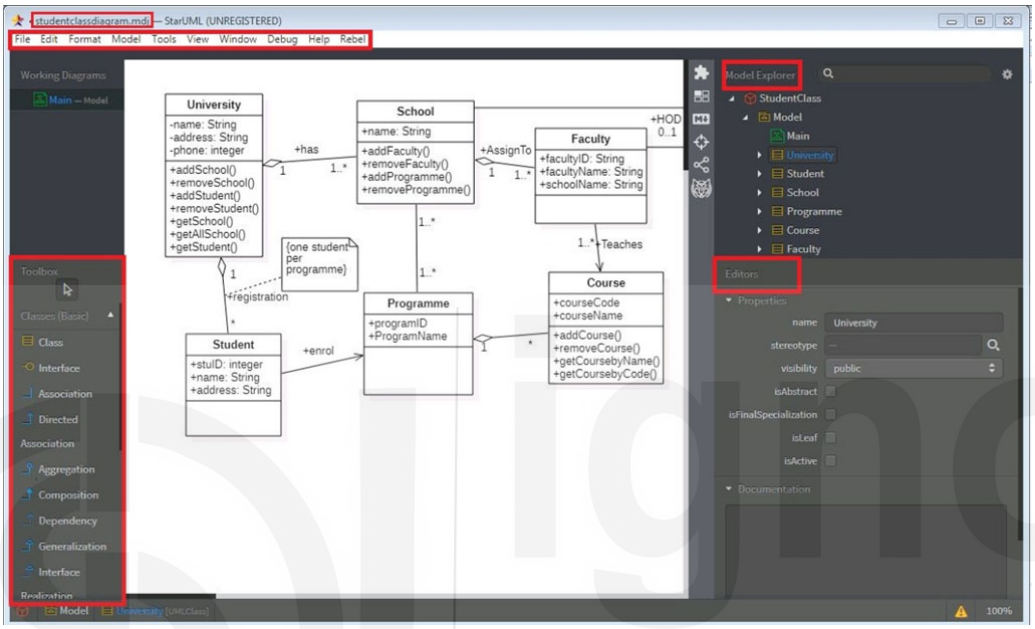

Figure 1.2: Class Diagram for Student Registration System

Figure-1.2 shows the class diagram for Student Registration System. Similarly, the figure-1.3 shows another class diagram for association between classes in which two classes, such as Schools and Universities are associated with a 'Works-for' association. You can set the properties of the elements in the Editor section.

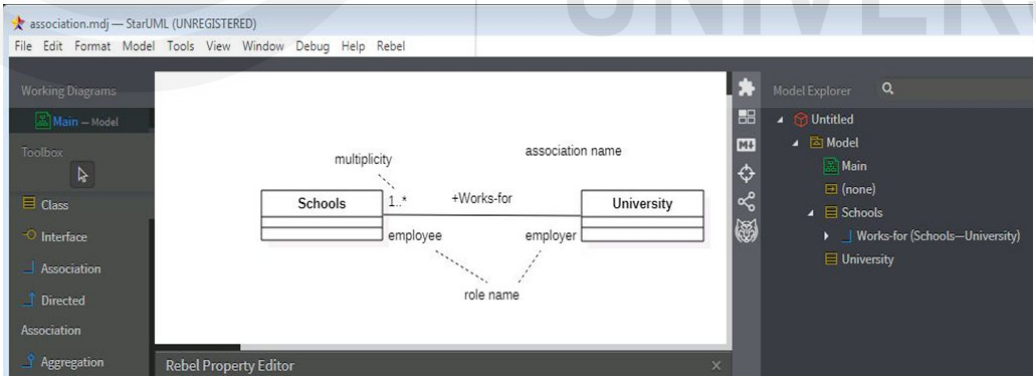

Figure 1.3: An association between classes

You can practice more by drawing the following class diagram.

