EXP.NO:01 **DATE:24-01-24**

STUDY OF STAR UML

AIM:

To study the basics of STAR UML.

INTRODUCTION:

Star UML is an open-source software modelling tool that supports the UML (Unified Modelling Language) framework for system and software modelling. It is based on UML version 1.4, provides eleven different types of diagrams, and accepts UML 2.0 notation, It actively supports the MDA (Model Driven Architecture) approach by supporting the UML profile concept and allowing to generate code for multiple languages.

SYSTEM REQUIREMENTS:

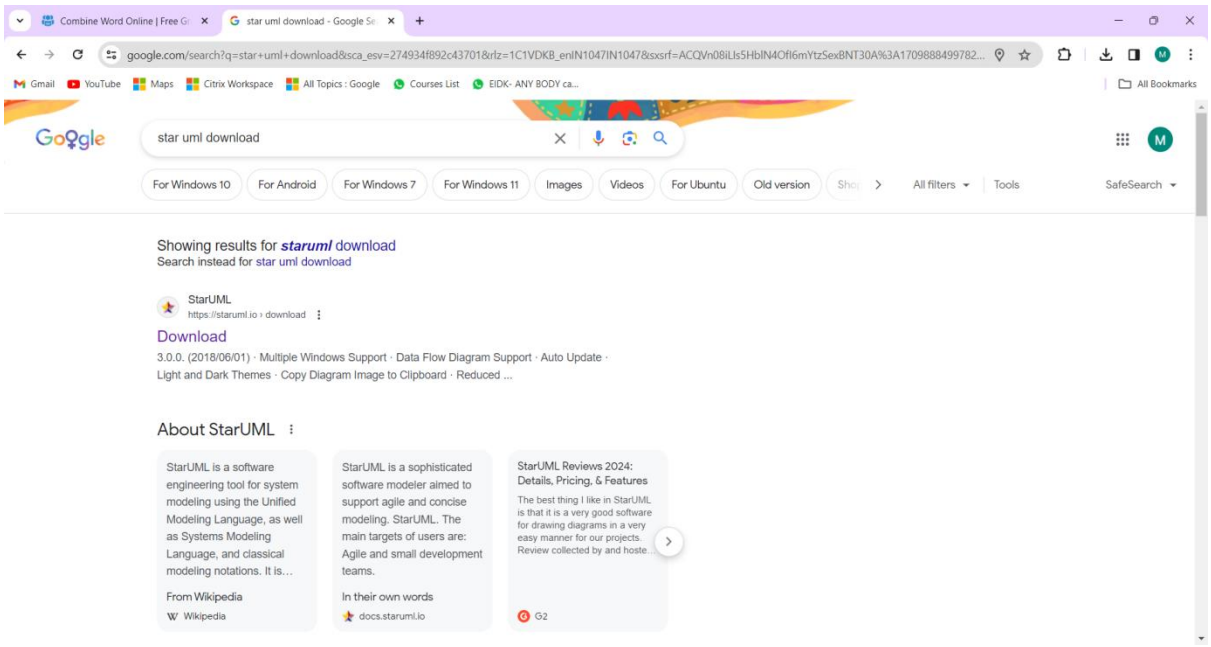
Windows 2000, Windows XP, or higher; Microsoft Internet Explorer 5.0 or higher; 128 MB RAM(256MB recommended); 110MB hard disc space (150MB space recommended).

INSTALLATION:

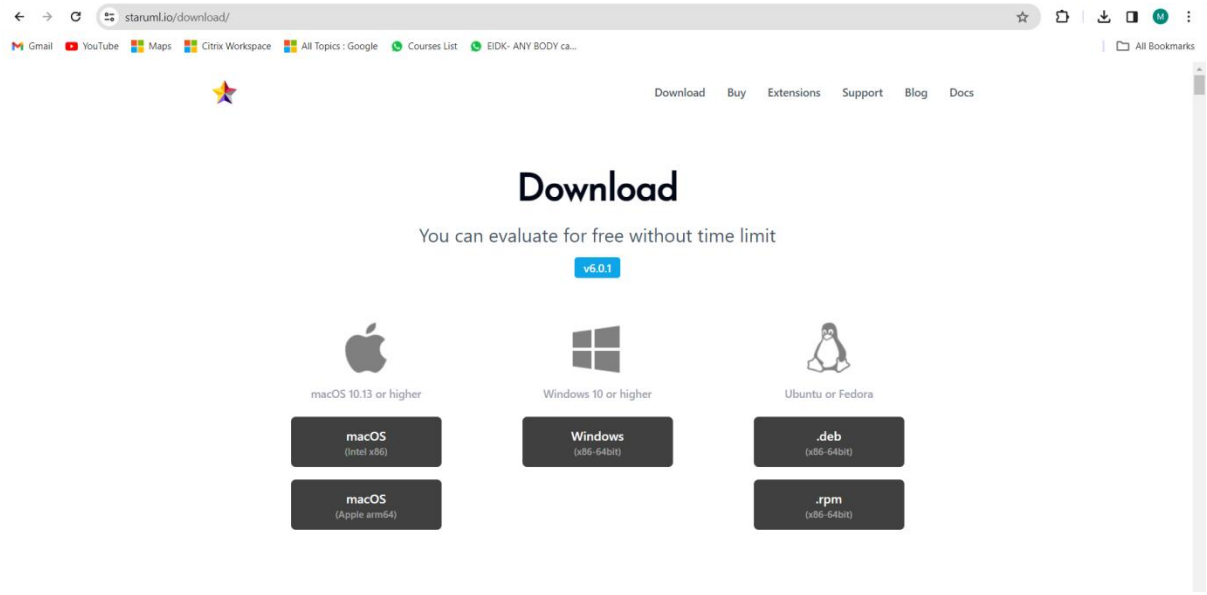
The installer follows the classic Windows install procedure without issues.

STEPS FOR DOWNLOADING STAR UML:

STEP1: Search for Star UML in Google.



STEP 2 : Download for Mac OS/Windows/.deb, .rpm.



STEP3: Install it

DOCUMENTATION:

The same help that could be browsed on the StarUML website is available with the tool on your desktop Documentation describes the concepts of the tool but on high-level vision. More detailed documentation is available for the diagramming functions. Sample projects are provided with the tool and one of them contains the model of the tool itself, showing that the developers were able to eat their dog food. Besides English, documentation exists in Korean, Japanese, and Russian.

CONFIGURATION:

Some general and diagram configuration options are available from the Tools/Options menu. You will find in this window also the configuration switches for the code generation. The interface is also very configurable as you can select what part of the tool you would like to view or not.

FEATURES:

When you start a new project, StarUML proposes which approach you want to use:4+1 (Kruchten).Rational, UML, components (from Cheesman and Daniels book), default or empty. Depending on the approach, profiles and/or frameworks may be included and loaded. If you don't follow a specific approach, the "empty" choice could be used. Although, a project can be managed as one file, it may be convenient to divide it into many units and manage them separately if many developers are working on it together.

StarUML makes a clear conceptual distinction between models, views, and diagrams. A Model is an element that contains information for a software model. A View is a visual expression of the information contained in a model, and a Diagram is a collection of view elements that represent the user's specific design thoughts.

StarUML is built as a modular and open tool. It provides frameworks for extending the functionality of the tool. It is designed to allow access to all functions of the model/meta-model and tool through COM Automation, and it provides an extension of menu and option items. Also, users can create their own approaches and frameworks according to their methodologies. The tool can also be integrated with any external tools.

1)Use Case Diagram

2)Class Diagram

3)Sequence Diagram

4)Collaboration Diagram

5)State chart Diagram

6)Activity Diagram

7)Component Diagram

8)Deployment Diagram

9)Composite Structure Diagram.

RESULT:

Thus, the basics of star UML have been studied successfully.

EX.NO: 02 **DATE:31-01-24**

ONLINE RAILWAY RESERVATION SYSTEM

AIM:

To develop the Passport Automation System using rational rose tools, visual basic and MS access.

PROBLEM ANALYSIS AND PROJECT PLANNING:

INTRODUCTION:

This document deals with online ticket reservation for railway. This document is designed in such a way that the reader understands it. The use case description and other documents are described in such a way that the system reaches the people easily.

OBJECTIVES:

The purpose of the document is to know about the availability of seats, railway etc. According to the requirements, passenger reserves his/her tickets.

SCOPE:

This document for online ticket reservation for airline makes the work easy for the passenger to book tickets.

PROBLEM STATEMENT:

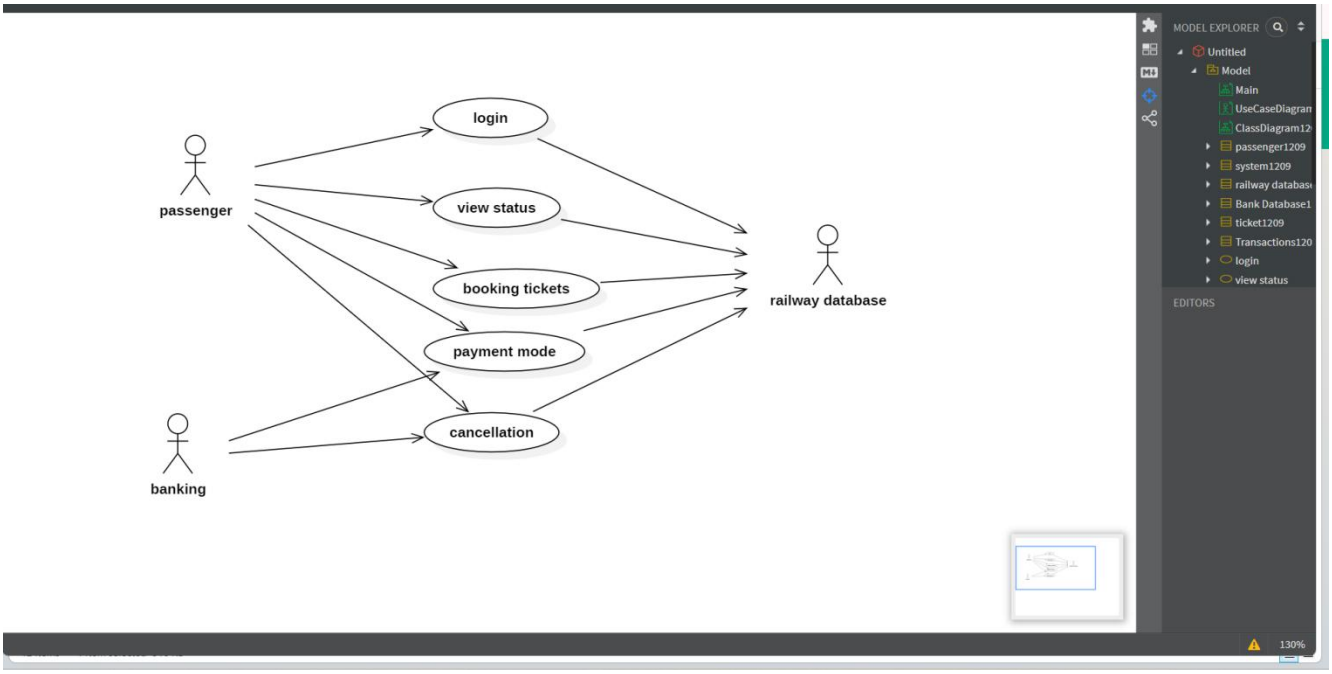
Computers are an integral part of our day today life. It makes the work easy and faster. Every job is computerized now. So is the ticket reservation, we can book our tickets online. During the reservation of tickets the passenger has to select the origin, date of journey, passport number, etc.

The reservation counter keeps track of the passenger's information. The system will have all the details about the trains and the facilities provided by them. There are various trains

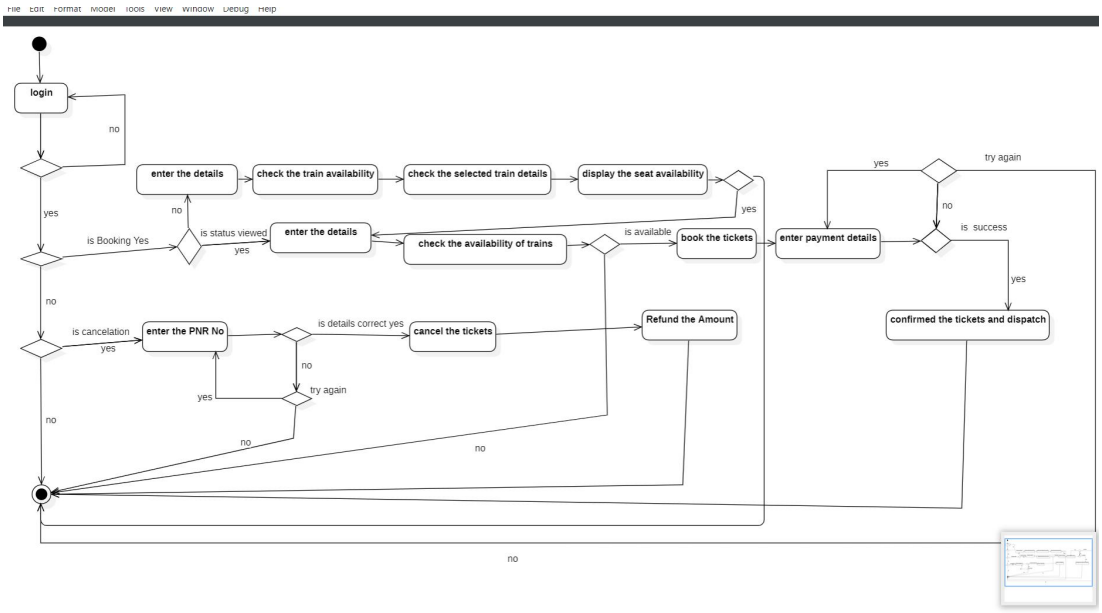
provided according to the convenience of the passenger. A database is maintained by the database administrator.

STAR UML DIAGRAMS:

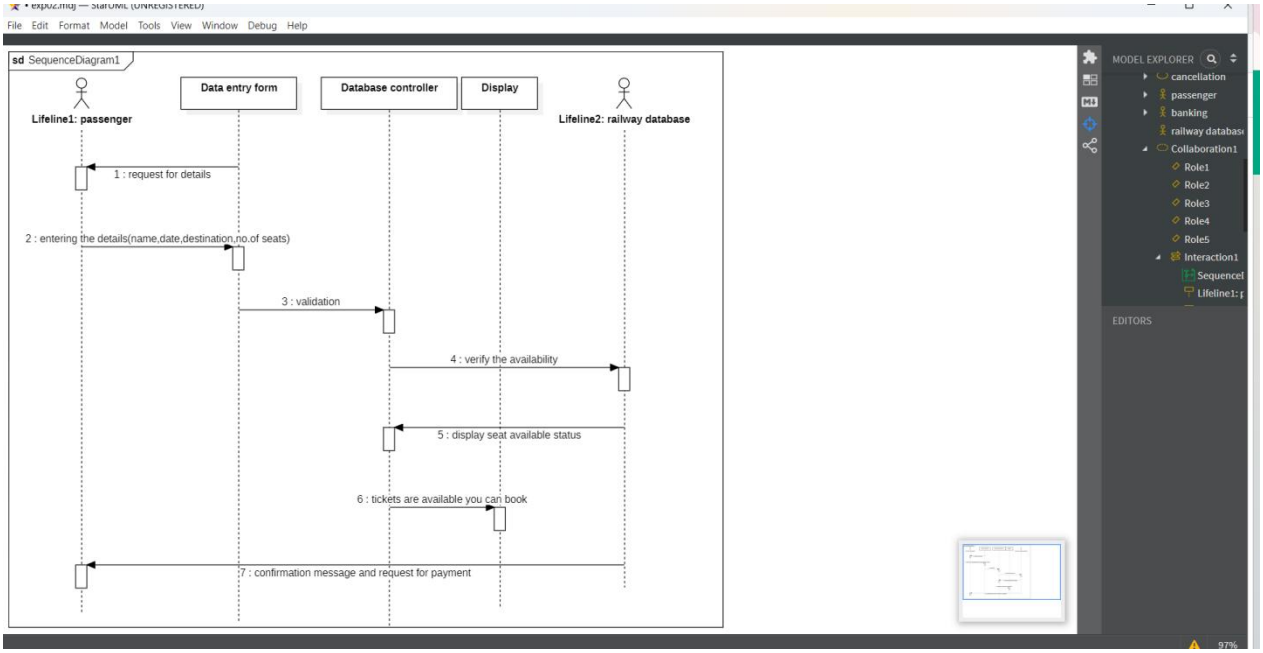
USE CASE DIAGRAM:



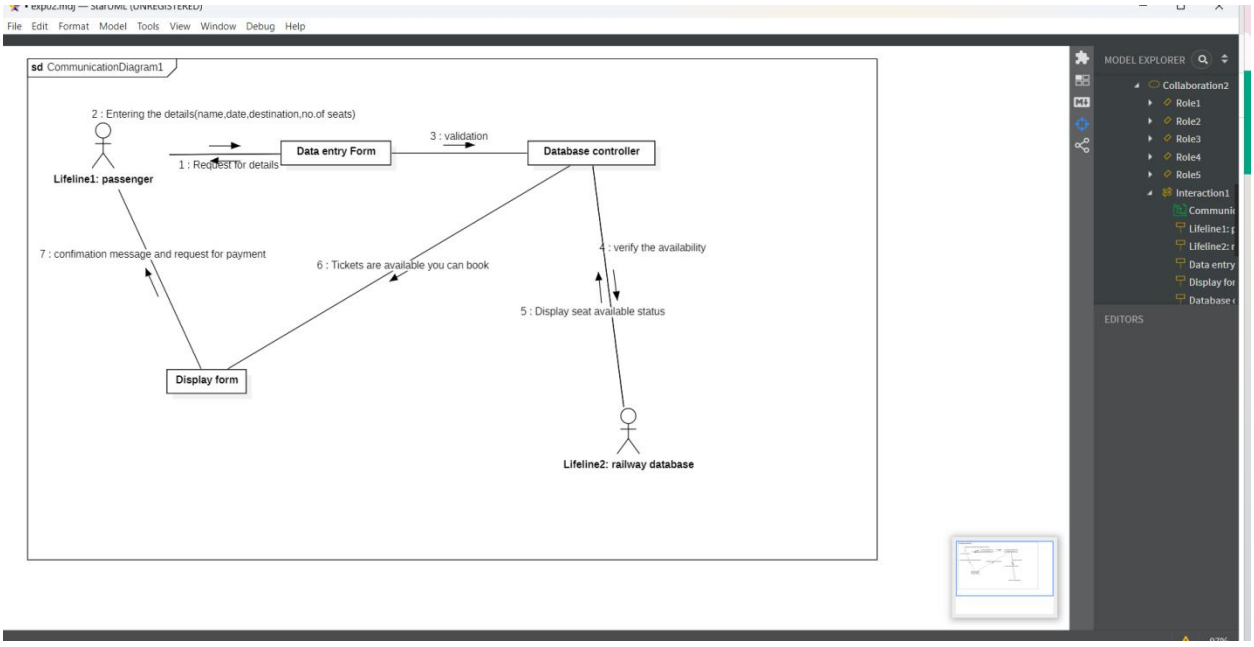
ACTIVITY DIAGRAM:



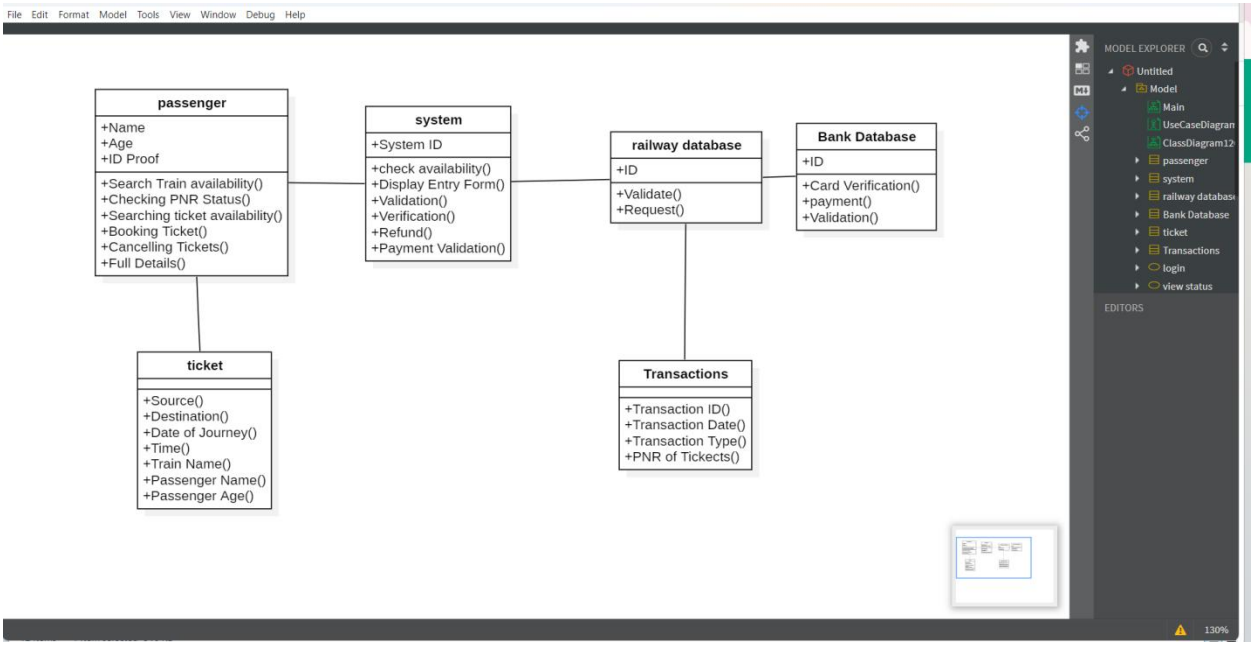
SEQUENCE DIAGRAM:



COMMUNICATION DIAGRAM:



CLASS DIAGRAM:



STEPS TO GENERATE CODE IN STAR UML:

) Draw the class diagram and then save it.

