

```
}
```

CASHIER:

```
public class cashier
{

    public Integer name;

    public Integer cast;

    public void amount()
    {
    }

}
```

CENTRAL SYSTEM:

```
public class central sys
{

    private Integer productname;public Integer productdetails;

    public void printbill()
    {
    }
    public void validatecard()
    {
    }

}
```

OOAD LAB

REGISTER NO:

RESULT:

Thus the diagrams[use case, activity, sequence, collaboration, class, state chart, component, deployment, package] for the credit card processing system has been designed, executed and output is verified.

EX:NO:08	SOFTWARE PERSONNEL MANAGEMENT SYSTEM
DATE:	

AIM:

To draw the diagrams [Usecase, Class, Activity, Sequence, Collaboration, State chart, Deployment, Component, package] for software personnel management system.

SOFTWARE REQUIREMENTS SPECIFICATION:

	SOFTWARE REQUIREMENTS SPECIFICATION
1.0	Hardware Requirements
1.1	Software Requirements
1.2	Project description
1.3	Reference

1.0 HARDWARE REQUIREMENTS:

Intel Pentium Processor I3/I5

1.1 SOFTWARE REQUIREMENTS:

Rational Rose/Argo UML

1.2 PROJECT DESCRIPTION:

This software is designed for the process of knowing the details of a person works in a software company. The details are being stored in the Central Management System for the cross checking the person's details.

1.3 REFERENCES:

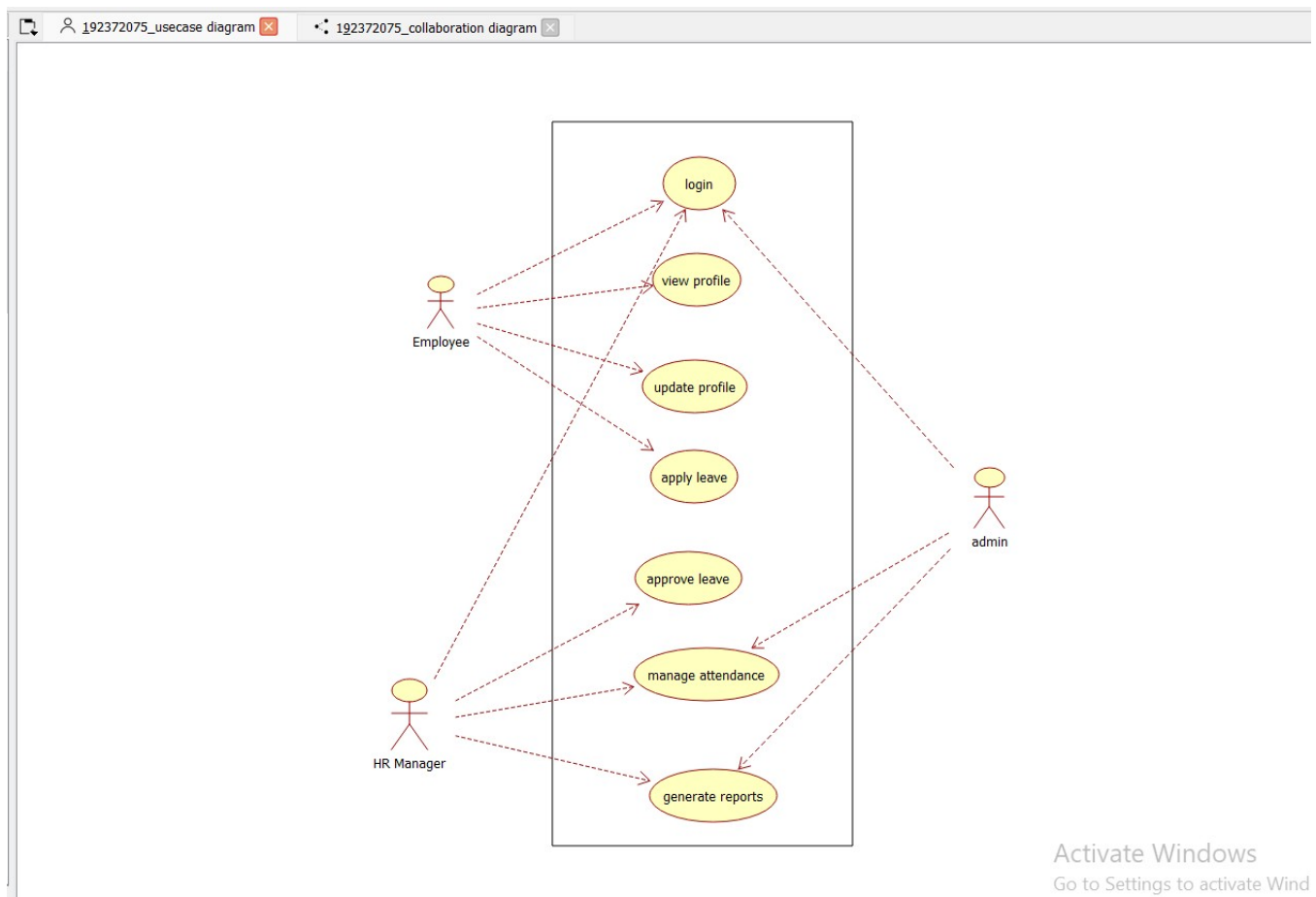
IEEE Software Requirement Specification format.