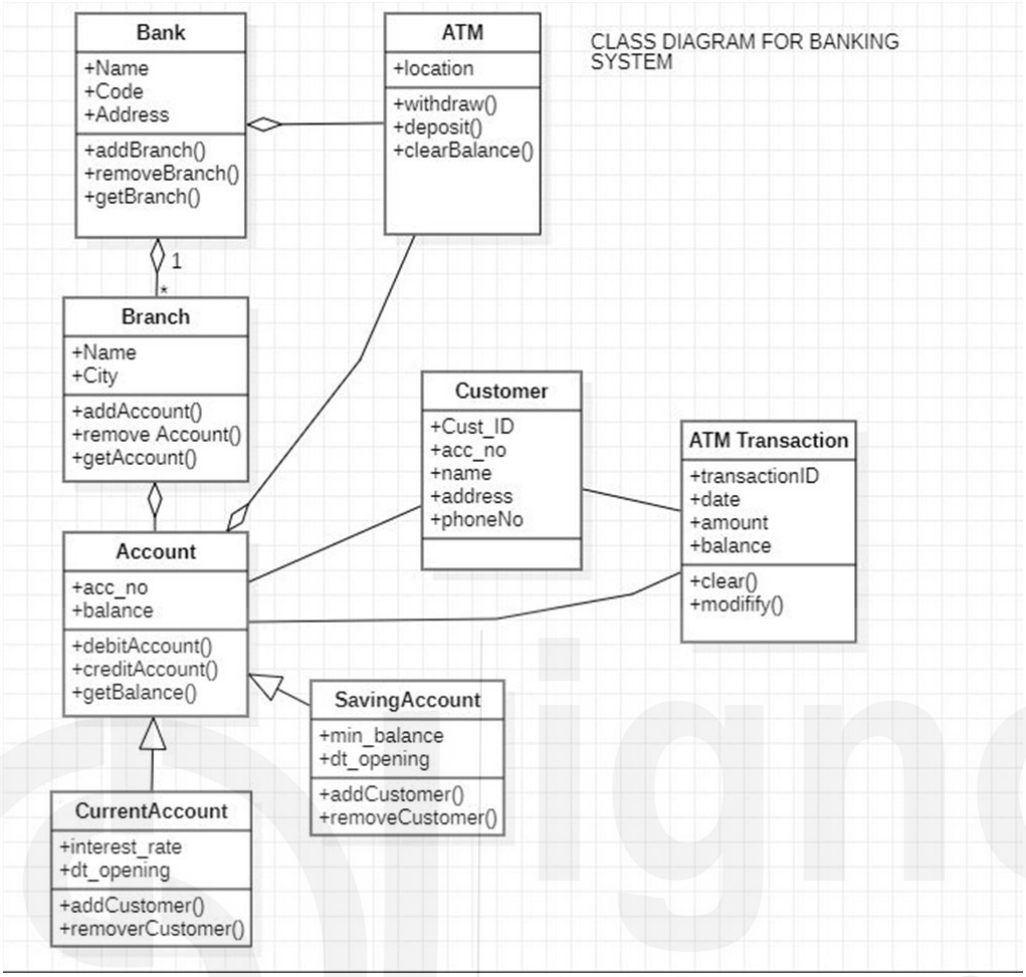

Figure 1.4 : A Typical Class Diagram for Banking System

Use Case Diagram

A use case is a description of the user's standpoint on the system working. You can say that a use case is used in system analysis to identify, clarify and organize system requirements. The use case diagram provides set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. The main component of this diagram is Actors, Use Cases and interrelationships among actors and use cases.

For creating a Use Case diagram, you can select the Model -> Add Diagram option from the main menu and then select Use Case Diagram (see Figure -1.4)

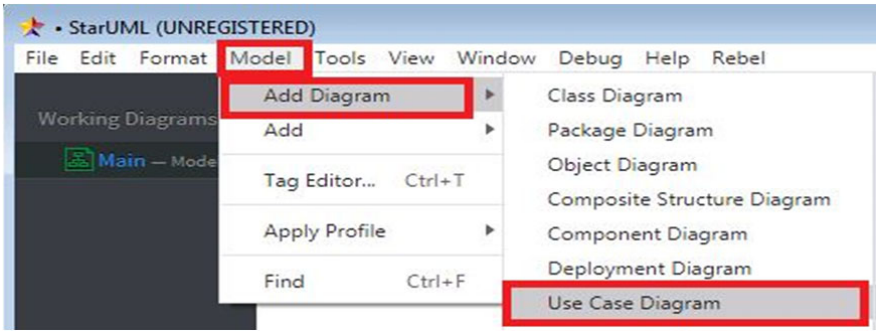

Figure 1.4: Use Case menu option

All the tools are available in Toolbox section. Now, select the 'Actor' option to create Actors such as Student and Librarian and click at the place within the diagram area where you want the object to be created. In the similar way, you can select the 'Use Case subject' option and give a name to this. After this, you can choose 'Use Case' to draw cases such as 'Borrow Book' and 'Return Book' and connect them with the Actors. Now your diagram will look like the following figure-1.5:

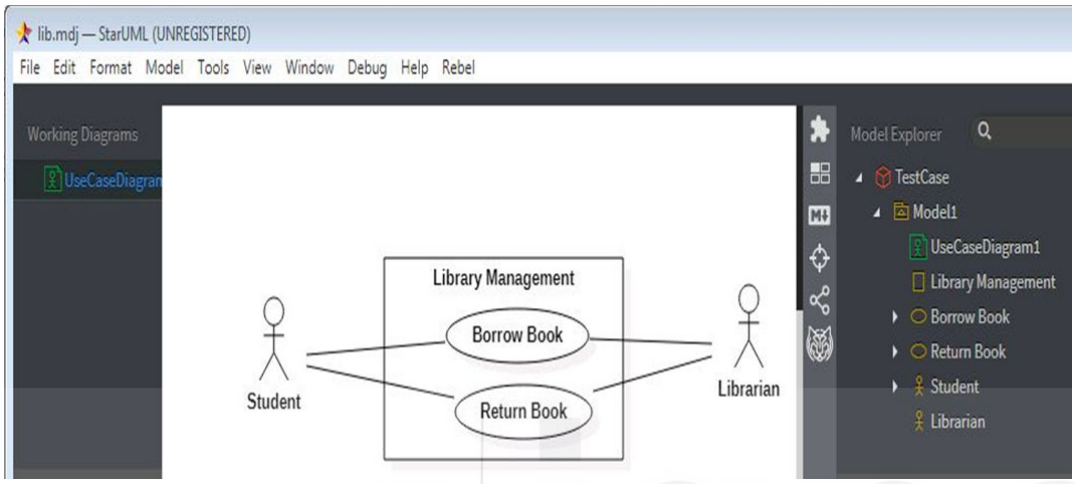

Figure 1.5: Use Case Diagram for Library Book (Borrow and Return)

You can practice more by drawing the following Use Case diagram.

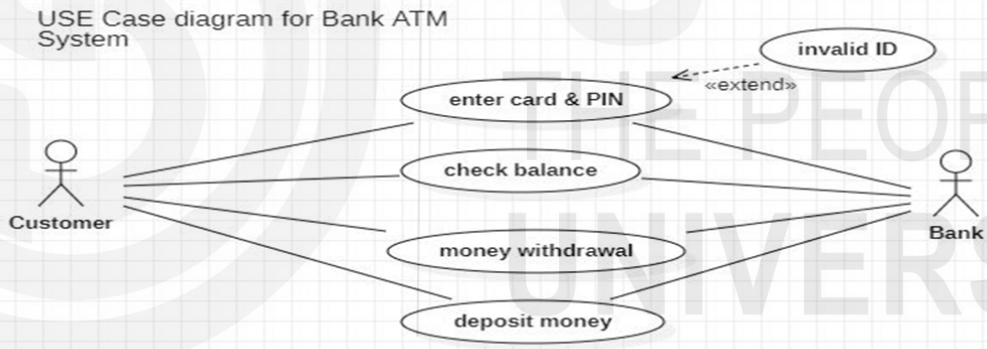

Figure 1.6 : Use Case Diagram for Bank ATM System

Activity Diagram

An activity Diagram is similar to a flowchart, used to represent the flow from one activity of the system to to another activity. The activities are nothing but the operations of the system. The control flow is drawn from one operation to another.

For creating an Activity diagram, you can select the Model -> Add Diagram option from the main menu and then select Activity Diagram. The tools related to the Activity diagram are accessible in the toolbox section. You can only select and place them on the diagram area and give a name to all elements. The following figure-6 shows the Activity diagram, which contains elements such as Initial, Action, Decision, Control Flow and Final.

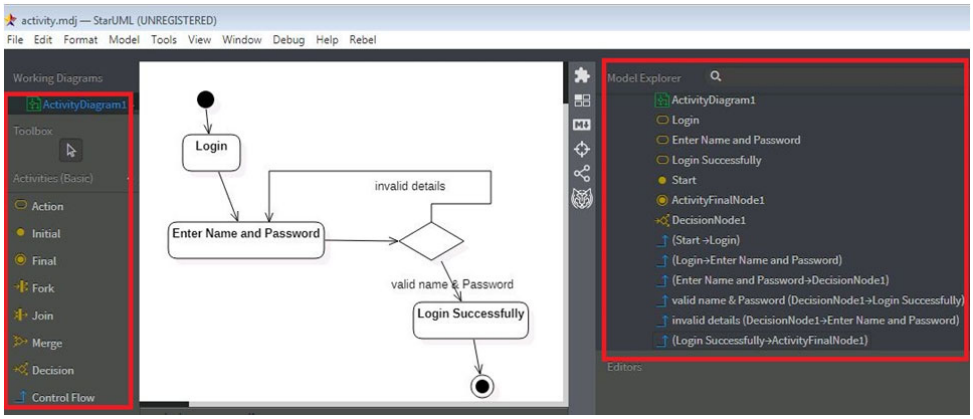

Figure 1.6: Activity Diagram

Statechart Diagram

At any given time, an object is in a particular state. This transition from one state to the next state is represented by a state diagram. For example, a person can be a newborn, infant, child, adolescent, teenager or adult. A washing machine can be either in the soaking, washing, rinsing and spinning or off state.