2) Open the saved file with star uml.

3) Then go to tools at right corner.

4) Select the code language which we can prefer.

5) Then we have the code generator in tools so give access to it.

6) Then go to tools and select the language then it will take us to another window.

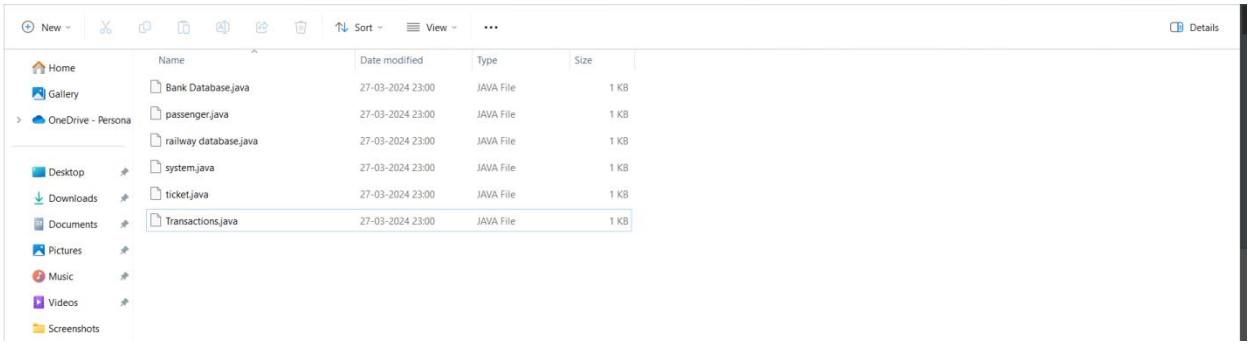
7) There, select the class attribute to generate the code.

8) Then it will ask for the directory to store the code.

9) Select the directory and click on generate code.

10) Then code generated successfully

OUT PUT:



PASSENGER:

```
import java.util.*;

public class passenger {

  public passenger () {

    }

       public void Name;

       public void Age;

       public void ID Proof;

             public void Search Train availability() {

       // TODO implement here

    }

             public void Checking PNR Status() {

       // TODO implement here

    }

             public void Searching ticket availability() {
```

```
        // TODO implement here
    }
            public void Booking Ticket() {
        // TODO implement here
    }
            public void Cancelling Tickets() {
        // TODO implement here
    }
            public void Full Details() {
        // TODO implement here
    }
}
```

BANK DATABASE:

```
import java.util.*;
public class Bank Database  {
    public Bank Database () {
    }
    public void ID;
    public void Card Verification() {
        // TODO implement here
    }
```

```
    public void payment() {
        // TODO implement here
    }
    public void Validation() {
        // TODO implement here
    }
}
```

RAILWAY DATABASE:

```
import java.util.*;
public class railway database  {
    public railway database {
    }
    public void ID;
    public void Validate() {
        // TODO implement here
    }
    public void Request() {
        // TODO implement here
    }
}
```

SYSTEM:

```
import java.util.*;

public class system  {

   public system () {

   }

   public void System ID;

   public void check availability() {

      // TODO implement here

   }

   public void Display Entry Form() {

      // TODO implement here

   }

   public void Validation() {

      // TODO implement here

   }

   public void Verification() {

      // TODO implement here

   }

   public void Refund() {

      // TODO implement here

   }
```

```
    public void Payment Validation() {
        // TODO implement here
    }
}
```

TICKET:

```
import java.util.*;
public class ticket  {
    public ticket () {
    }
    public void Source() {
        // TODO implement here
    }
    public void Destination() {
        // TODO implement here
    }
    public void Date of Journey() {
        // TODO implement here
    }
    public void Time() {
        // TODO implement here
    }
```

```
    public void Train Name() {
        // TODO implement here
    }
    public void Passenger Name() {
        // TODO implement here
    }
    public void Passenger Age() {
        // TODO implement here
    }
}
```

TRANSACTIONS:

```
import java.util.*;
public class Transactions  {
    public Transactions () {
    }
    public void Transaction ID() {
        // TODO implement here
    }
    public void Transaction Date() {
        // TODO implement here
    }
```

```
    public void Transaction Type() {
        // TODO implement here
    }
    public void PNR of Tickects() {
        // TODO implement here
    }
}
```

RESULT:

Thus, the UML diagrams for Online Reservation System in Star UML has been designed and java code the template was generated for coding successfully.

EXP.NO:03 **DATE:03-02-24**

STUDENT MARK ANALYSIS SYSTEM

AIM:

To develop the Student Mark Analysis System using StarUML.

PROBLEM ANALYSIS AND PROJECT PLANNING

INTRODUCTION:

Student mark analysing system has been designed to carry out the mark analysis process in an educational institution. The results of respective departments can be efficiently computed without much manual involvement.

OBJECTIVES:

The purpose of this project is to define the requirements of the mark analysis system. This system reduces manual work to a great extent. The mark analysis is carried out by the system in an efficient manner.

SCOPE:

This system is very essential for every educational institution as it reduces manpower. This system can be used by all kinds of educational institutions to evaluate and analyse the marks and generate reports of specified criteria.

PROBLEM STATEMENT:

For analysing the marks obtained by students in an educational institution. We are tasked to build up a student mark analysing system. This is done to replace the manual entering and processing of marks which are error-prone and tedious. This system also maintains information about students. The system will have a Windows-based desktop interface to allow the faculty to enter marks obtained by the students, update them and generate various reports.

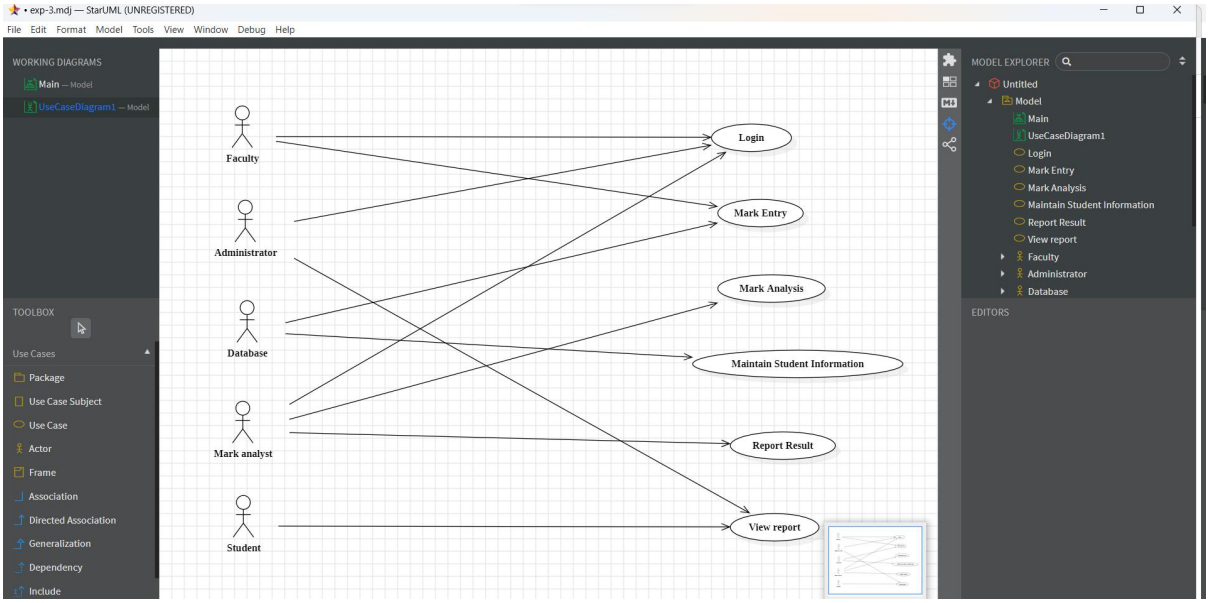
For security reasons, the administrator and faculty only can update the marks and other information. First, the user needs to login into to the system for accessing it. The system will retain information on all the students and the institution. The system analyses the marks and generates the result reports. The marks and information about the students are stored in a

database and the system works with the database. The faculty can enter the marks and student information through a visual environment. The updated details are stored in the database.

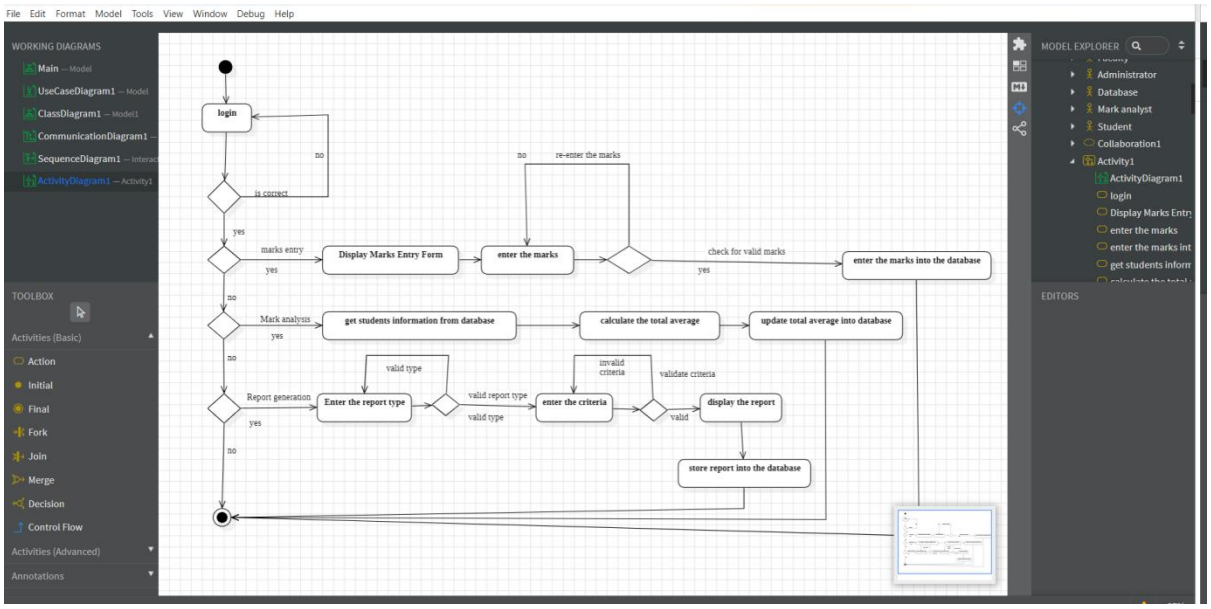
The system generates the overall result by analysing the marks. Mark analyser monitors this process. One of the most important features of the system is creating reports based on the given criteria. The user can create the following reports.

STAR UML DIAGRAMS:

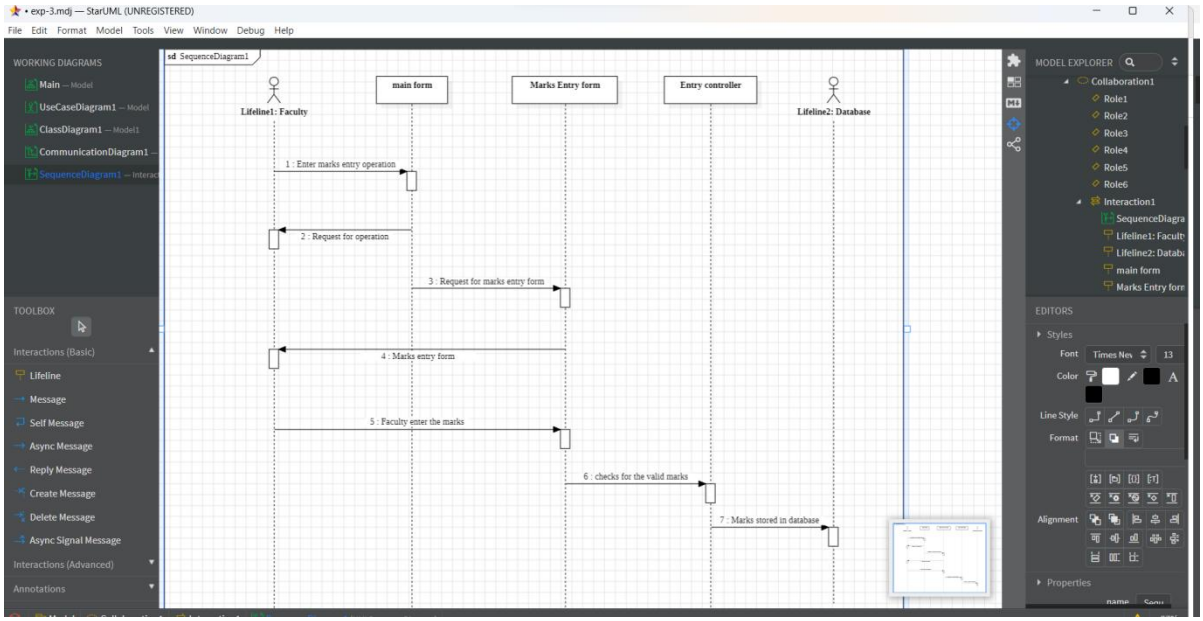
USE CASE DIAGRAM:



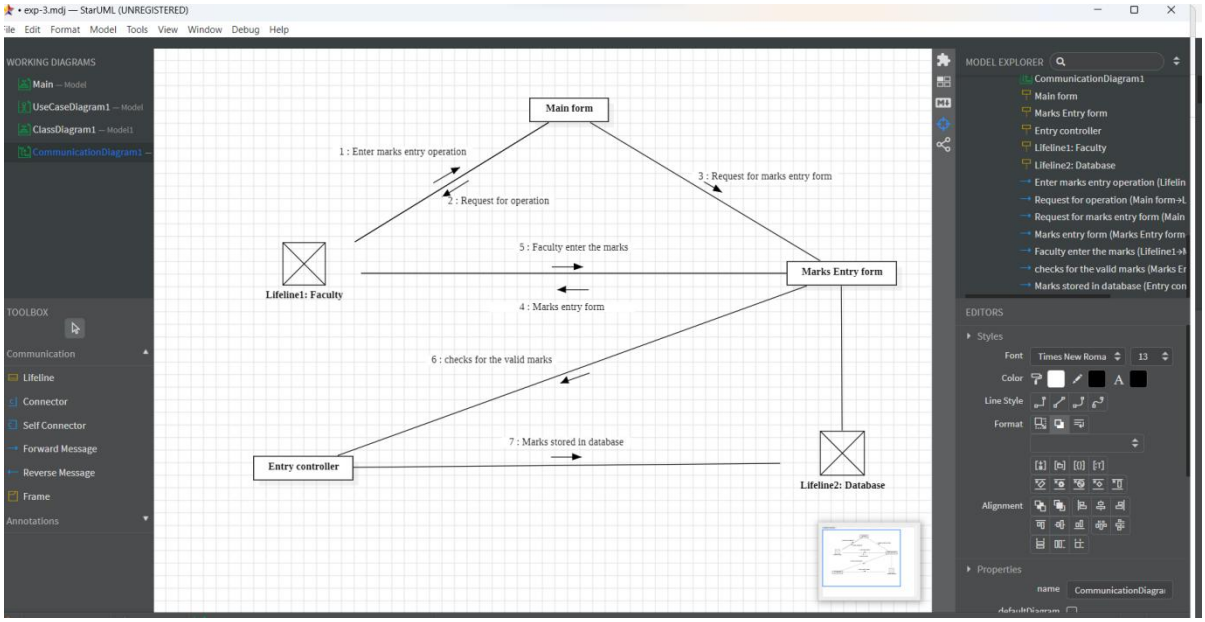
ACTIVITY DIAGRAM:



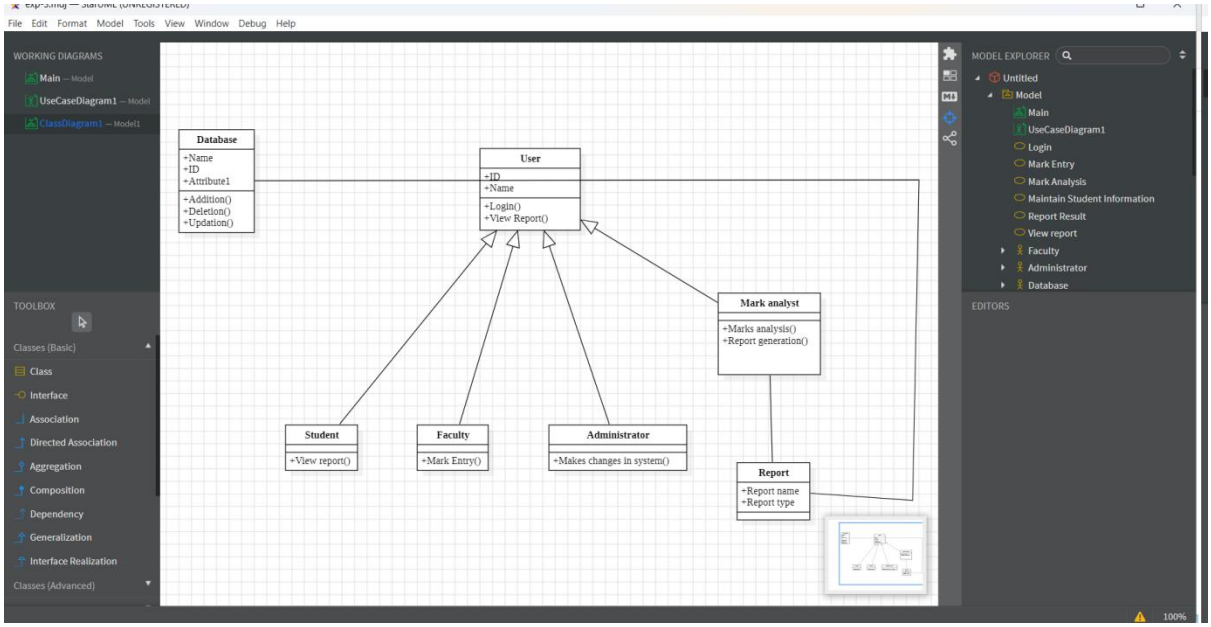
SEQUENCE DIAGRAM:



COMMUNICATION DIAGRAM:



CLASS DIAGRAM:



ALGORITHM:

1. Draw the class diagram and then save it.

2. Open the saved file with star uml.

3. Then go to tools at right corner.

4. Select the code language which we can prefer.

5. Then we have the code generator in tools so give access to it.

6. Then go to tools and select the language then it will take us to another window.

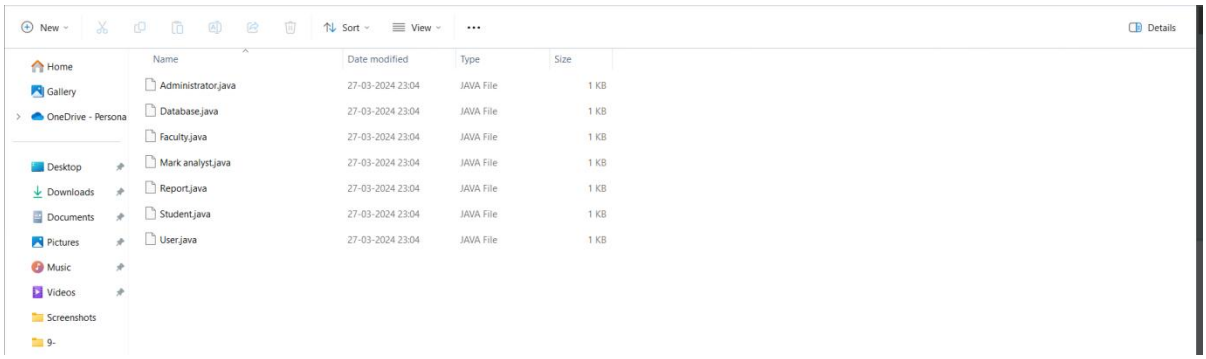
7. There select the class attribute to generate the code.