USECASE DIAGRAM:

This diagram will contain the actors, usecases which are:

ACTORS: Employee, HR, Central System

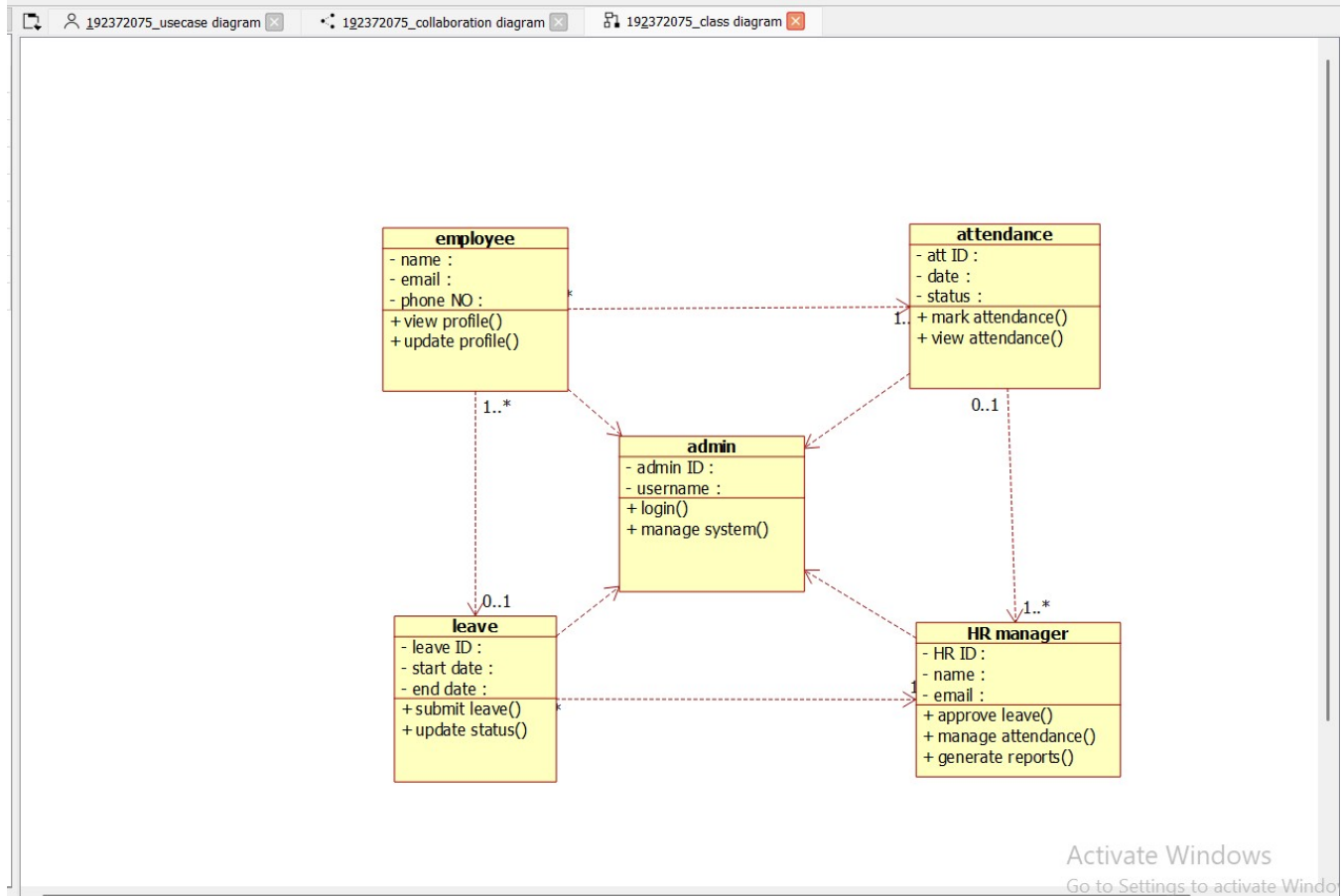
USECASE: Name and address, qualification, experience, internet, loan, verificatio