For creating a Statechart diagram, you can select the Model -> Add Diagram option from the main menu and then select Statechart Diagram. Now, you can select Initial State and click at the place within the diagram area where you want the object to be created. In a similar way, you can choose Simple State and Final State options from Toolbox to draw their respective elements of the Statechart diagram and assign a name to all elements of the diagram. You can see the Statechart diagram for the washing machine in the following Figure-1.7.

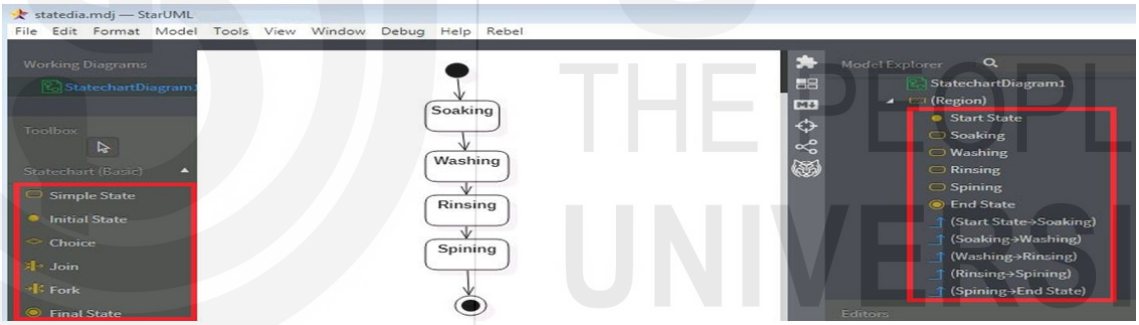

Figure 1.7: Statechart Diagram

In this way, you can easily draw other UML diagrams.