8. Then it will ask for the directory to store the code.

9. Select the directory and click on generate code.

10. Then code generated successfully.

OUTPUT:



ADMINISTRATOR:

```
import java.util.*;

/**

*

*/

public class administrator extends user {

    /**

    * Default constructor

    */

   public administrator() {

   }

    /**

    *

    */

   public void makes changes in systems() {

       // TODO implement here

   }
```

```
}
```

DATABASE:

```
import java.util.*;
/**
 *
 */
public class database {
    /**
    * Default constructor
    */
   public database() {
   }
    /**
    *
    */
   public void Name;
   /**
    *
    */
   public void id;
    /**
    *
    */
   public void addition() {
```

```
        // TODO implement here
    }
     /**
     *
     */
    public void deletion() {
        // TODO implement here
    }
     /**
     *
     */
    public void updation() {
        // TODO implement here
    }
}
```

FACULTY:

```
import java.util.*;
/**
 *
 */
public class faculty extends user {
     /**
     * Default constructor
     */
```

```
    public faculty() {
}
     /**
     *
     */
    public void marks entry() {
        // TODO implement here
    }
}
```

MARK ANALYST:

```
import java.util.*;
/**
 *
 */
public class mark analyst extends user {
     /**
     * Default constructor
     */
    public mark analyst() {
    }
     /**
     *
     */
    public void mark analyst() {
```

```
        // TODO implement here
    }
    /**
     *
     */
    public void report generator() {
        // TODO implement here
    }
}
```

REPORT:

```
import java.util.*;
/**
 *
 */
public class report extends mark analyst {
      /**
     * Default constructor
     */
    public report() {
    }
     /**
     *
     */
    public void report name;
```

```
    /**
     *
     */
    public void report type;
}
```

STUDENT:

```
import java.util.*;
/**
 *
 */
public class student extends user {
     /**
     * Default constructor
     */
    public student() {
    }
     /**
     *
     */
    public void view report() {
        // TODO implement here
    }
}
```

USER:

```
import java.util.*;
/**
 *
 */
public class user1190 {
    /**
    * Default constructor
    */
   public user() {
   }
    /**
    *
    */
   public void ID;
    /**
    *
    */
   public void name;
   /**
    *
    */
   public void login() {
       // TODO implement here
```

```
    }
    /**
     *
     */
    public void view report() {
        // TODO implement here
    }
}
```

Result:

Thus, the UML diagrams for Student Mark Analysis system in StarUML have been designed and java code template was generated for coding successfully.

EXP.NO:04 **DATE:07-02-24**

PAYROLL SYSTEM

AIM:

To develop the Payroll System using StarUML.

INTRODUCTION:

This document deals with online Payroll Processing System. This document is designed in such a way that the reader understands it. The use case description and other documents are described in such a way that the system reaches the people easily.

OBJECTIVES:

To develop a user-friendly online pay roll processing system which makes the work easy and faster.

SCOPE:

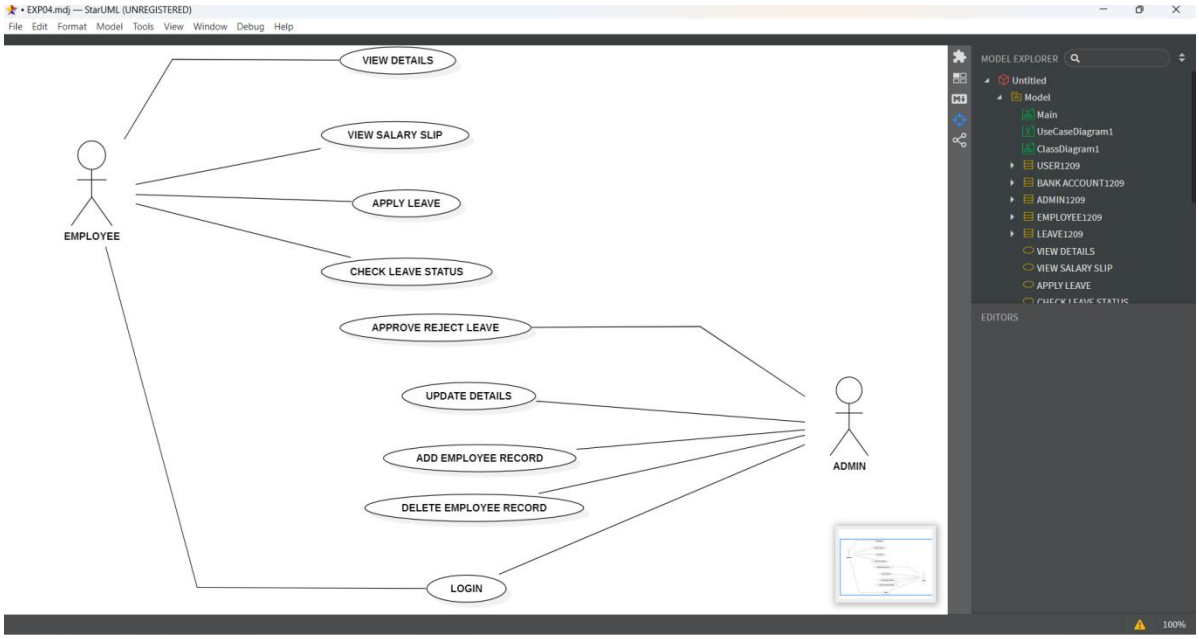
This project describes about the e payroll system, there are two types of Users, normal users and Administrator. Users can log on the directory by a unique User ID that is provided to them and for every ID check there is a password. Administrator does authorization, he is the manager of that directory, and he can Search employee details with names and their employee id. This would help users to Create or find employee details easily.

PROBLEM STATEMENT:

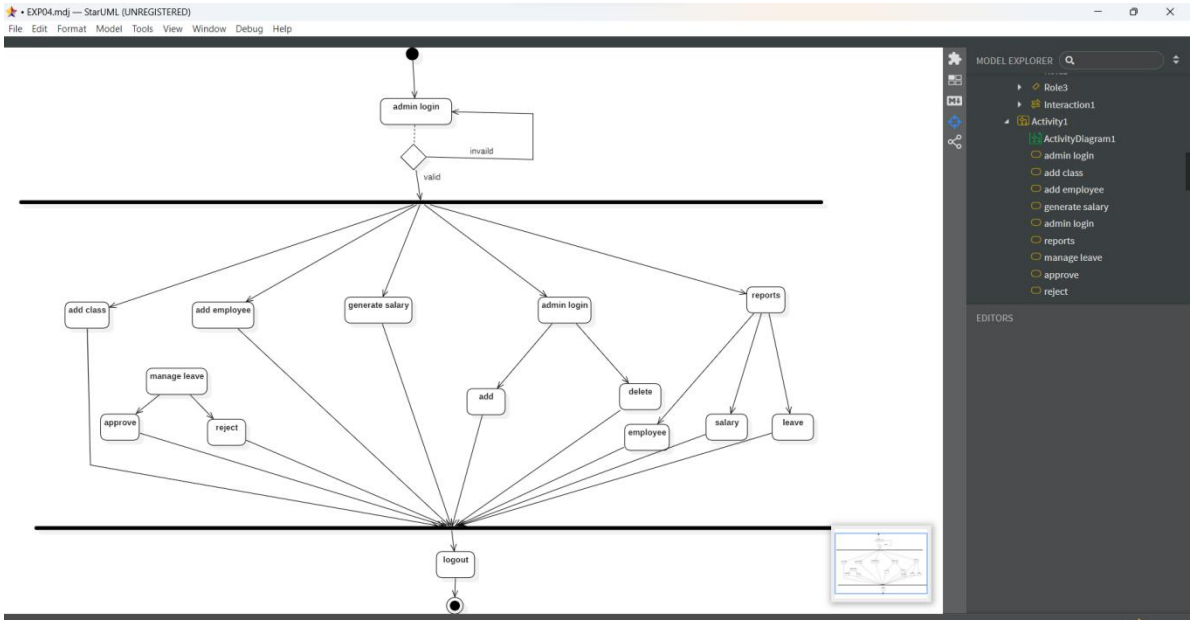
Designing a payroll system for users and administrator through Authentication Process, where only administrator can alter the data in Oracle database, Checking ID’s and password of authorized users.

STAR UML DIAGRAMS:

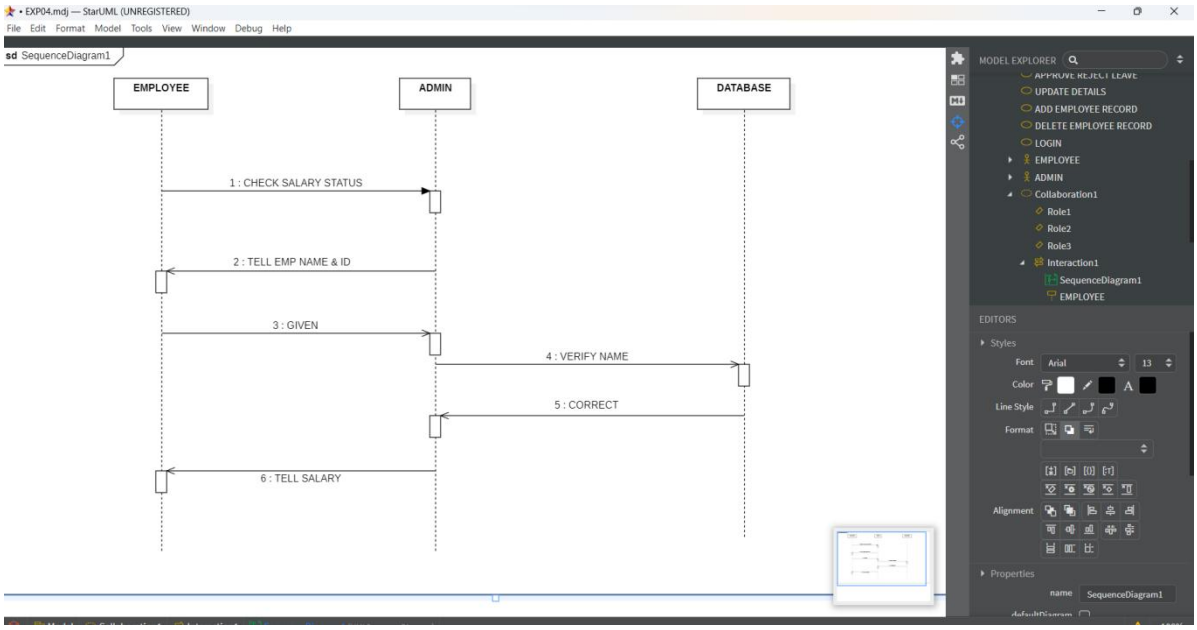
USE CASE DIAGRAM:



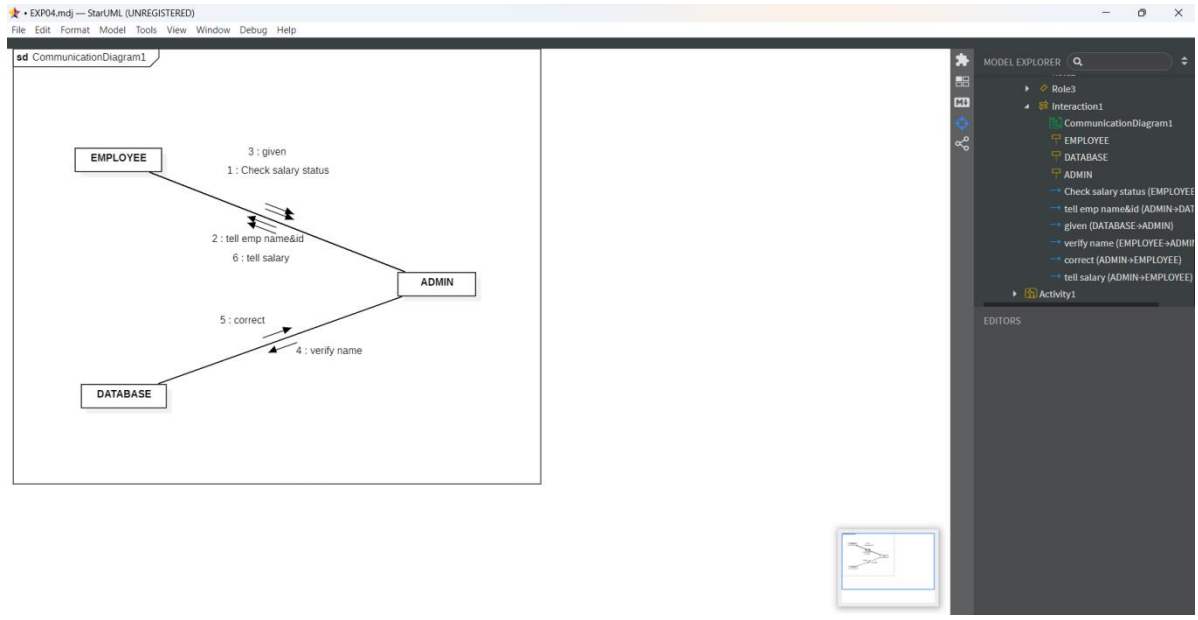
ACTIVITY DIAGRAM:



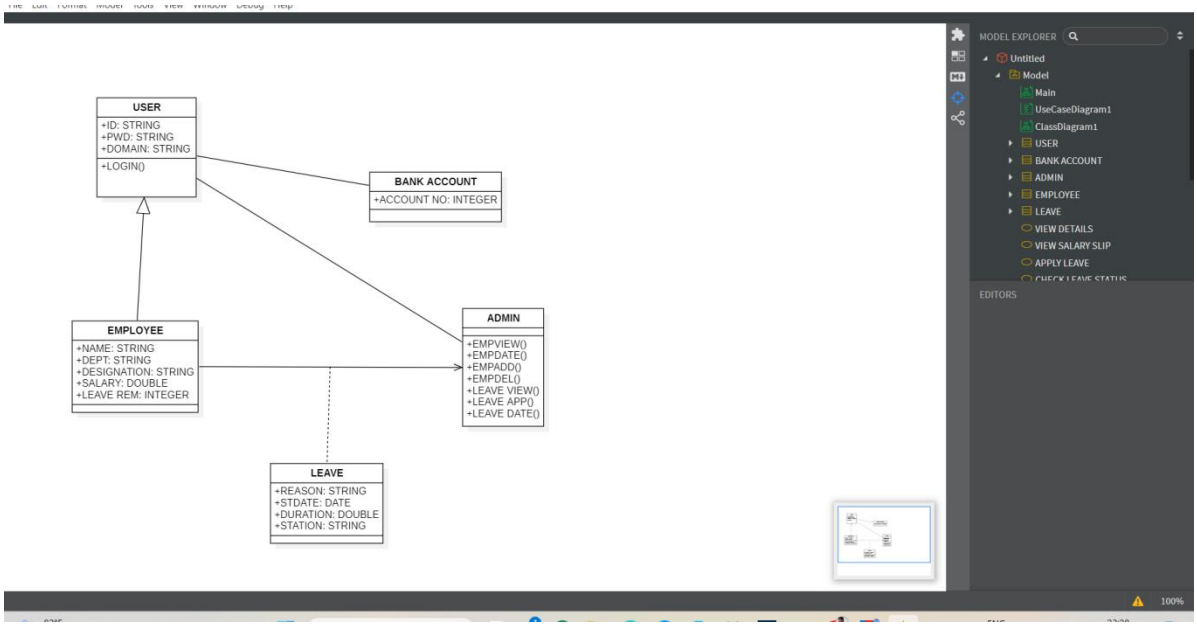
SEQUENCE DIAGRAM:



COMMUNICATION DIAGRAM:



CLASS DIAGRAM:



ALGORITHM:

1. Draw the class diagram and then save it.

2. Open the saved file with star uml.

3. Then go to tools at right corner.

4. Select the code language which we can prefer.

5. Then we have the code generator in tools so give access to it.

6. Then go to tools and select the language then it will take us to another window.

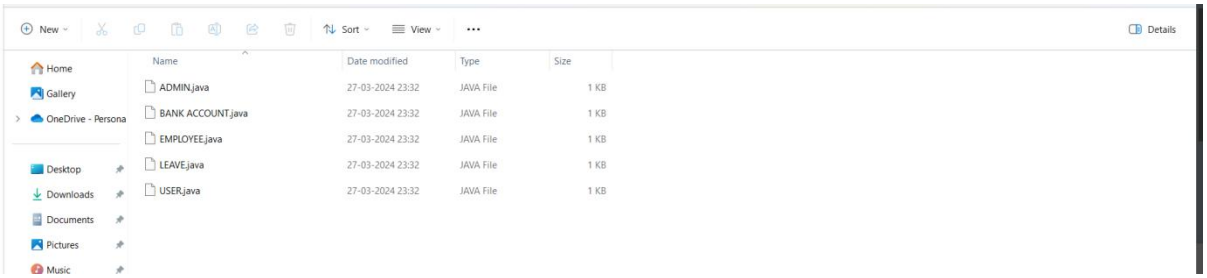
7. There select the class attribute to generate the code.

8. Then it will ask for the directory to store the code.

9. Select the directory and click on generate code.

10. Then code generated successfully.

OUTPUT:



ADMIN:

```
import java.util.*;

public class admin {

    public admin() {

    }

    public void empview() {

        // TODO implement here

    }

   public void empdate() {

        // TODO implement here

    }

  public void empadd() {

        // TODO implement here

   public void empdel() {

        // TODO implement here

   }

    public void leave view() {

        // TODO implement here
```

```
}
    public void leave app() {
        // TODO implement here
}
    public void leave date() {
        // TODO implement here
    }
}
```

BANK ACCOUNT:

```
import java.util.*;
public class bank account {
    public bank account() {
    }
   public integer account no;
}
```

EMPLOYEE:

```
import java.util.*;
public class employee extends user {
     public employee() {
     }
public string name;
public string dept;
public string designation;
public double salary;
```

```
public integer leave rem;
}
```

LEAVE:

```
import java.util.*;
public class leave {
    public leave() {
    }
public string reason;
public date stdate;
public double duration;
public string station;
}
```

USER:

```
import java.util.*;
public class user {
    public user() {
      }
    public void ID string;
    public void pwd string;
    public void domain string;
    public void login() {
        // TODO implement here
    }
}
```

RESULT:

Thus, the UML diagrams for Payroll system in Star UML has been designed and java code the template was generated for coding successfully.

EXP.NO:05 **DATE:14-02-24**

INVENTORY SYSTEM

AIM: To develop the Inventory system using Star UML.

INTRODUCTION:

The stock maintenance system is basically for the customers who access the information about the stock (here it is books in the book store) and retrieves the information.

OBJECTIVES:

The purpose of the document is to define the requirements of the stock maintenance system. This supplementary specification lists the requirements that are not readily captured in the use cases of the use case model. The supplementary specification & the use case model together capture a complete set of requirement on the system.

SCOPE:

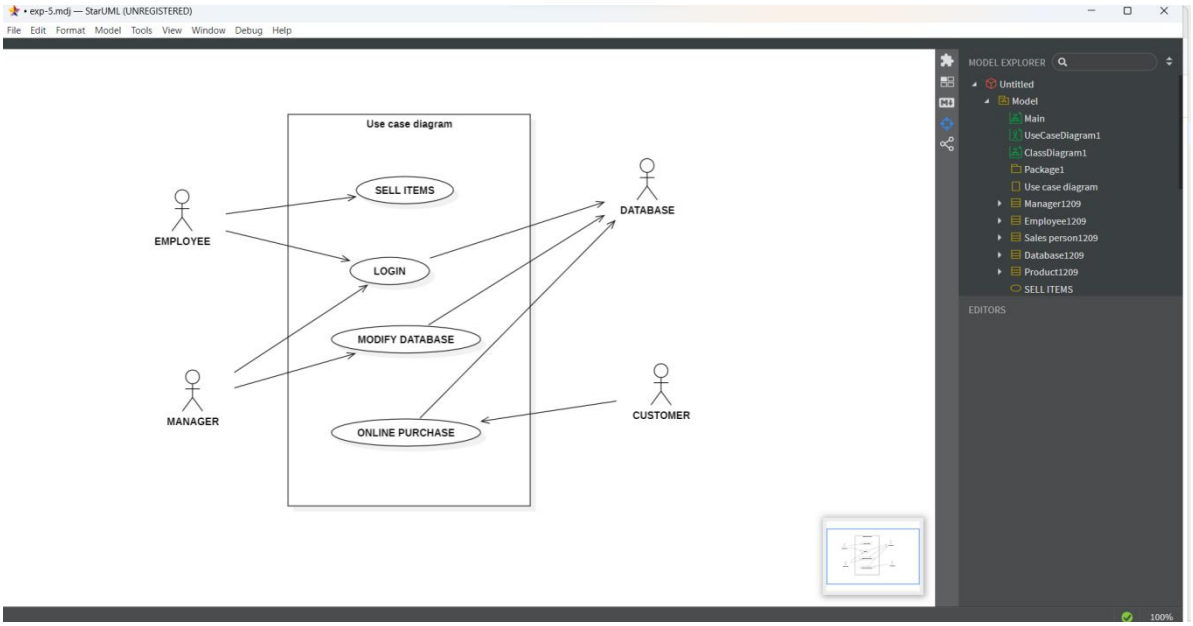
This supplementary specification applies to the stock maintenance system. This specification defines the non-functional requirements of the system, such as reliability, usability, performance and supportability as well as functional requirements that are common across a number of use cases.