CLASS DIAGRAM:

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

CLASSES	ATTRIBUTES	OPERATIONS
Central Management System	Employee name Employee number	Tax() Loan()
Employee1	Employee details	Leave taken()
HR	Check details	Loss of pay()

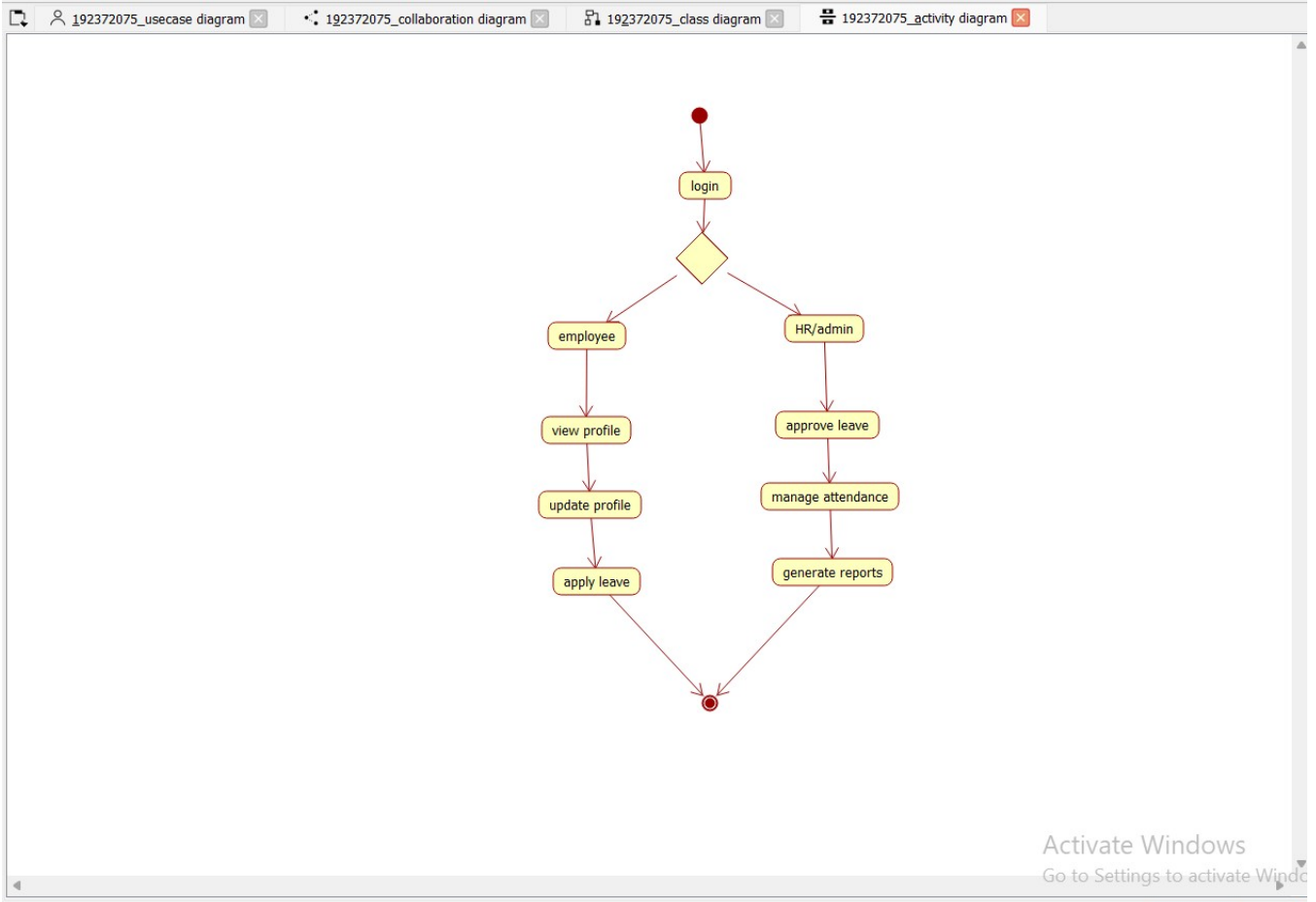