1.4.1 Some UML Diagrams for Practice

1. Use Case Diagram for Working of ATM Technical Expert.

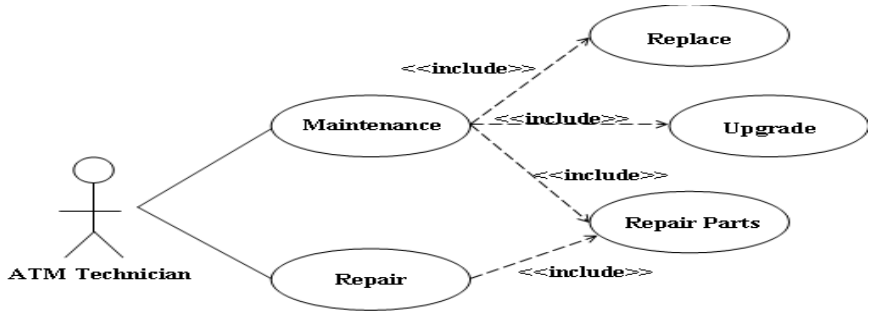

Figure 1.8: Use Case Diagram for Working of ATM Technical Expert

2. Use Case Diagram for ATM System.

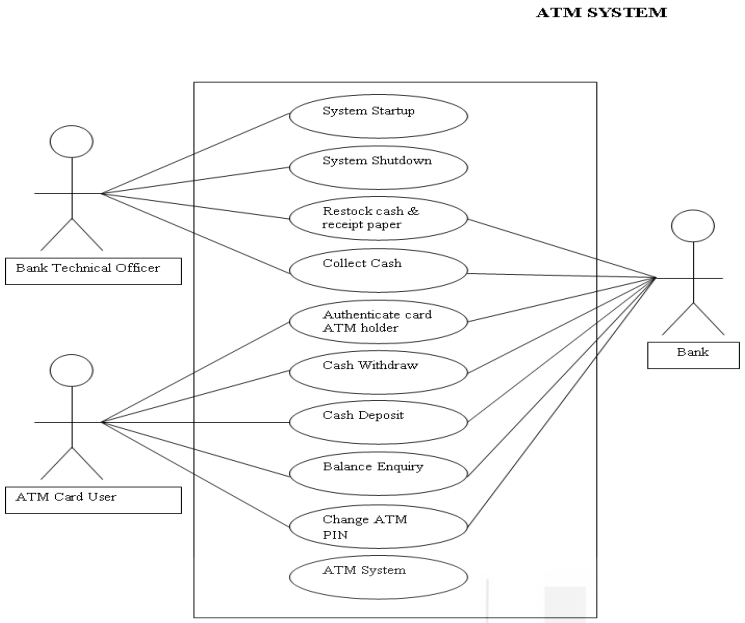

Figure 1.9: Use Case Diagram for ATM System

3. Sequence Diagram.

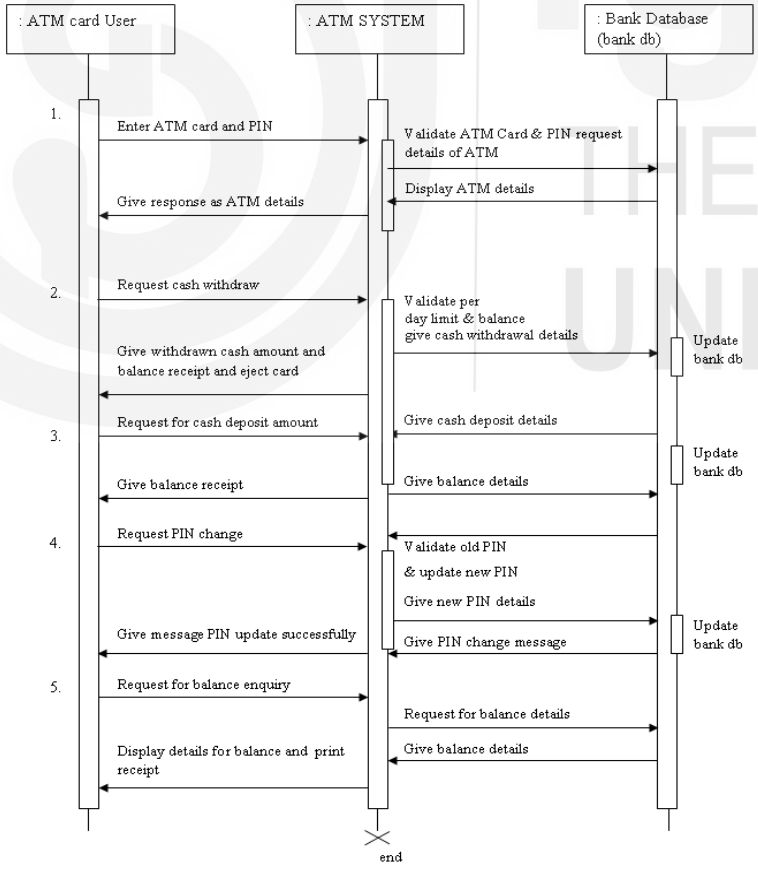

Figure 1.10: Sequence Diagram for Bank ATM Operations

2 State transition Diagram.

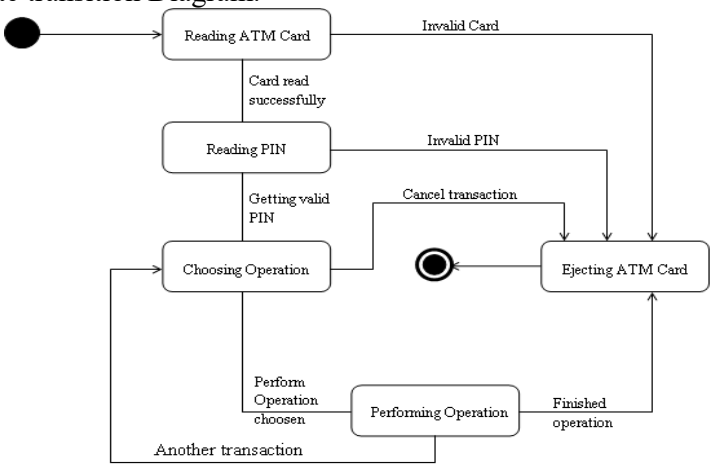

Figure 1.11: State Transition Diagram for ATM System

3. Activity Diagram.

Activity Diagram:- For Bank ATM Operations

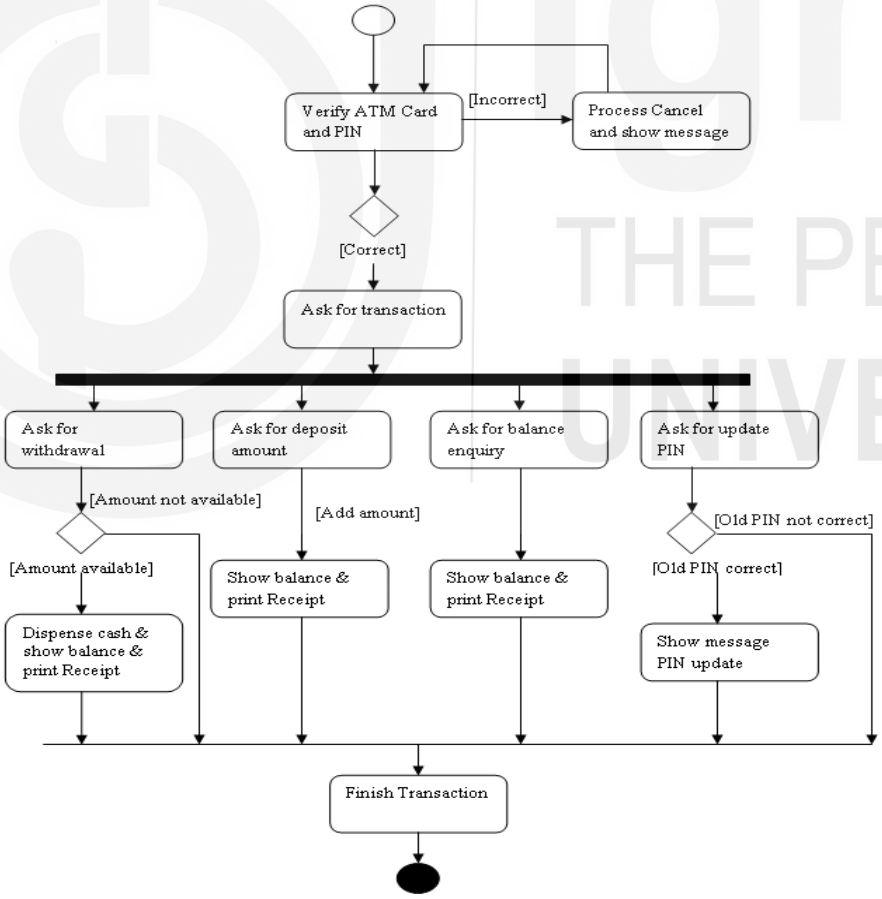

Figure 1.12: Activity Diagram for Bank ATM Operations

4. Collaboration Diagram.

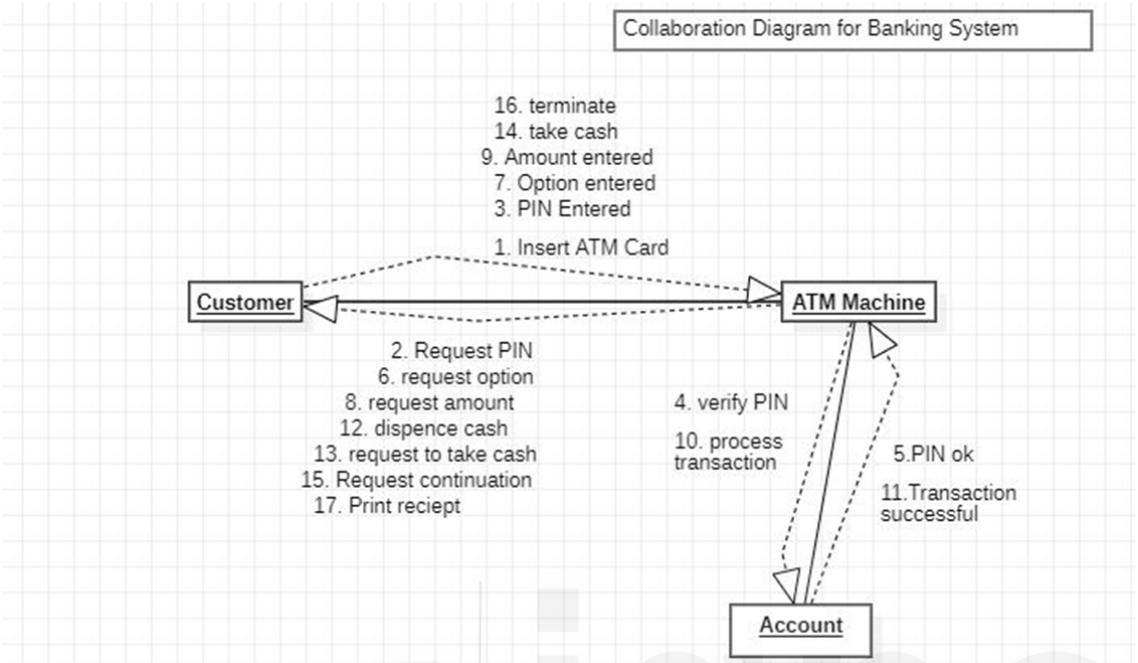

Figure 1.13: Collaboration Diagram for Banking System

5. Deployment Diagram.

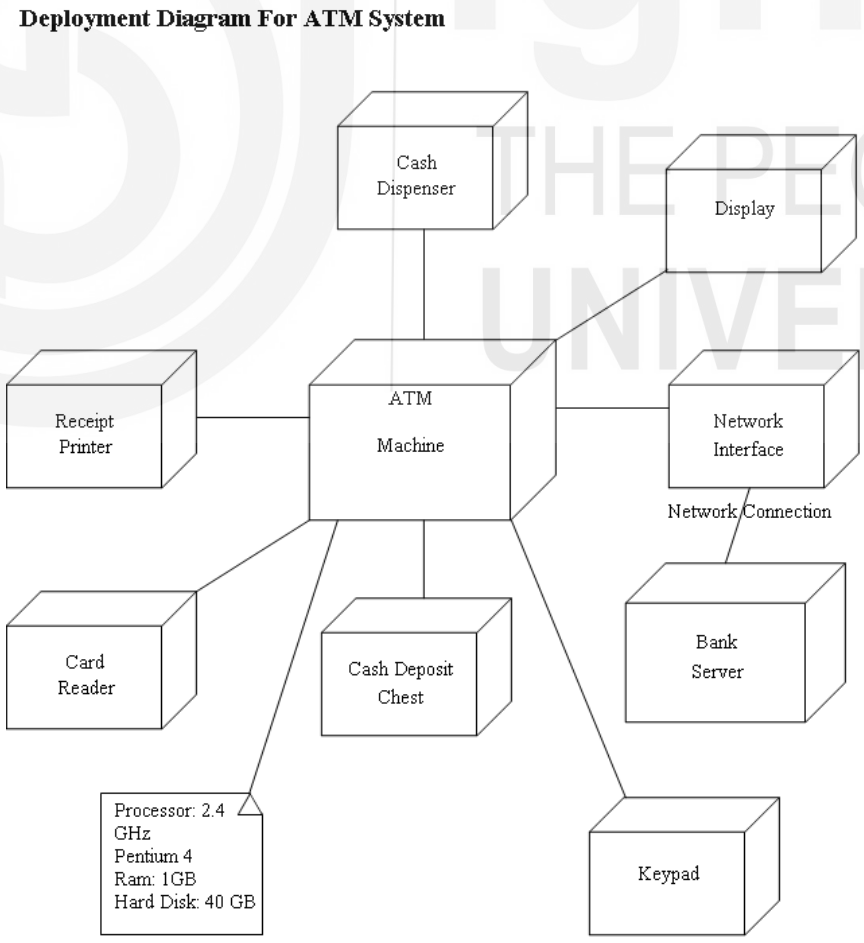