PROBLEM STATEMENT:

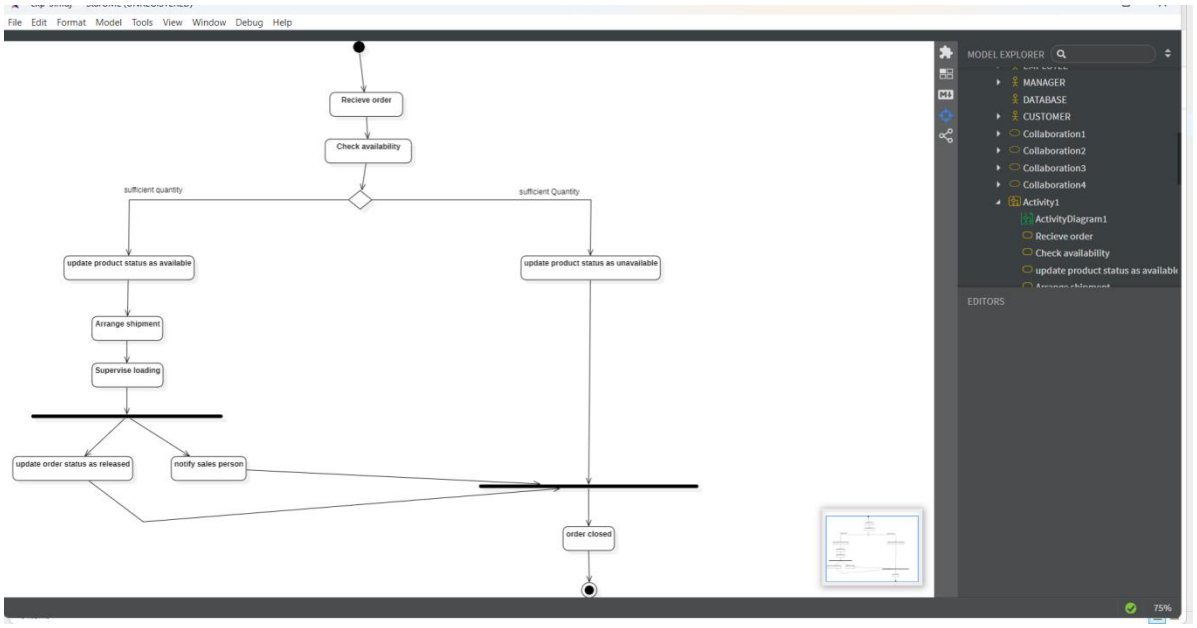
A new stock maintenance system for a book store is to replace the existing maintenance system which is inefficient. The new stock maintenance system will allow the employee to record information of the books available in the book store and generate report based on the total amount of sales. The new system will have a Windows based desktop interface to allow employees to enter the information of sale, purchase orders, change employee preferences and create reports. Employees can only access the information and purchase orders for security purpose. The system retains information on all the books in the shop. The system retains the records of the cost, edition, author, publication of the books. The employee maintains the information of the sale of books. He can add the books at right time and update the database. The customer can view the availability of the required books and the price of the books. The customer can just view them but cannot make any changes.

STAR UML DIAGRAMS:

USE CASE DIAGRAM:

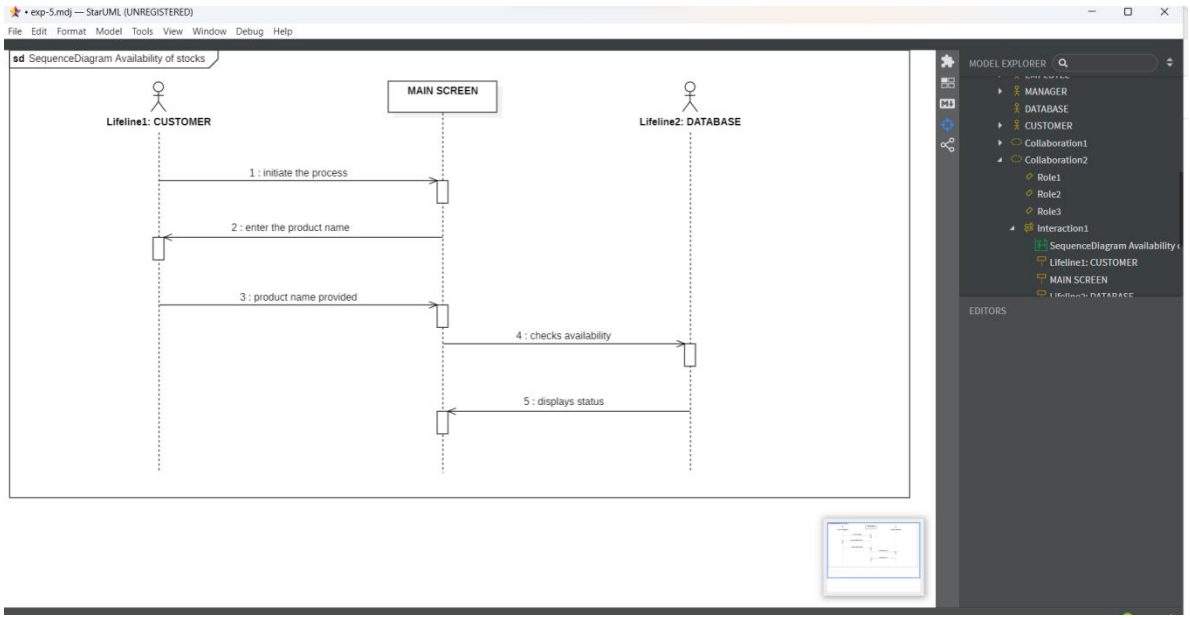


ACTIVITY DIAGRAM:

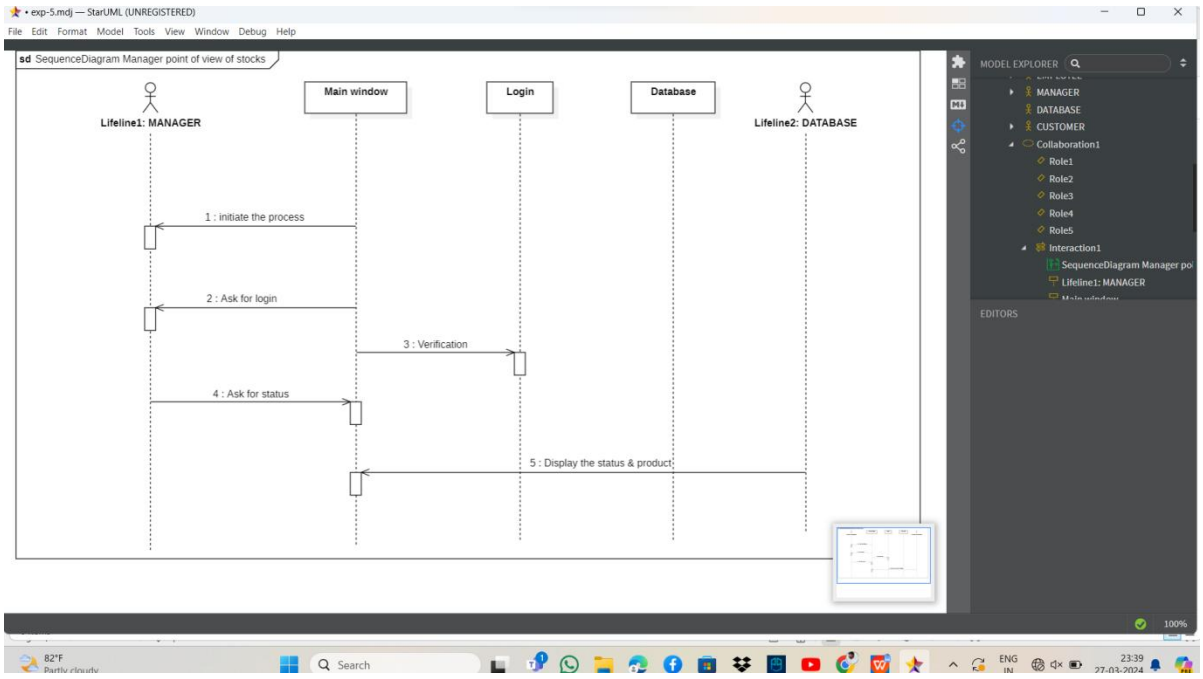


SEQUENCE DIAGRAM:

AVAILABILITY OF STOCKS

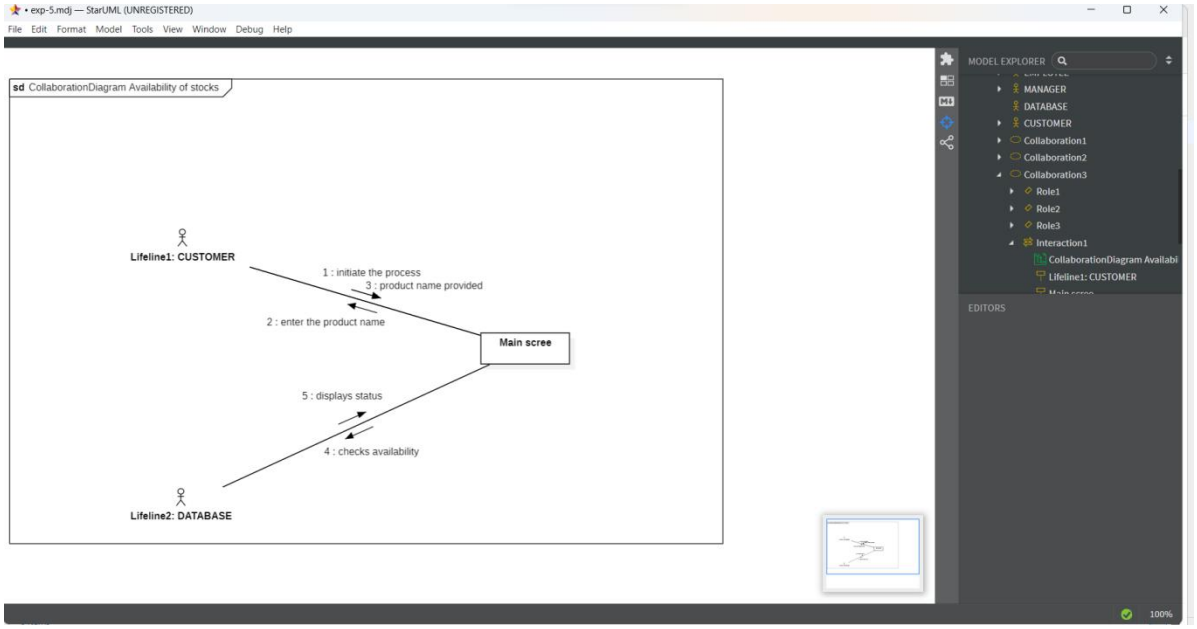


MANAGER POINT OF VIEW

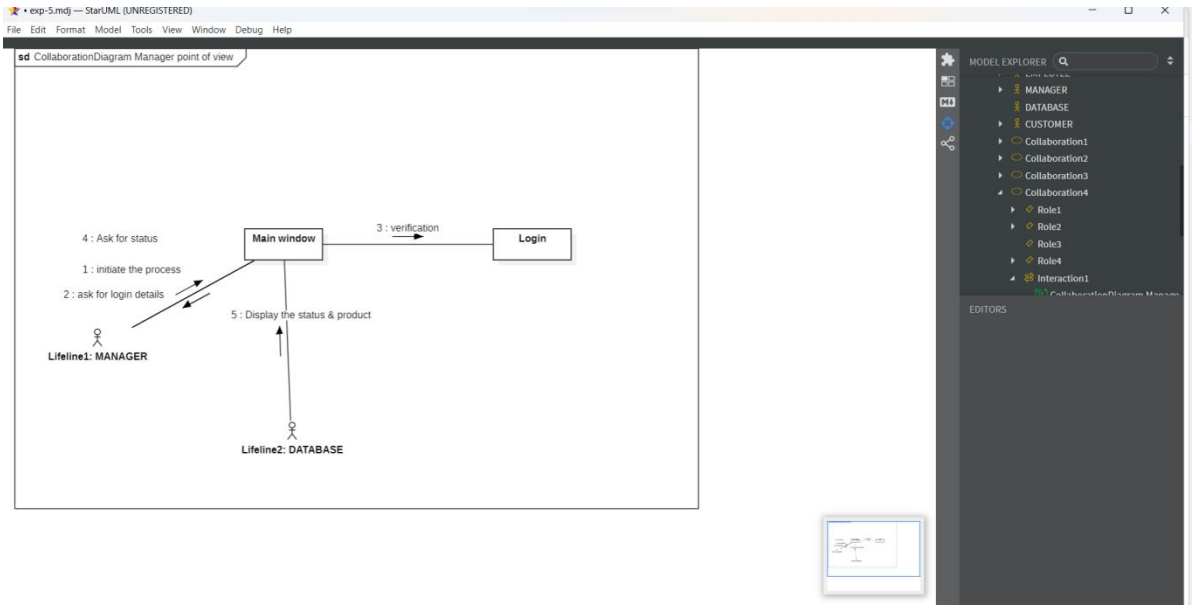


COMMUNICATION DIAGRAM:

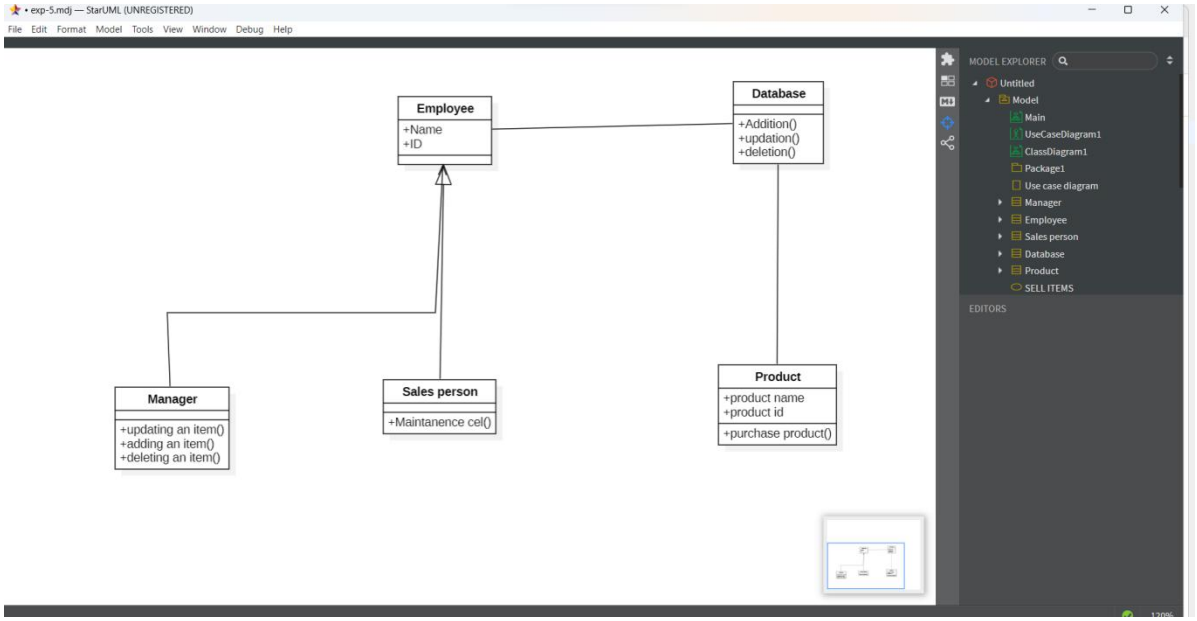
AVAILABILITY OF STOCKS



MANAGER POINT OF VIEW



CLASS DIAGRAM:



ALGORITHM:

1. Draw the class diagram and then save it.

2. Open the saved file with star uml.

3. Then go to tools at right corner.

4. Select the code language which we can prefer.

5. Then we have the code generator in tools so give access to it.

6. Then go to tools and select the language then it will take us to another window.

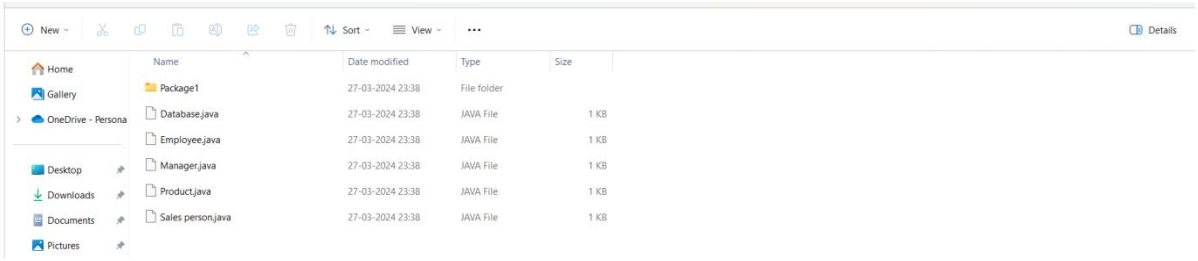
7. There select the class attribute to generate the code.

8. Then it will ask for the directory to store the code.

9. Select the directory and click on generate code.

10. Then code generated successfully.

OUTPUT:



DATA BASE:

```
import java.util.*;

public class Database  {

 public Database () {

    }

    public void addition() {

        // TODO implement here

    }

    public void updation() {

        // TODO implement here

    }

    public void deletion() {

        // TODO implement here

    }

}
```

EMPLOYEE:

```
import java.util.*;
public class Employee  {
public Employee () {
   }
public void Name;
public void ID;
public void Operation1() {
      // TODO implement here
 } }
```

MANAGER:

```
import java.util.*;
public class Manager  {
public Manager () {
   }
public void Attribute1;
public void updating an item() {
      // TODO implement here
   }
public void adding an item() {
      // TODO implement here
   }
public void deleting an item()
 // TODO implement here   } }
```

PRODUCT:

```
import java.util.*;
public class Product  {
public Product () {
   }
public void Product name;
public void purchase product() {
      // TODO implement here
   }
}
```

SALES PERSON:

```
import java.util.*;
public class Sales person  extends Employee  {
public Sales person () {
   }
public void Maintainence cell() {
      // TODO implement here
   }
}
```

RESULT:

Thus, the UML diagrams for Inventory system in Star UML has been designed and java code template was generated for coding successfully.

EXP.NO:06 **DATE:21-02-24**

AUTOMATING THE BANKING SYSTEM

AIM :

To develop the Automating banking System using StarUML.

PROBLEM ANALYSIS AND PROJECT PLANNING

INTRODUCTION:

Banking is one of the common day to day activities of life. Nowadays it is totally different from what existed a few years back, today banking has become completely computerized, new facilities such as credit cards, debit cards & ATM has been introduced. ATM stands for Automatic Teller Machine which is basically used to withdraw money from an account.

OBJECTIVES:

The objective of this software is similar to the ATM software installed in an ATM center. It first validates the pin of the ATM card, then the type of transaction is enquired and the information from the customer is validated. If it is a withdrawal the amount is asked. After the money is delivered the transaction details are updated in the database where the customer's information is stored.