ACTIVITY DIAGRAM:

This diagram will have the activities as start point, end point, decision boxes as:

ACTIVITIES: Enter the option to check, enter the salary, enter the working days, leave taken, loss of pay

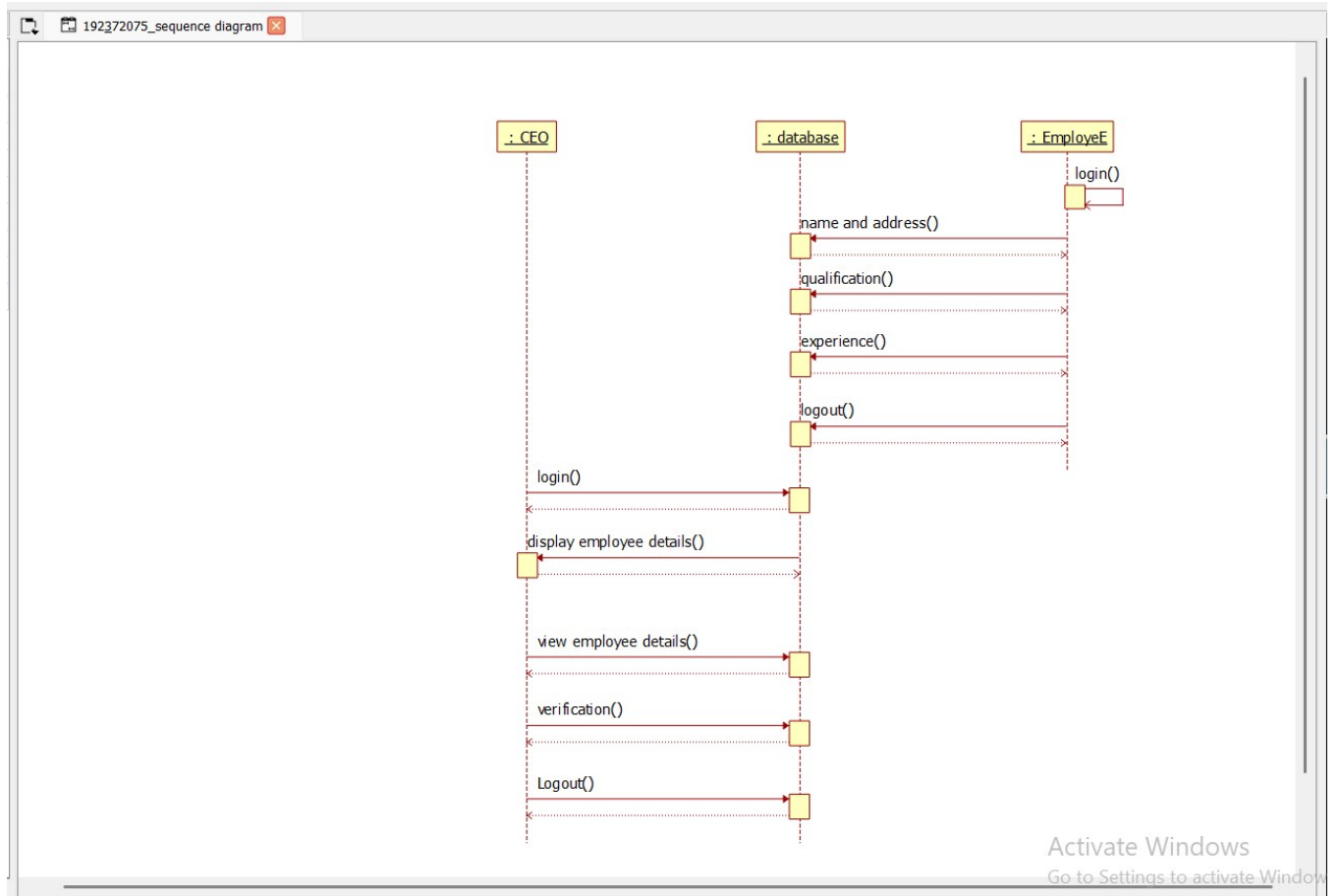
DECISION BOX: Option to check



SEQUENCE DIAGRAM:

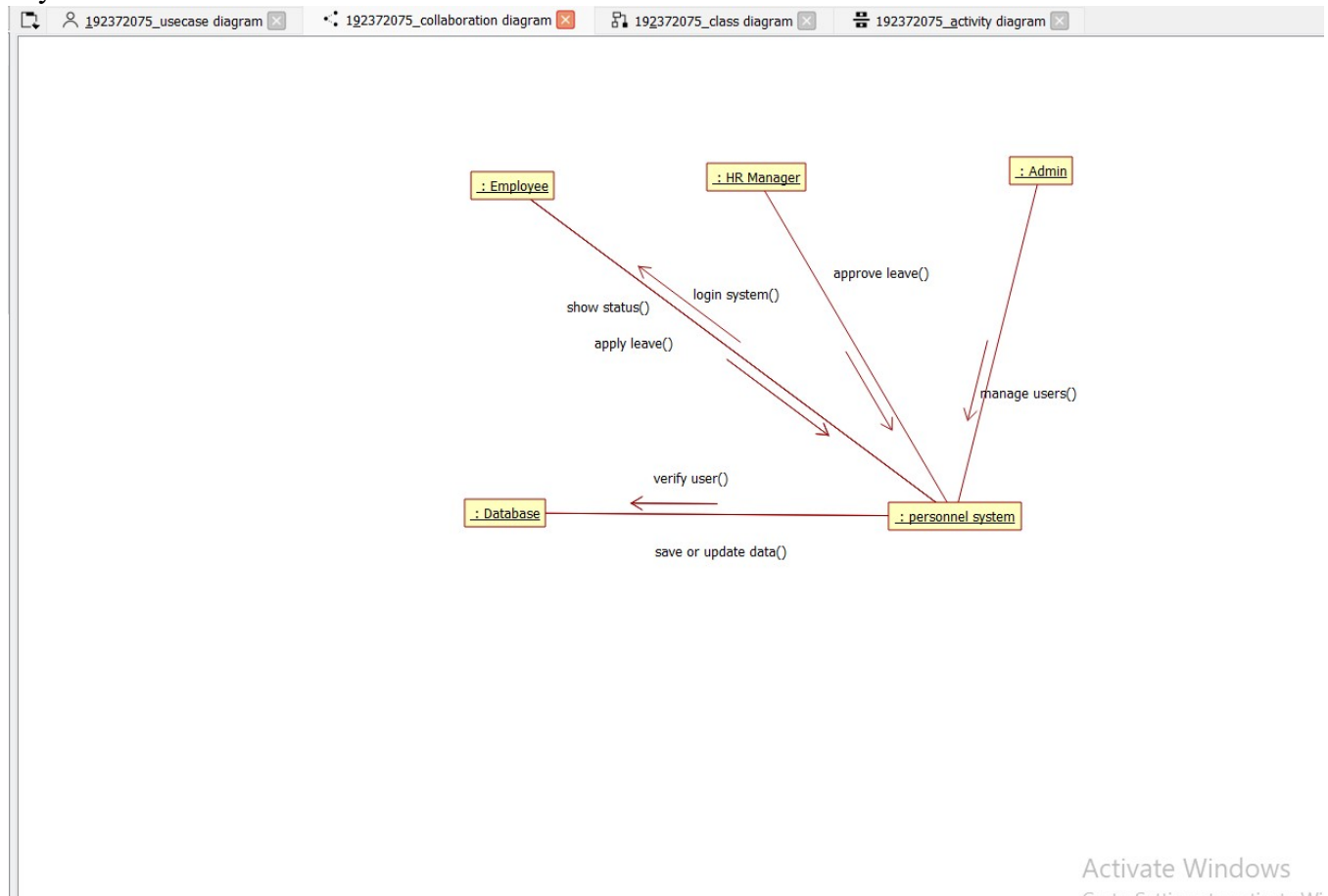
This diagram consists of the objects, messages and return messages