Figure 1.14: Deployment Diagram

1.5 SESSION WISE LIST OF LAB EXERCISE

This section contains a list of exercises that you should perform during your lab sessions. You need to prepare a draft solution prior to start of the session and implement it during the designated lab session. **While solving/answering each of the following exercises, you should write problems description in about 300-500 words and also list the basic assumptions related to the problem. This will help you to limit your solution within the specified scope and will also help the evaluator in evaluating your answers/solutions.**

Session-1

1. Draw Class Diagram for Online Shopping System (First, list the assumptions of the shopping system and then identify classes, and then draw the diagram).
2. Draw Class Diagram for Online Examination System.
3. Draw Object Diagram for Online Banking System.

Session-2

4. Draw Use Case Diagram for Online Shopping System.
5. Draw Use Case Diagram for Library Management System.

Session-3

6. Draw Sequence Diagram for Online Shopping System.
7. Draw Sequence Diagram for Online Examination System.
8. Draw a Sequence diagram for Employee Management System.

Session-4

9. Draw Collaboration Diagram for Student Registration Process in Masters Program of a University.
10. Draw Collaboration Diagram for Online Banking System.
11. Draw a Collaboration diagram for Employee Management System.

Session-5

12. Activity Diagram Online Banking System.
13. Draw Activity Diagram for Online Examination System.

Session-6

14. State Chart Diagram for Online Examination System.
15. State Chart Diagram for Online Fund Transfer through Netbanking Banking System.

Session-7

16. Draw Component Diagram for Online Examination System.
17. Draw Component Diagram for Order Processing Application.
18. Diagram Deployment Diagram for Online Student Admission System.