SCOPE:

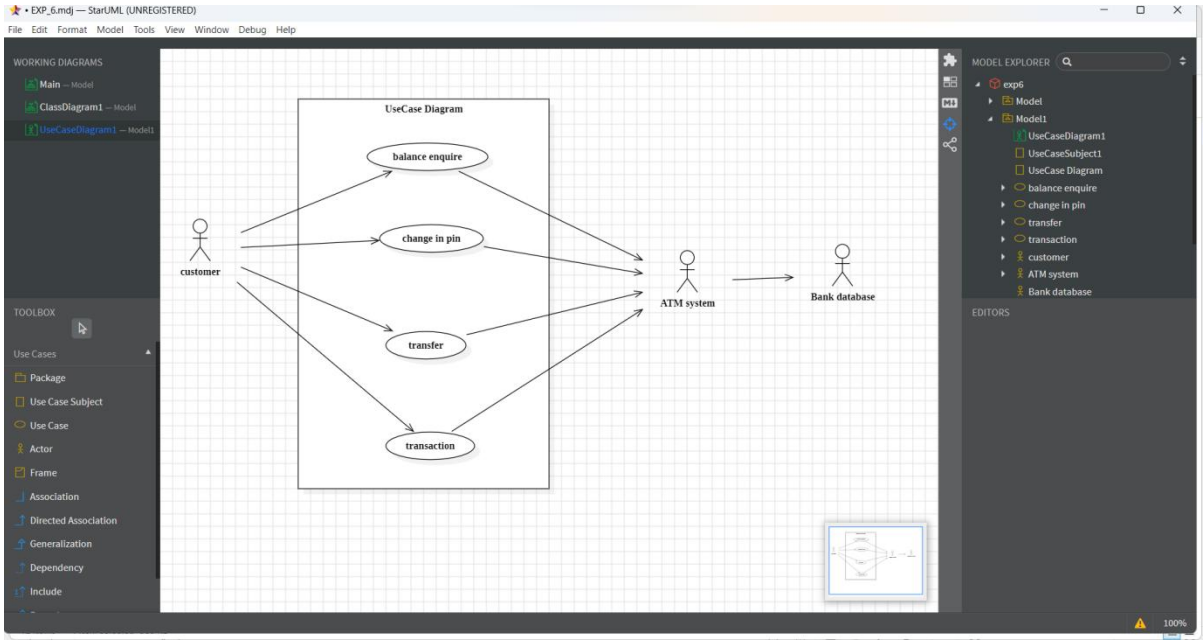
The scope of the project is to design an ATM system that will help in completely automatic banking, this software is going to be designed for withdrawal and deposit and also register the transaction in the database where the customer's information is stored.

PROBLEM STATEMENT:

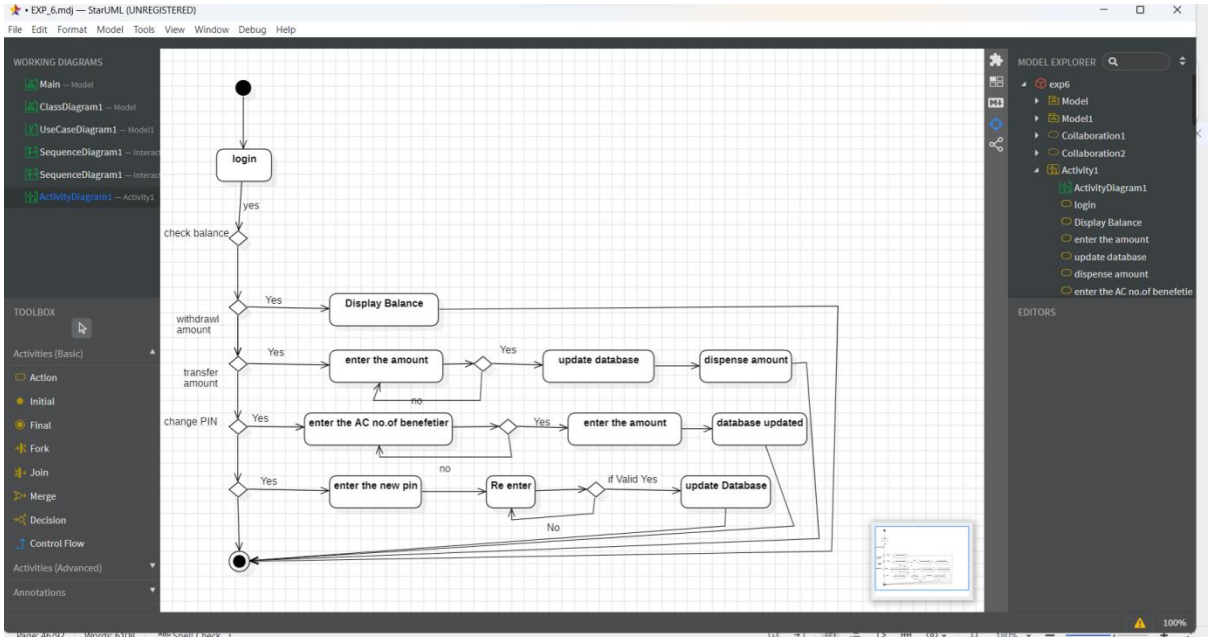
ATM banking is another type of banking where the most frequent type of transaction made is withdrawal. A user may withdraw as much as he wants until his account holds a sum greater than his withdrawal amount. ATM is completely automated and there is no necessity of the ATM center being placed at the bank itself. It can be placed in the shopping malls, airports, railway stations etc. This ATM system can use any kind of interface. But it should be user friendly and not confusing. Help manuals should be provided in case any customer has problem working with the software.

STAR UML DIAGRAMS :

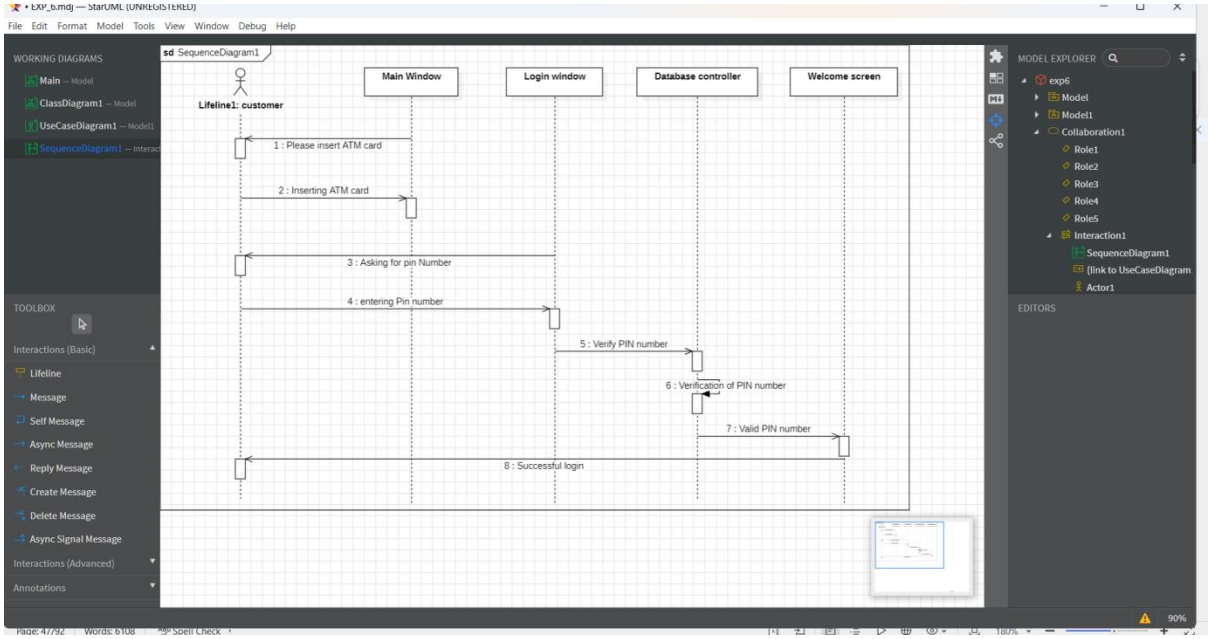
USE CASE DIAGRAM



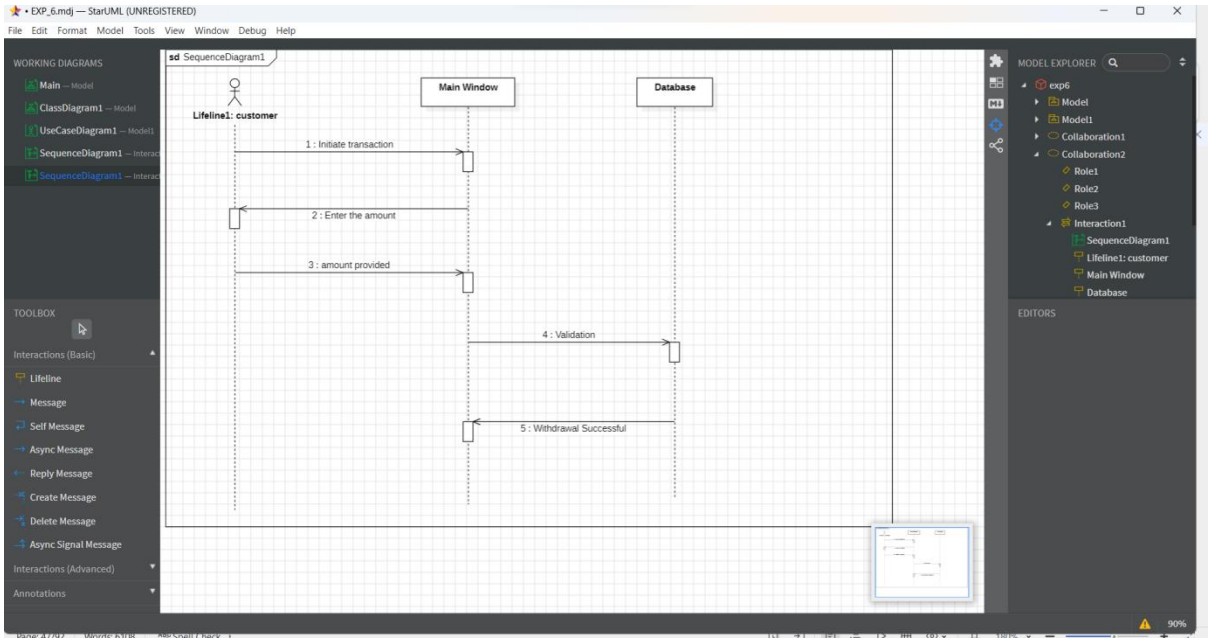
ACTIVITY DIAGRAM



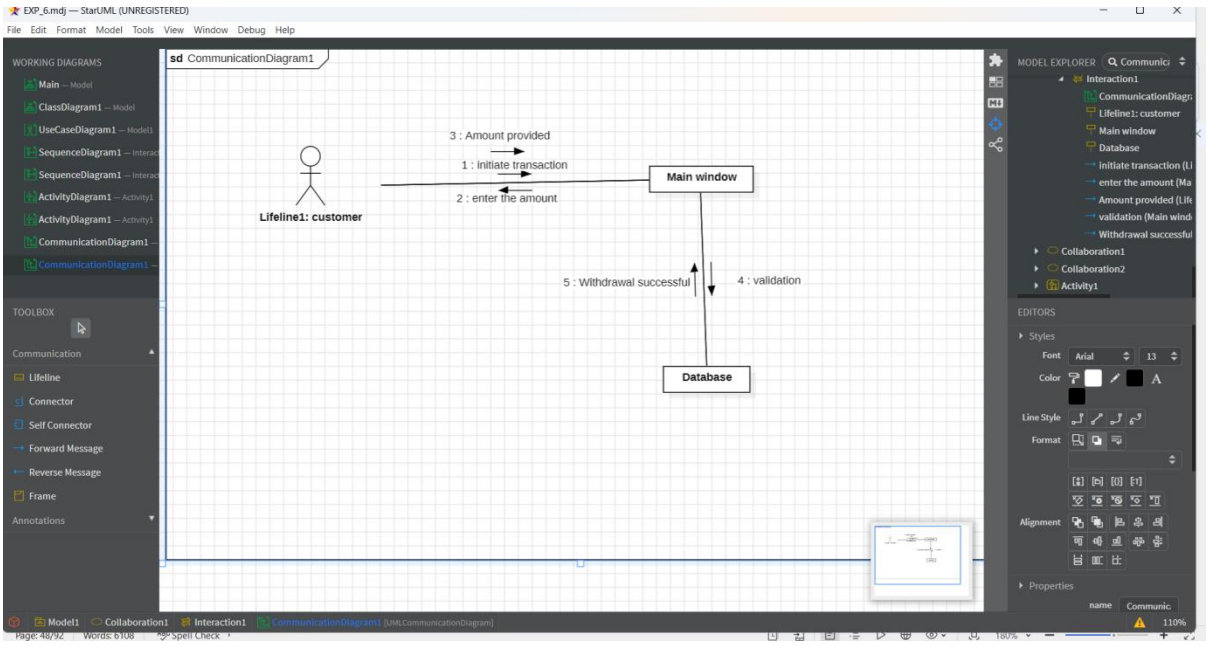
SEQUENCE DIAGRAM



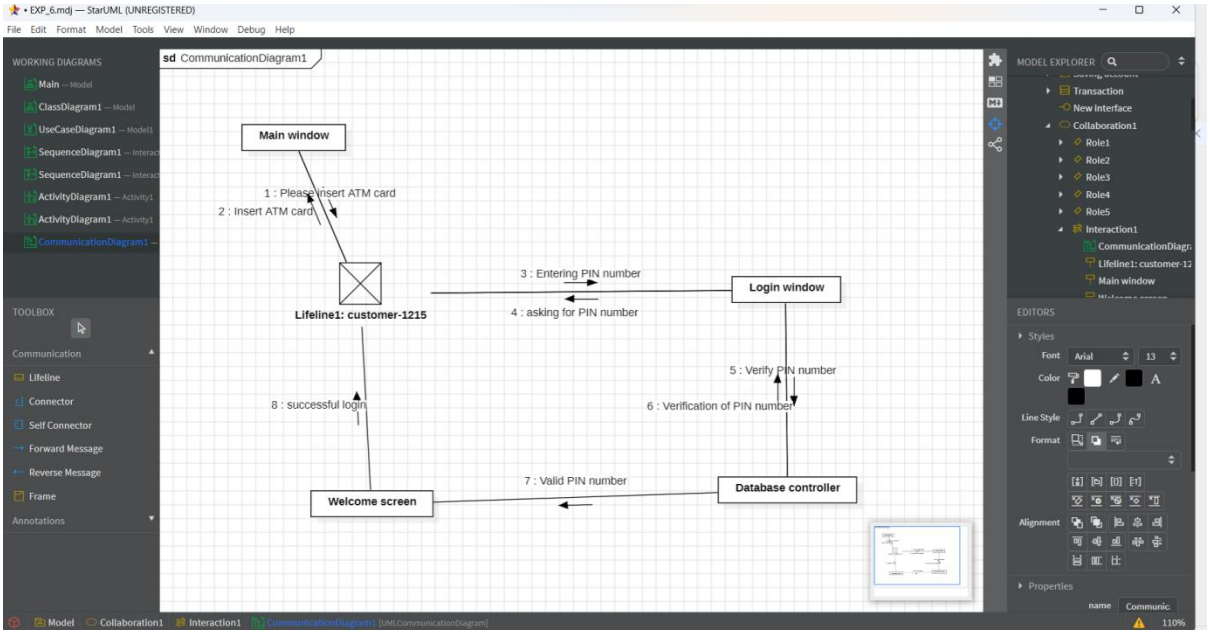
SEQUENCE DIAGRAM_02



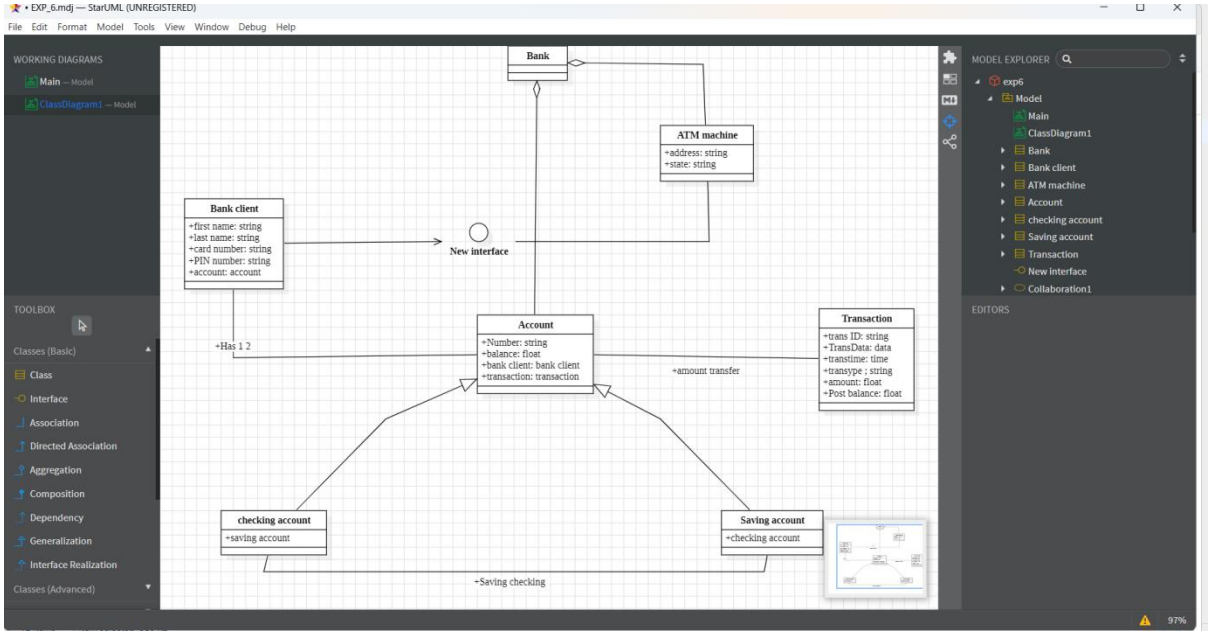
COMMUNICATION DIAGRAM



COMMUNICATION DIAGRAM_02



CLASS DIAGRAM



STEPS TO GENERATE CODE IN STAR UML

1. Draw the class diagram and then save it.

2. Open the saved file with star uml.

3. Then go to tools at right corner.

4. Select the code language which we can prefer.

5. Then we have the code generator in tools so give access to it.

6. Then go to tools and select the language then it will take us to another window.

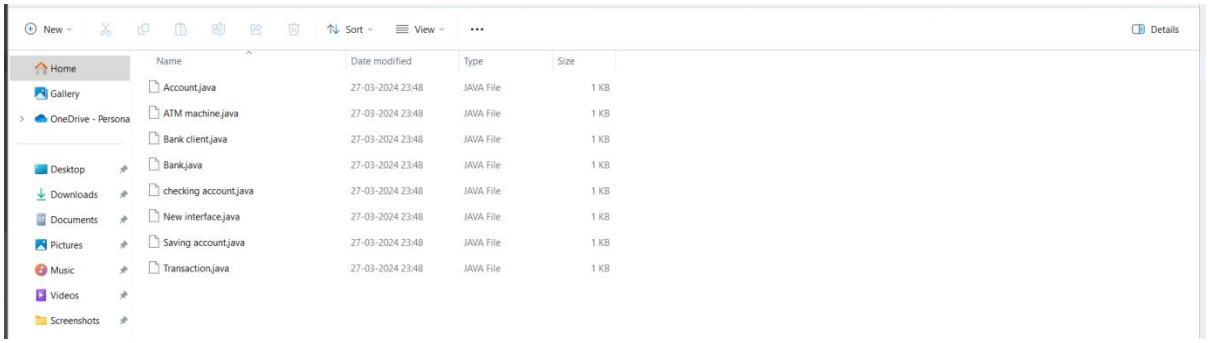
7. There select the class attribute to generate the code.

8. Then it will ask for the directory to store the code.

9. Select the directory and click on generate code.

10. Then code generated successfully

OUTPUT :



ACCOUNTS:

```
import java.util.*;

/**

 *

 */
public class Account  {
    /**
     * Default constructor
     */
    public Account () {
    }
     /**
      *
      */
    public string Number;


    /**
      *
      */
    public float balance;


    /**
      *
```

```
    */
    public bank client bank client;

    /**
     *
     */
    public transaction transaction;

}
```

ATM MACHINE:

```
import java.util.*;

/**
*
*/
public class ATM machine {

/**
                Default constructor
*/
public ATM machine() {
}

/**
*
*/
public string address;

/**
*
*/
```

```
public string state;

}
```

BANK CLIENT:

```
import java.util.*;

/**
 *
 */
public class Bank client  {

    /**
     * Default constructor
     */
    public Bank client () {
    }

    /**
     *
     */
    public string first name;

    /**
     *
     */
    public string last name;

    /**
     *
     */
    public string card number;
```

```
    /**
     *
     */
    public string PIN number;

    /**
     *
     */
    public account account;

}
```

BANK:

```
import java.util.*;

/**
 *
 */
public class Bank  {

    /**
     * Default constructor
     */
    public Bank () {
    }

}
```

CHECKING ACCOUNT:

```
import java.util.*;
```

```
/**
 *
 */
public class checking account  extends Account  {

    /**
     * Default constructor
     */
    public checking account () {
    }

    /**
     *
     */
    public void saving account;

}
```

NEW INTERFACE:

```
import java.util.*;

/**
 *
 */
public interface New interface {

}
```

SAVING ACCOUNT:

```
import java.util.*;
/**
 *
```

```
 */
public class Saving account  extends Account  {
    /**
    * Default constructor
    */
   public Saving account () {
   }
    /**
    *
    */
   public void checking account;

}
```

TRANSACTIONS:

```
import java.util.*;
/**
 *
 */
public class Transaction {
     /**
    * Default constructor
    */
   public Transaction() {
   }
    /**
    *
    */
 public string trans ID;
    /**
    *
    */
```

```
    public data TransData;
     /**
     *
     */
    public time transtime;
     /**
     *
     */
    public void transype ; string;
    /**
     *
     */
    public float amount;
     /**
     *
     */
    public float Post balance;
}
```

Result:

Thus, the UML diagrams for Banking system in StarUML has been designed and java code template was generated for coding successfully.