Session-8

19. Implement the following Class Diagram in C++/Java.

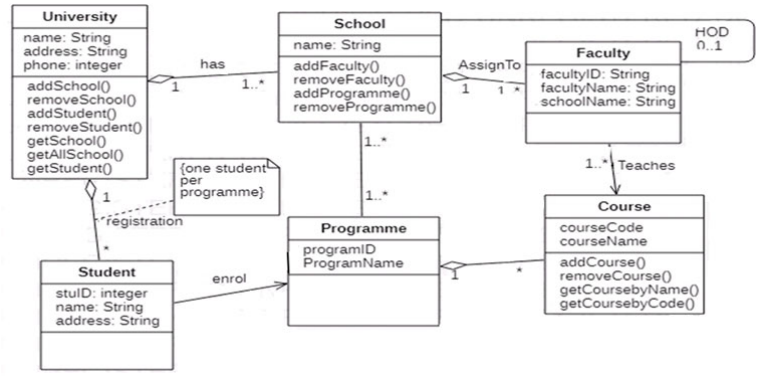

Figure 1.15: Class Diagram for Student Registration

20. Implement the Class Diagram of figure 1.18, in C++ or Java.

Session-9

21. Implement the following Associations using C++/Java.

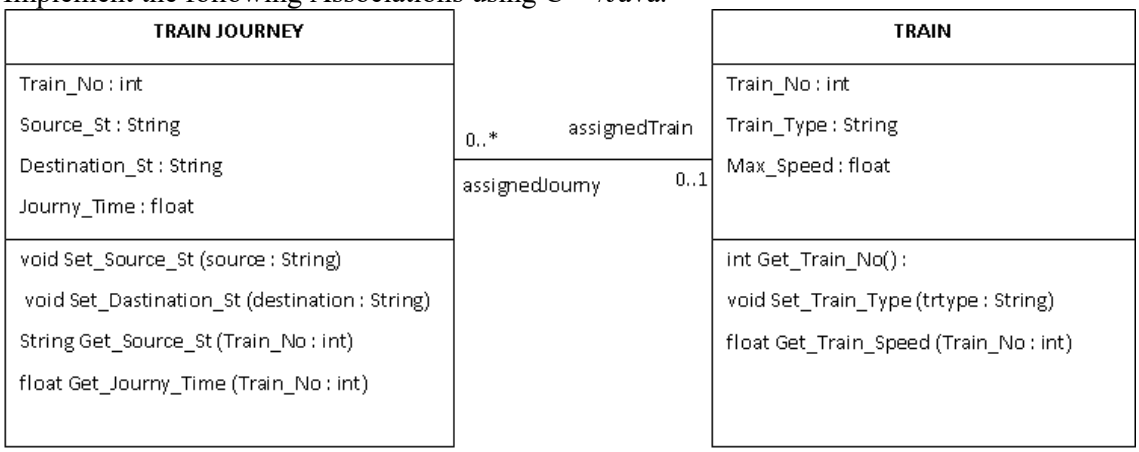

Figure1.16: Train-Journey Association

22. Implement the following Associations using C++/Java.

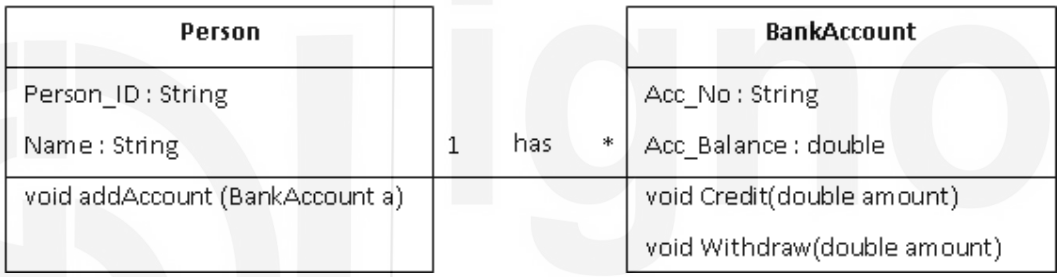

Figure 1.17: Person and Bank Account

Session-10

23. Do mapping of the following Classes into database tables

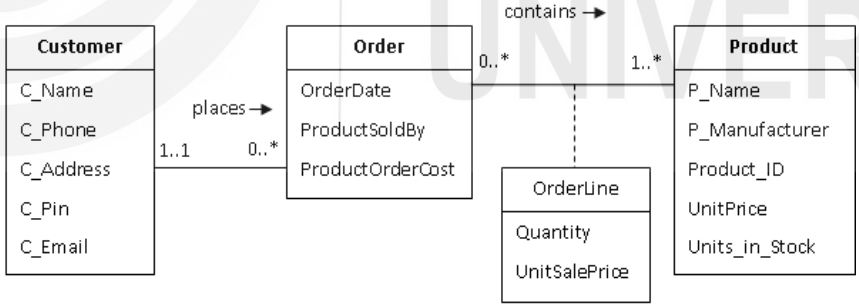

Figure1.18: Customer Order Association Class

1.6 REFERENCES/FURTHER READINGS

- Grady Booch, James Rumbaugh and Ivar Jacobson, "The Unified Modeling Language User Guide", Pearson Education,2004.
- https://argouml.en.softonic.com/
- https://argouml-tigris-org.github.io/tigris/argouml/
- https://staruml.io/
- https://docs.staruml.io/
- https://docs.staruml.io/user-guide/editing-elements

- http://staruml.sourceforge.net/docs/user-guide(en)/toc.html
- https://www.bouml.fr/
- https://docplayer.net/21262400-Staruml-documentation.html
- https://en.wikipedia.org/wiki/List_of_Unified_Modeling_Language_tools
- https://www.uml.org/
- https://www.uml-diagrams.org/index-examples.html
- https://www.uml-diagrams.org/classifier.html
- https://www.uml-diagrams.org/dependency.html
- https://www.uml-diagrams.org/association.html
- https://www.uml-diagrams.org/package-diagrams-overview.html
- https://www.uml-diagrams.org/package-diagrams.html