Ex. No:7 **Date:28-02-24**

COURSE REGISTRATION SYSTEM

AIM

To develop the Course Registration System using Star UML.

PROBLEM ANALYSIS AND PROJECT PLANNING

INTRODUCTION

This software is designed in such a way that it receives the name and other particulars from the student. Based on marks the student has scored the list of possible branches that will accommodate for the student will be displayed. Only work for the student is what he has to fill in the form and submit it.

OBJECTIVES

The ultimate objective of this software is to eliminate hassles that the student overcomes while registering himself. This software will reduce the paper work. This also reduces the time delay.

SCOPE

The student is first requested to fill the form. This form will contain important particulars of the student like his name, DOB, preferred branch, his marks. Once the student fills it, a unique id number will be generated. An important thing within this is to decide what mode the student opts to make the payment. It may either be by demand draft or credit card. As soon as a student is registered then the number of seats available is displayed.

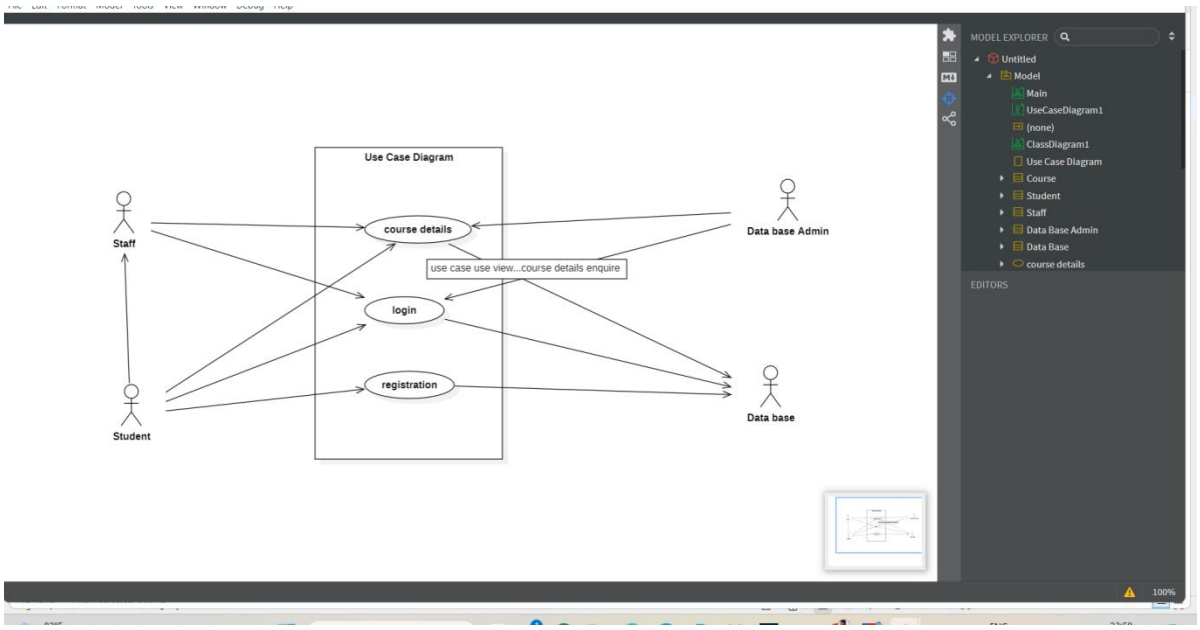
PROBLEM STATEMENT

As project developers we developed a new course registration system to replace the existing manual registration since manual system are prone to errors and take more time. The system made is user friendly and reduces the burden of users. Our system can be made available even in the website of a college. Students can easily register the course in our system without any difficulty and can easily understand and also the time taken for registration is less when compared to manual registration. Options are given to the student to select their electives and

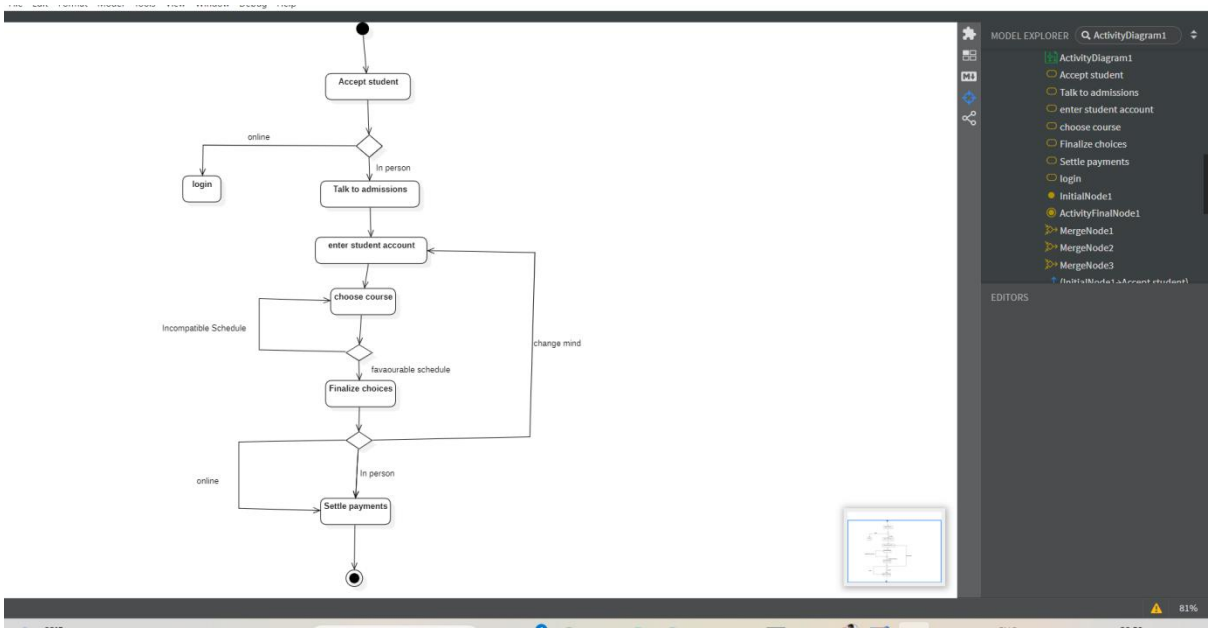
also it shows the number of papers available along with the number of students who have registered and also the number of days for particular elective per semester is also displayed at the side. This makes their work easier as compared to manual registration since they need to make a copy of HOD, staff separately and even if one is missed out the whole process has to be redone. Their information will be stored as soon as they are registered.

STAR UML DIAGRAMS:

USE CASE DIAGRAM

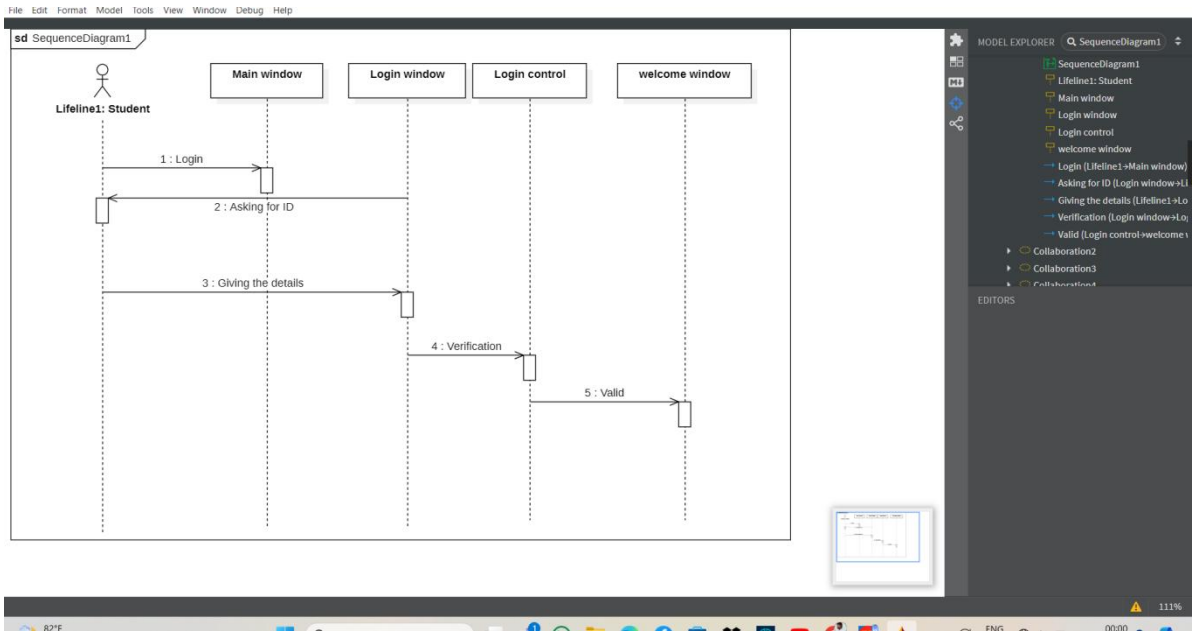


ACTIVITY DIAGRAM

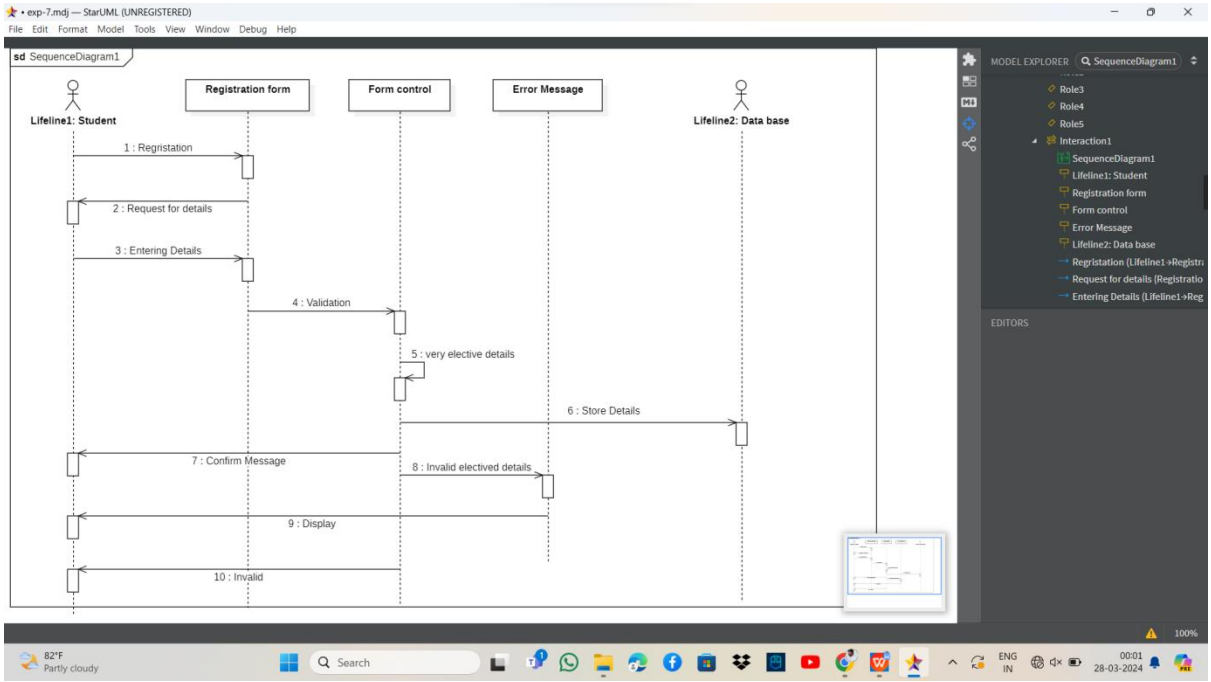


SEQUENCE DIAGRAM

VALID LOGIN

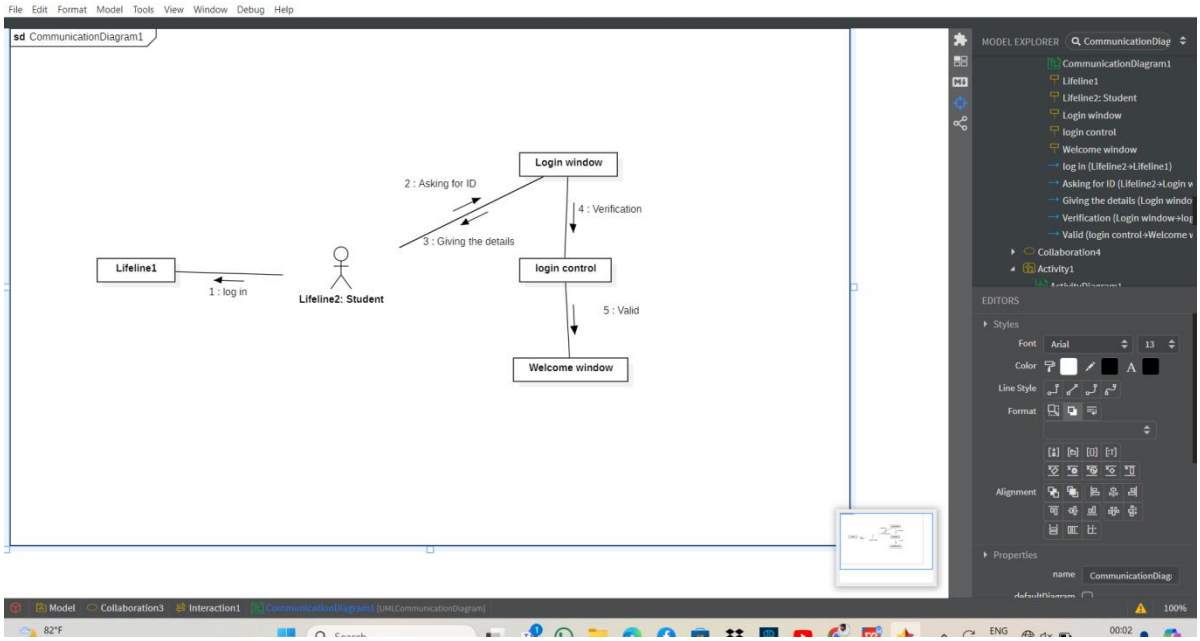


REGISTRATION

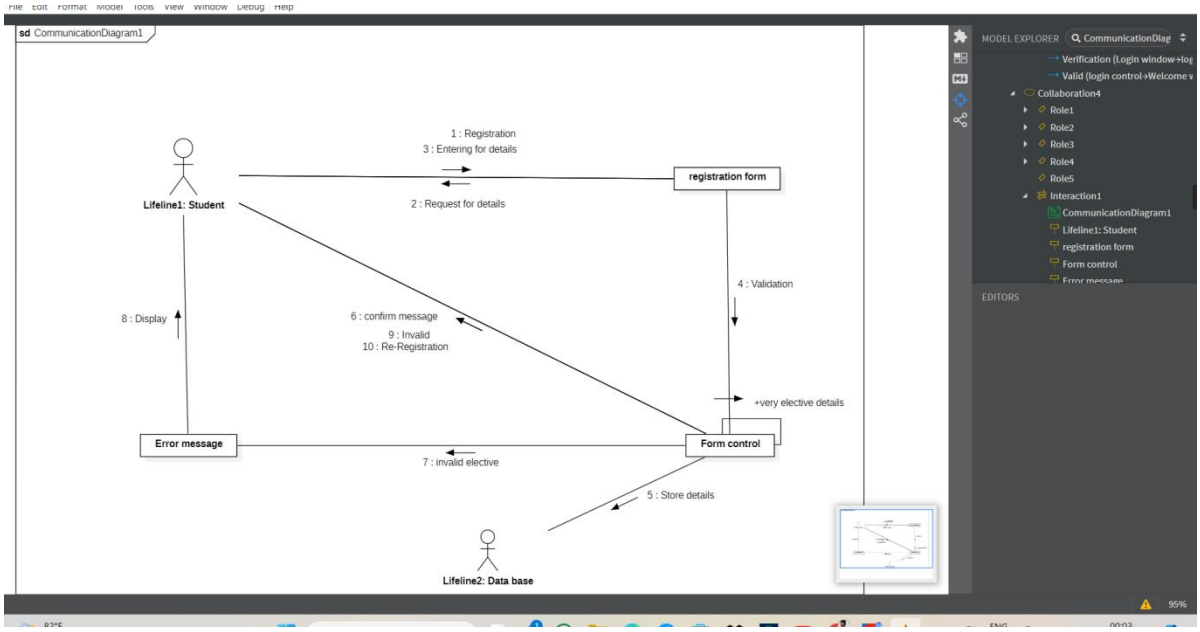


COMMUNICATION DIAGRAM

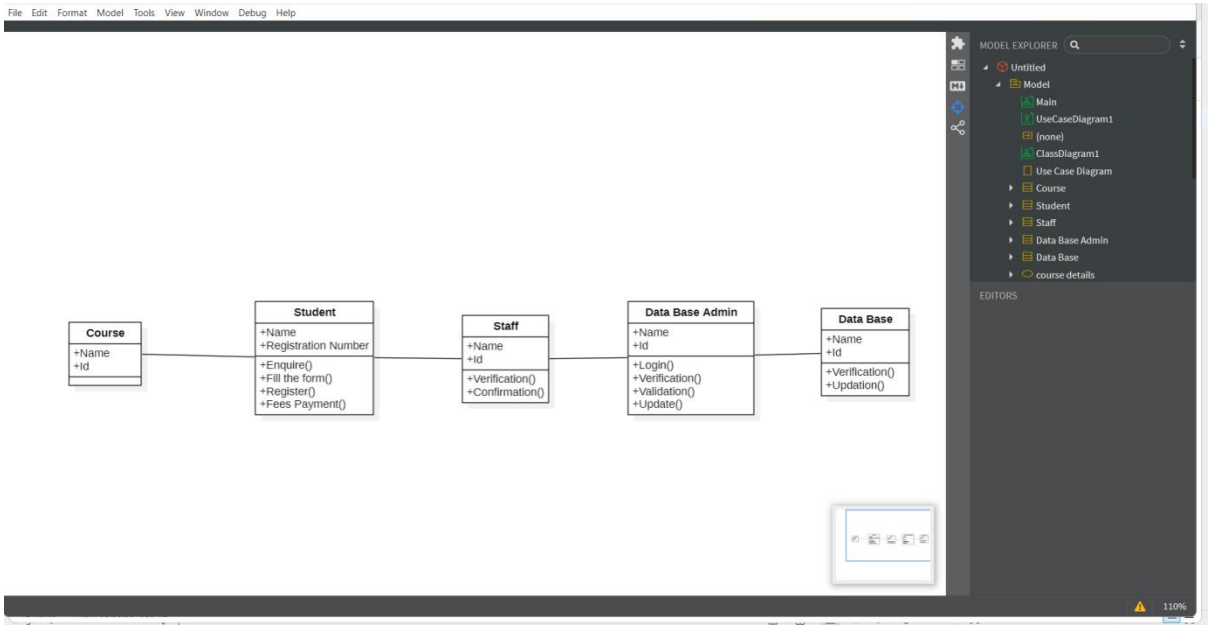
LOGIN



REGISTRATION



CLASS DIAGRAM



STEPS TO GENERATE CODE IN STAR UML

1. Draw the class diagram and then save it.

2. Open the saved file with star uml.

3. Then go to tools at right corner.

4. Select the code language which we can prefer.

5. Then we have the code generator in tools so give access to it.

6. Then go to tools and select the language then it will take us to another window.

7. There select the class attribute to generate the code.

8. Then it will ask for the directory to store the code.

9. Select the directory and click on generate code.

10. Then code generated successfully.

OUTPUT

COURSE

```
import java.util.*;

/**

 *

 */

public class Course {

    /**

     * Default constructor

     */

    public Course() {

    }

    /**

     *

     */

    public void Name;

    /**

     *

     */

    public void id;

}
```

DATA BASE

```
import java.util.*;

/**

 *

 */

public class Data Base {

    /**

     * Default constructor

     */

    public Data Base() {

    }

    /**

     *

     */

    public void Name;

    /**

     *

     */

    public void id;

    /**

     *

     */

    public void Verification() {

        // TODO implement here
```

```
    }
    /**
     *
     */
    public void Updation() {
        // TODO implement here
    }
}
```

DATA BASE ADMIN

```
import java.util.*;
/**
 *
 */
public class Data Base Admin {
    /**
     * Default constructor
     */
    public Data Base Admin() {
    }
    /**
     *
     */
    public void Name;
    /**
```

```
 *
 */
public void id;
/**
 *
 */
public void Login() {
    // TODO implement here
}
 /**
 *
 */
public void Verification() {
    // TODO implement here
}
/**
 *
 */
public void Validation() {
    // TODO implement here
}
/**
 *
 */
```

```
    public void Update() {
        // TODO implement here
    }
}
```

STAFF

```
import java.util.*;
/**
 *
 */
public class Staff {
    /**
     * Default constructor
     */
    public Staff() {
    }
    /**
     *
     */
    public void Name;
    /**
     *
     */
    public void id;
```

```
    /**
     *
     */
    public void Verification() {
        // TODO implement here
    }
    /**
     *
     */
    public void Confirmation() {
        // TODO implement here
    }
}
```

STUDENT

```
import java.util.*;
/**
 *
 */
public class Student {
    /**
     * Default constructor
     */
    public Student() {
    }
```

```
/**
 *
 */
public void Name;
/**
 *
 */
public void Registration Number;
/**
 *
 */
public void Enquire() {
    // TODO implement here
}
/**
 *
 */
public void Fill the Form() {
    // TODO implement here
}
/**
 *
 */
public void Register() {
```

```
        // TODO implement here
    }
    /**
     *
     */
    public void Fees Payment() {
        // TODO implement here
    }
}
```

RESULT

Thus, the UML diagrams for Course registration system in Star UML has been designed and java code template was generated for coding successfully.

EXP.NO:08 **DATE:06-03-24**

LIBRARY MANAGEMENT SYSTEM

AIM:

To develop the Library Management System using Star UML.

INTRODUCTION:

To design a library management system using IBM Rational Rose. Implement this system using Visual. Basic Language with Oracle/SQL.

OBJECTIVE:

Recently the importance of library management system within the service context of libraries has been unchallenged. Students check the list of books available and borrow the books if the book is a borrow book otherwise it is of waste for the student to come to the library to come to check for the books if the student doesn't get the book. Then the librarian checks the student id and allows the member to check out the book and the librarian then updates the member database and also the books database.

SCOPE:

This system would be used by members who may be students or professors of that University to check the availability of the books and borrow the books, and by the librarian to update the databases. The purpose of this document is to analyze and elaborate on the high level needs and features of the Online Library System. It focuses on the capabilities and facilities provided by a Library.

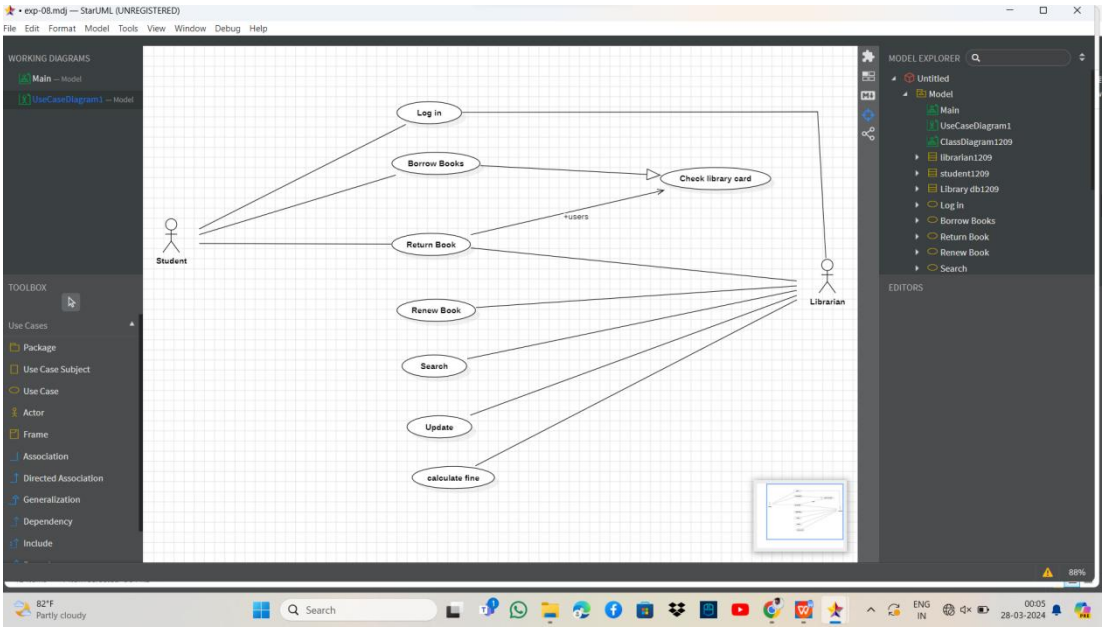
PROBLEM STATEMENT:

The Library Management System is designed & developed for a receipt and issuance of books in the library along with the student's details. A librarian can add, delete and update book status and search from the database. A user can borrow, return books, reserve books and search for books. He can also renew his loan period. If a book is overdue, the user will be fined Rs.5 each day over the due . If a book is reported lost, the user will pay the full cost of the book. The books received in the library are entered in Books Entry form and the new

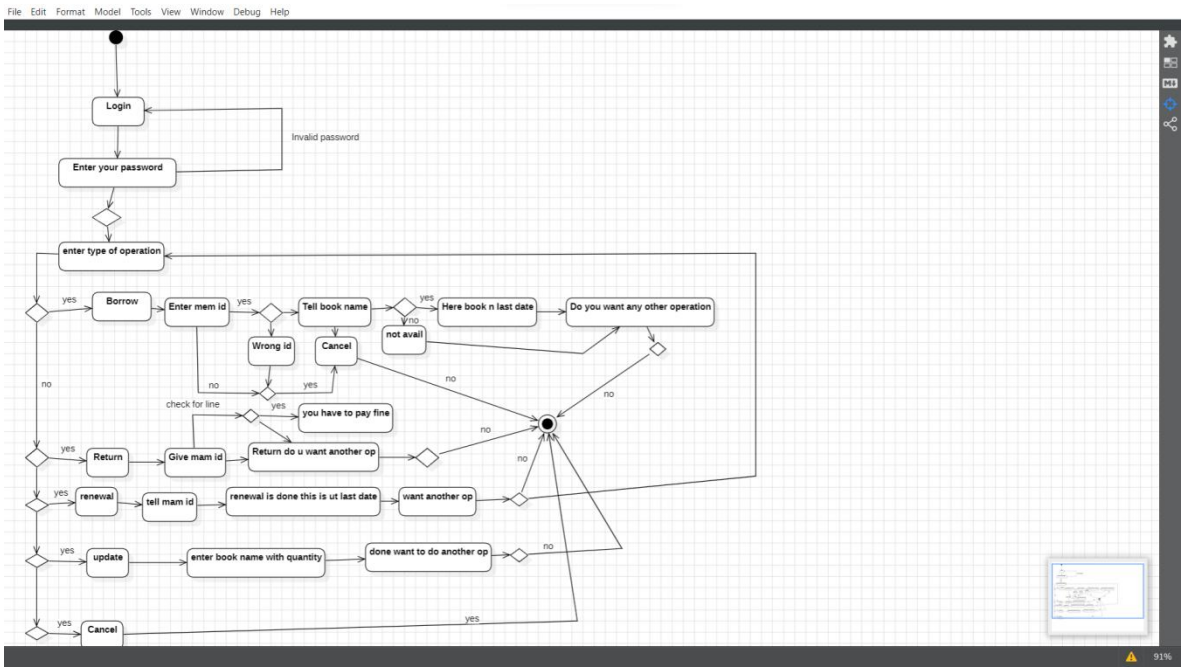
student is entered in the student entry form. When the student wants to get the desired book the same is issued on availability basis to the student.

STAR UML DIAGRAMS:

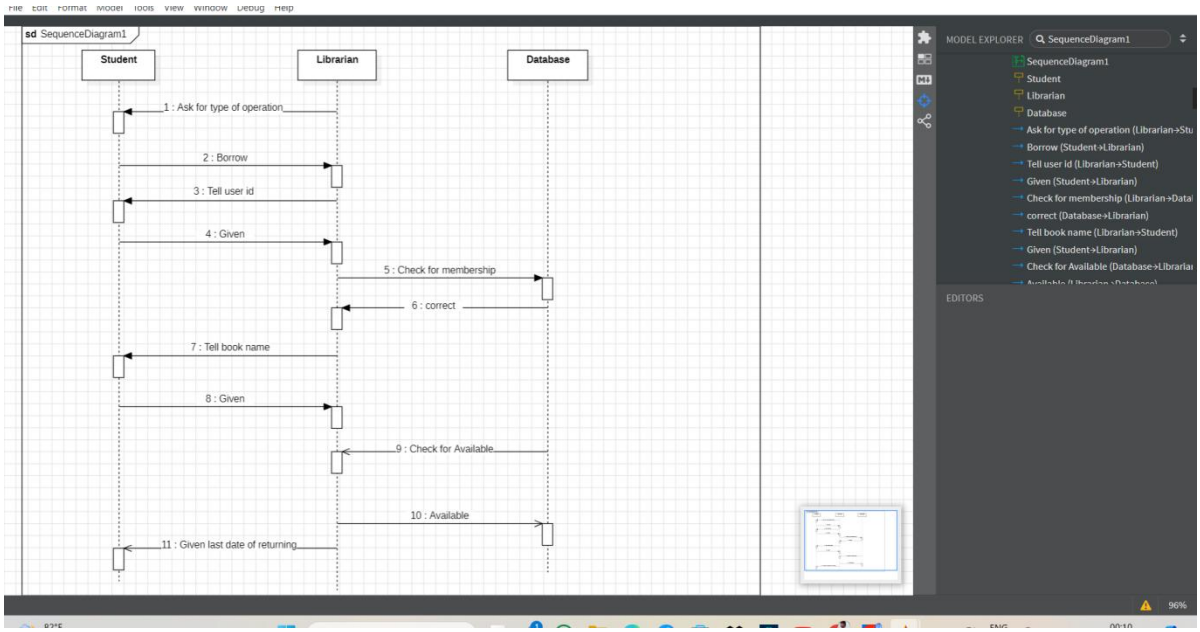
USE CASE DIAGRAM:



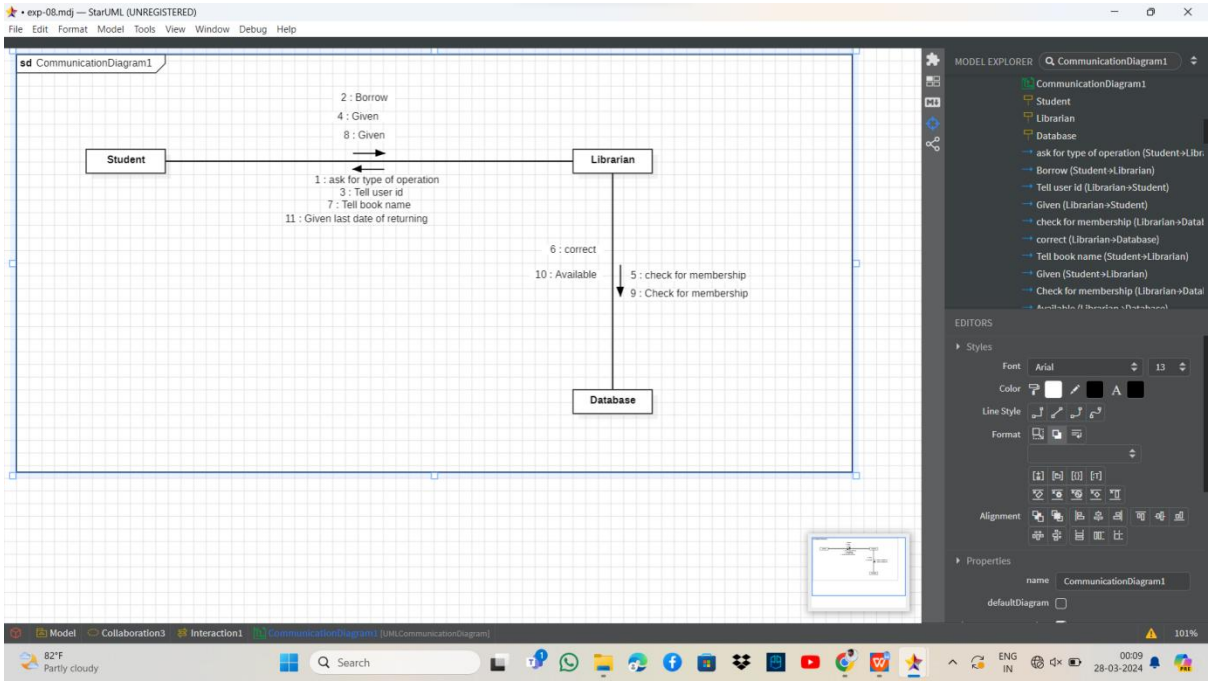
ACTIVITY DIAGRAM:



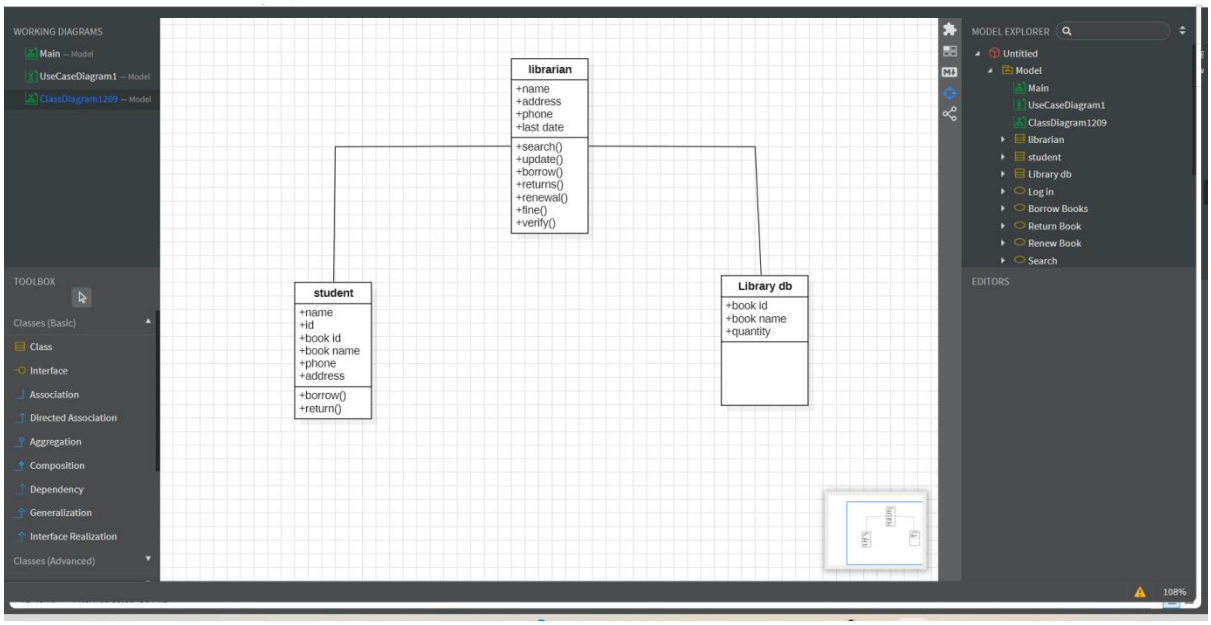
SEQUENCE DIAGRAM:



COMMUNICATION DIAGRAM:



CLASS DIAGRAM:



STEPS TO GENERATE CODE IN STAR UML

1. Draw the class diagram and then save it.

2. Open the saved file with star uml.

3. Then go to tools at right corner.

4. Select the code language which we can prefer.

5. Then we have the code generator in tools so give access to it.

6. Then go to tools and select the language then it will take us to another window.

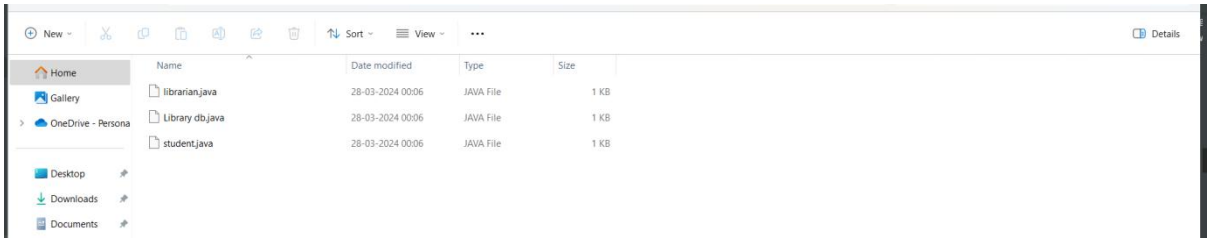
7. There select the class attribute to generate the code.

8. Then it will ask for the directory to store the code.

9. Select the directory and click on generate code.

10. Then code generated successfully

OUTPUT:



LIBRARIAN:

```
import java.util.*;

/****/public class librarian {

 /**

     * Default constructor

     */

    public librarian () {

    }

    /***

     */

    public void name;

    /**

 **/

    public void address;

 /****/

    public void phone;

    /****/

    public void last date;

    /****/

    public void search() {
```

```
    // TODO implement here
    }
    /****/
    public void update() {
    // TODO implement here
    }
    /****/
    public void borrow() {
        // TODO implement here
    }
/****/
public void returns() {
        // TODO implement here
    }
   /****/
    public void renewal() {
        // TODO implement here
    }
    /****/
    public void fine() {
        // TODO implement here
    }
 /****
    public void verify() {
```

```
        // TODO implement here
    }
}
```

LIBRARY DATABASE:

```
import java.util.*;
/****/
public class Library db  {
/**
     * Default constructor
     */
    public Library db () {
    }
/*** */
    public void book id;
    /****/
    public void book name;


    /***/
    public void quantity;
}
```

STUDENT:

```
import java.util.*;
/****/
```

```
public class student  {
/*** Default constructor*/
    public student () {
    }
    /****/
    public void name;
/****/
    public void id;
 /****/
    public void book id;
/****/
    public void book name;

    /****/
    public void phone;

    /****/
    public void address;
  /****/
    public void borrow() {
        // TODO implement here
    }
    /****/
```

```
    public void return() {
        // TODO implement here
    }

}
```

Result:

Thus, the UML diagrams for Library Management system in Star UML has been designed and java code template was generated for coding successfully.

EXP.NO:9 **DATE:13-03-24**

PASSPORT AUTOMATION SYSTEM

AIM:

To develop the Passport Automation System using StarUML.

OBJECTIVES:

To develop the passport automation system software using UML language. It is the interface between applicant and authority responsible for issue the passport. It aims at improving efficiency and reducing complexities.

SCOPE:

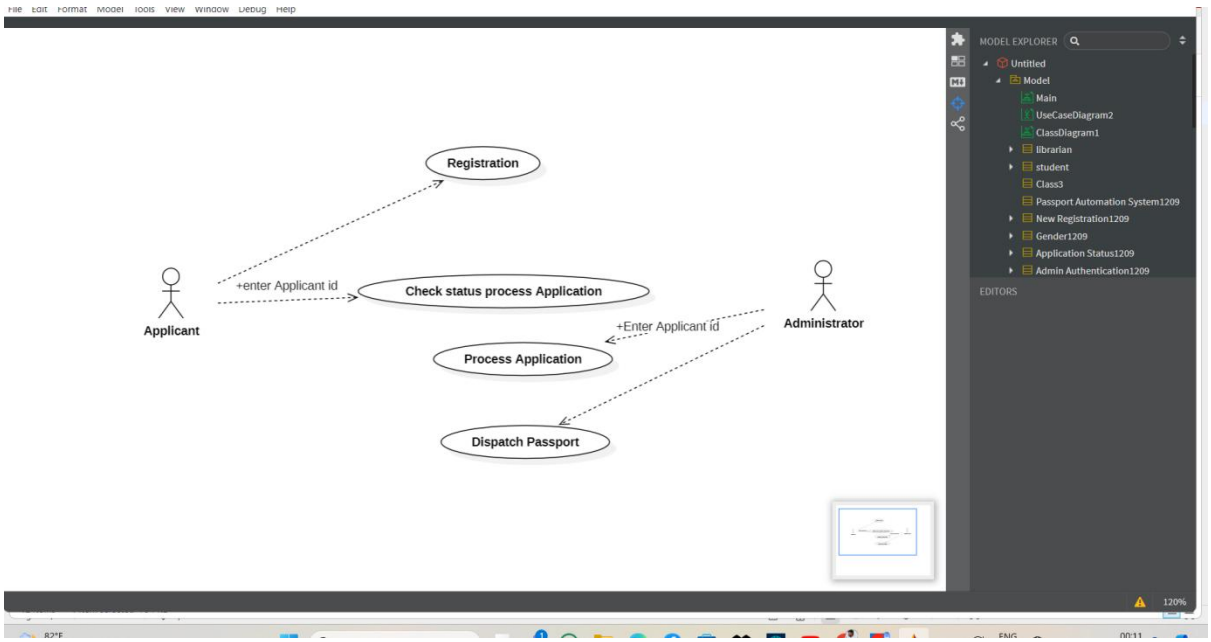
The system provides the online interface to the user where we can fill their form and personal detail with necessary proof. The authority concerned with the issue of passport can use this system to reduce is workload and process it speedy manner. It provides communication platform between administrator and applicant. To transfer the data between passport authority and local police verifying the applicant's information.

PROBLEM STATEMENT:

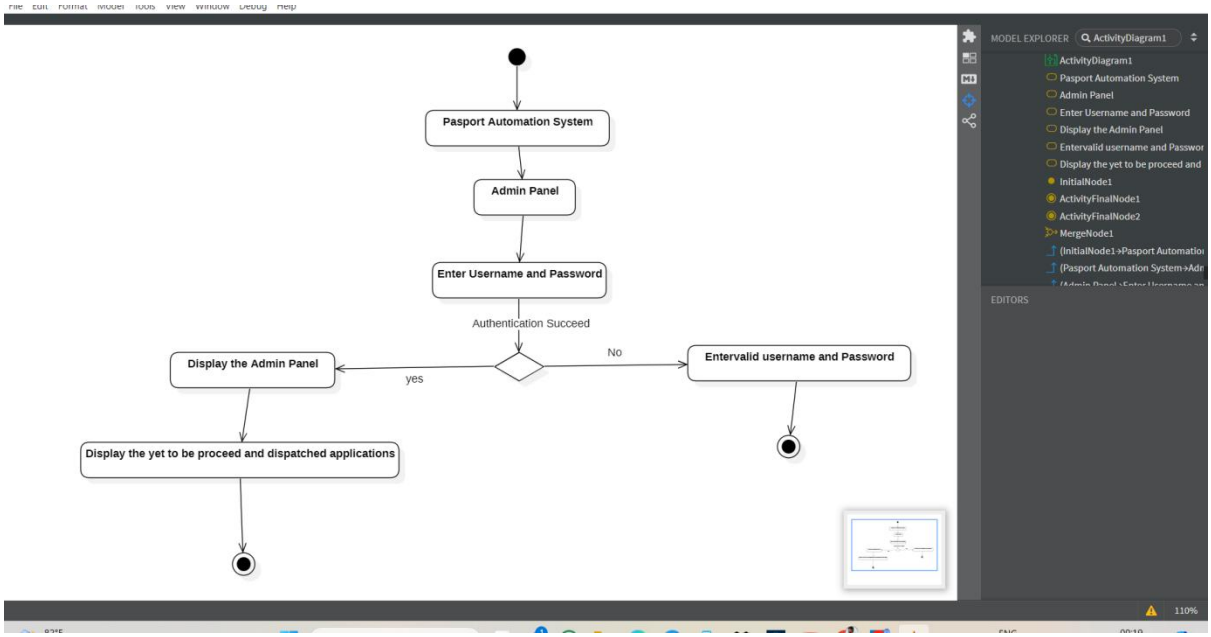
My project title is passport automation system. In this project we can develop the modules such as login, fill the form with necessary proof, verifying the applicant's information, validity checking and issue the passport for that particular applicant. In this login module, we can perform that that perform that enters into the login website for the different actors, and then fill for the can be done by the primary and verification, checking and issuing the passport can be done for the supporting actor.

STAR UML DIAGRAMS:

USE CASE DIAGRAM:

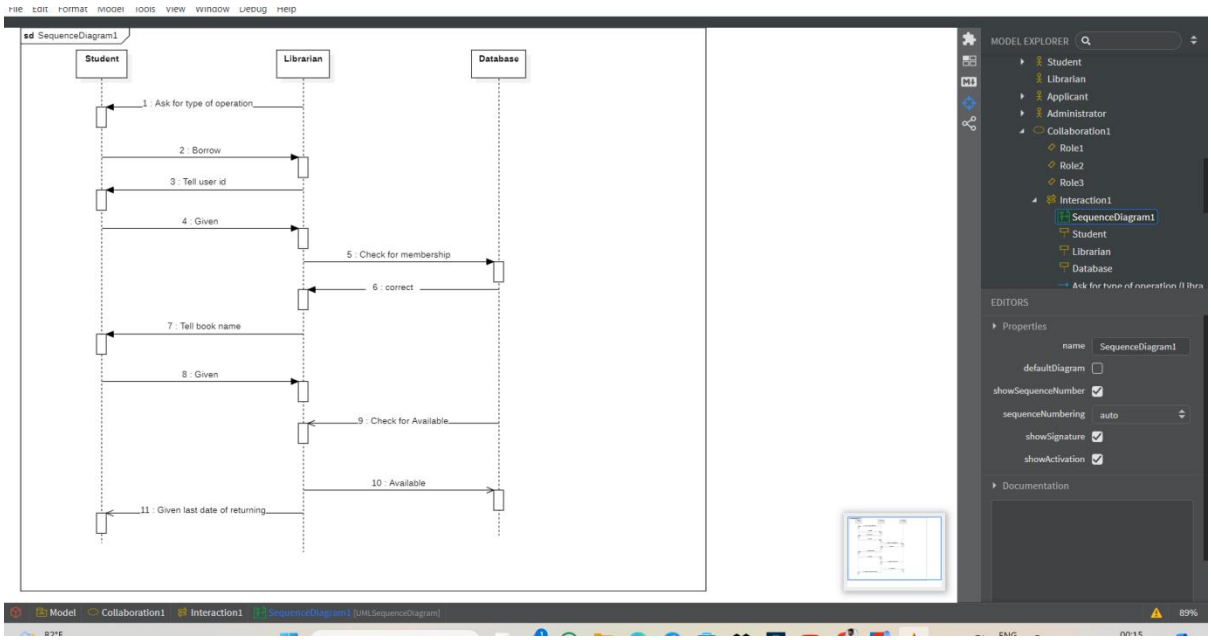


ACTIVITY DIAGRAM:

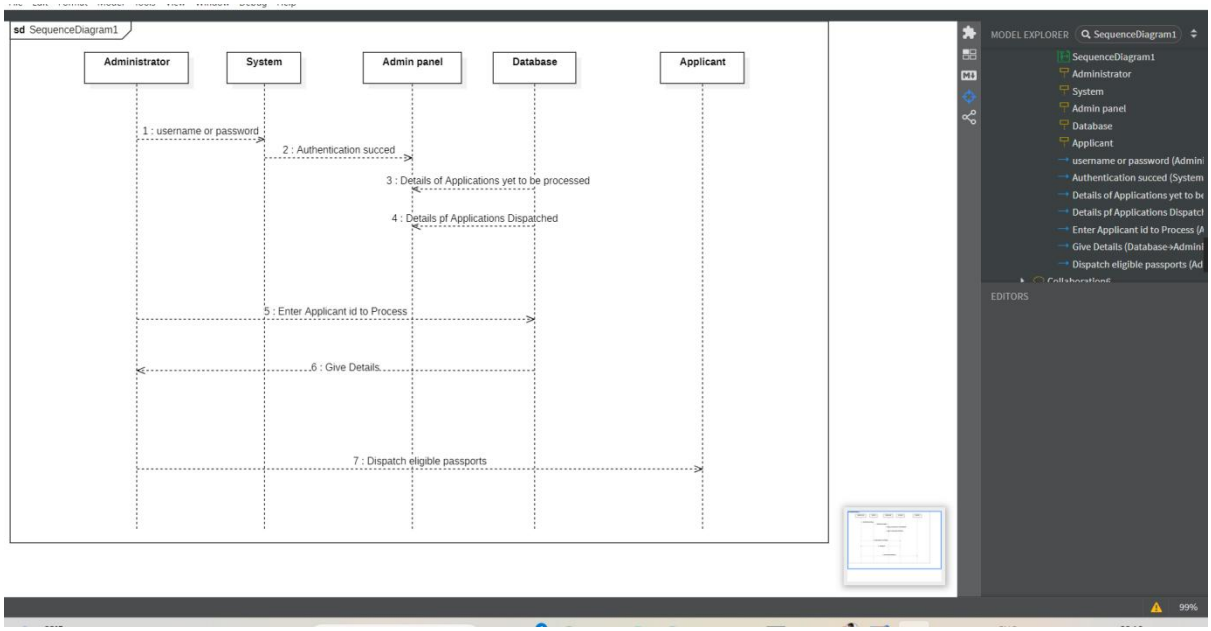


SEQUENCE DIAGRAM:

NEW REGISTRATION

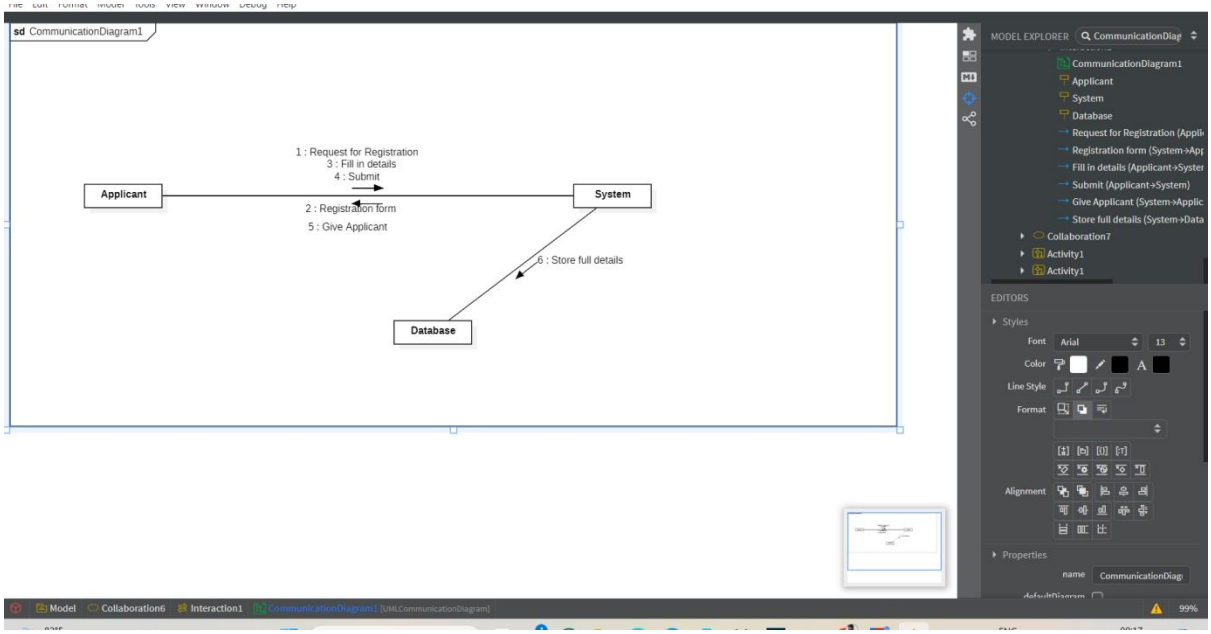


ADMIN PANEL

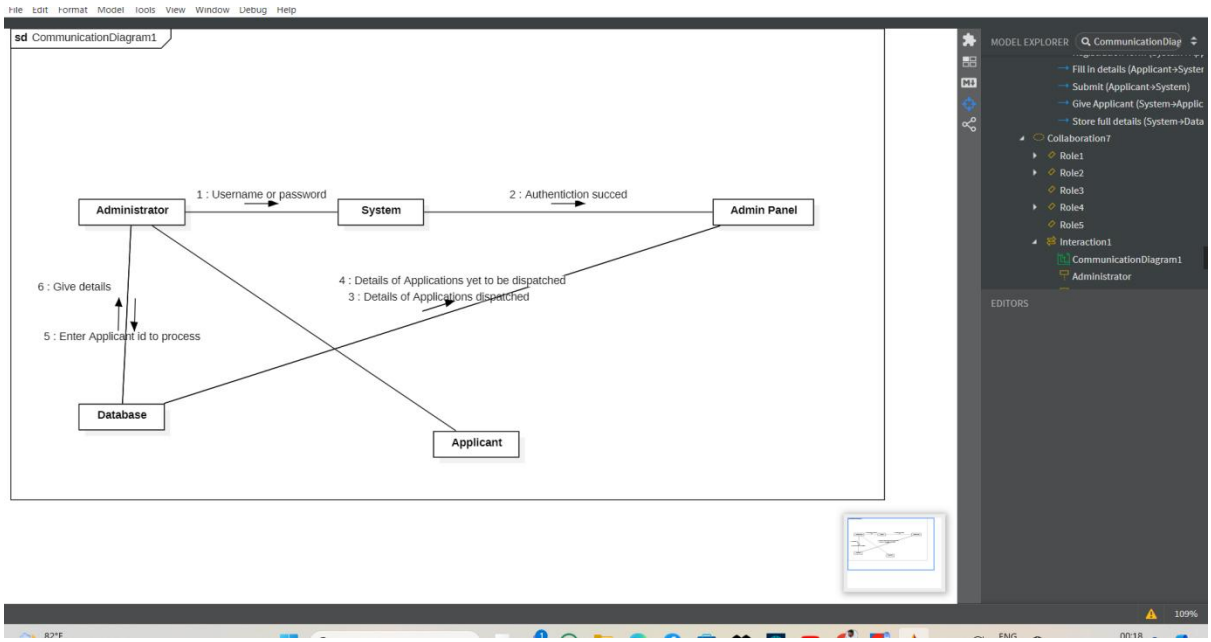


COMMUNICATION DIAGRAM:

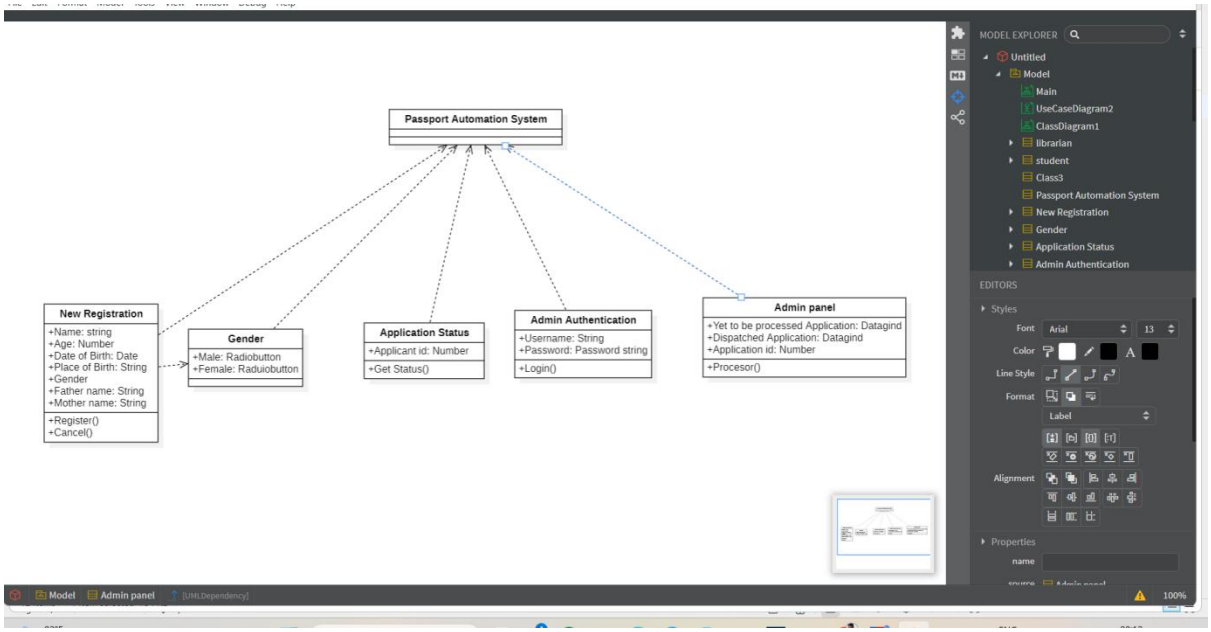
NEW REGISTRATION



ADMIN PANEL



CLASS DIAGRAM:



ALGORITHM:

1. Draw the class diagram and then save it.

2. Open the saved file with star uml.

3. Then go to tools at right corner.

4. Select the code language which we can prefer.

5. Then we have the code generator in tools so give access to it.

6. Then go to tools and select the language then it will take us to another window.

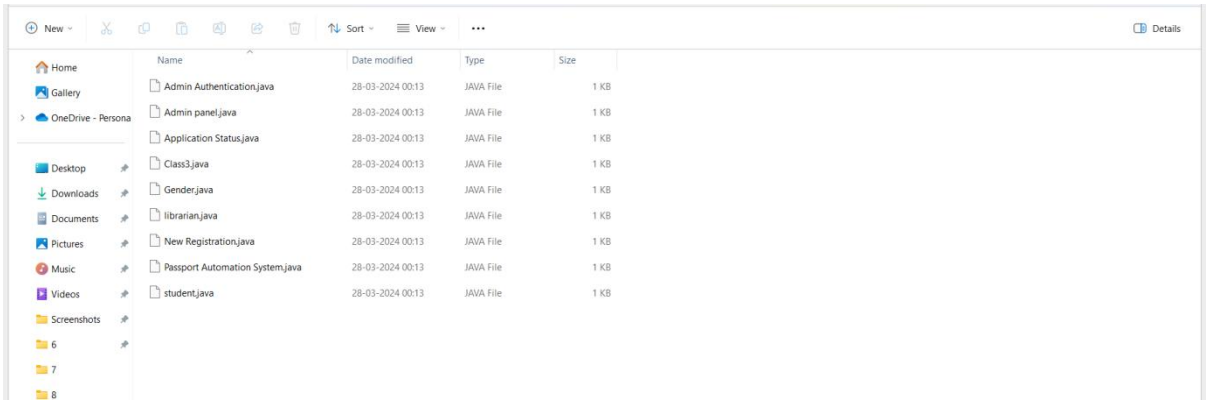
7. There select the class attribute to generate the code.

8. Then it will ask for the directory to store the code.

9. Select the directory and click on generate code.

10. Then code generated successfully.

OUTPUT:



ADMIN PANEL:

```
import java.util.*;
/**
 *
 */
public class Admin  panel {
    /**
     * Default constructor
     */
    public Admin  panel() {
    }

    /**
     *
     */
    public Datagind yet to be processed application;

    /**
     *
     */
```

```
    public Datagind- Dispatched Application;

  /**
     *
     */
    public Number Application id;

    /**
     *
     */
    public void Process() {
        // TODO implement here
    }
}
```

ADMIN AUTHENTICATION:

```
import java.util.*;
/**
 *
 */
public class Admin Authentication {

    /**
     * Default constructor
     */
    public Admin Authentication() {
    }

    /**
     *
     */
```

```
    public string Username;

    /**
     *
     */
    public password string password;
    /**
     *
     */
    public void Login() {
        // TODO implement here
    }
}
```

APPLICATION STATUS:

```
import java.util.*;
/**
 *
 */
public class Application Status {

    /**
     * Default constructor
     */
    public Application Status() {
    }

    /**
     *
     */
    public Number Application id;
```

```
    /**
     *
     */
    public void Attribute2;
     /**
     *
     */
    public void Get Status() {
        // TODO implement here
    }
}
```

GENDER:

```
import java.util.*;
/**
 *
 */
public class Gender {
     /**
     * Default constructor
     */
    public Gender() {
    }
     /**
     *
     */
    public Radiobutton Male;
     /**
     *
     */
    public Radio button Female;
     /**
```

```
     *
     */
    public void Attribute3;
}
```

NEW REGISTRATION:

```
import java.util.*;
/**
 *
 */
public class New Registration {
     /**
     * Default constructor
     */
    public New Registration() {
    }

    /**
     *
     */
    public string Name;

    /**
     *
     */
    public Number Age;

    /**
     *
     */
    public string Date of Birth;
```

```
/**
 *
 */
public void Gender;

/**
 *
 */
public string Father name;

/**
 *
 */
public string Mother name;

/**
 *
 */
public void Attribute7;

/**
 *
 */
public void Attribute8;

/**
 *
 */
public void Register() {
    // TODO implement here
}
```

```
    /**
     *
     */
    public void cancel() {
        // TODO implement here
    }
}
```

PASSPORT AUTOMATION SYSTEM:

```
import java.util.*;
/**
 *
 */
public class Passport Automation System {
    /**
     * Default constructor
     */
    public Passport Automation System() {
    }
}
```

RESULT:

Thus, the UML diagrams for Passport Automation system in StarUML has been designed and java code template was generated for coding successfully.