OBJECT: Employee, HR, Central System



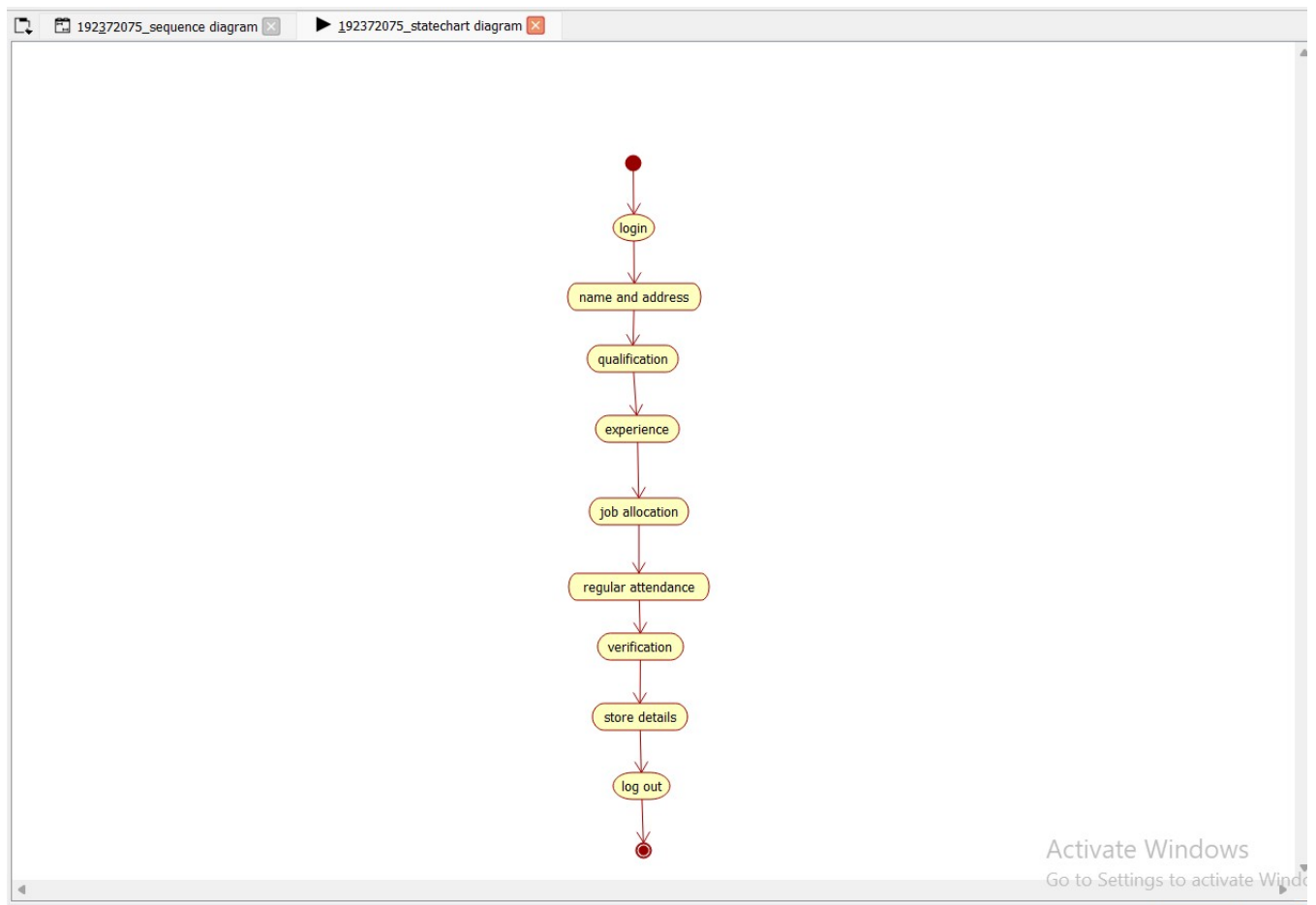
COLLABORATION DIAGRAM:

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



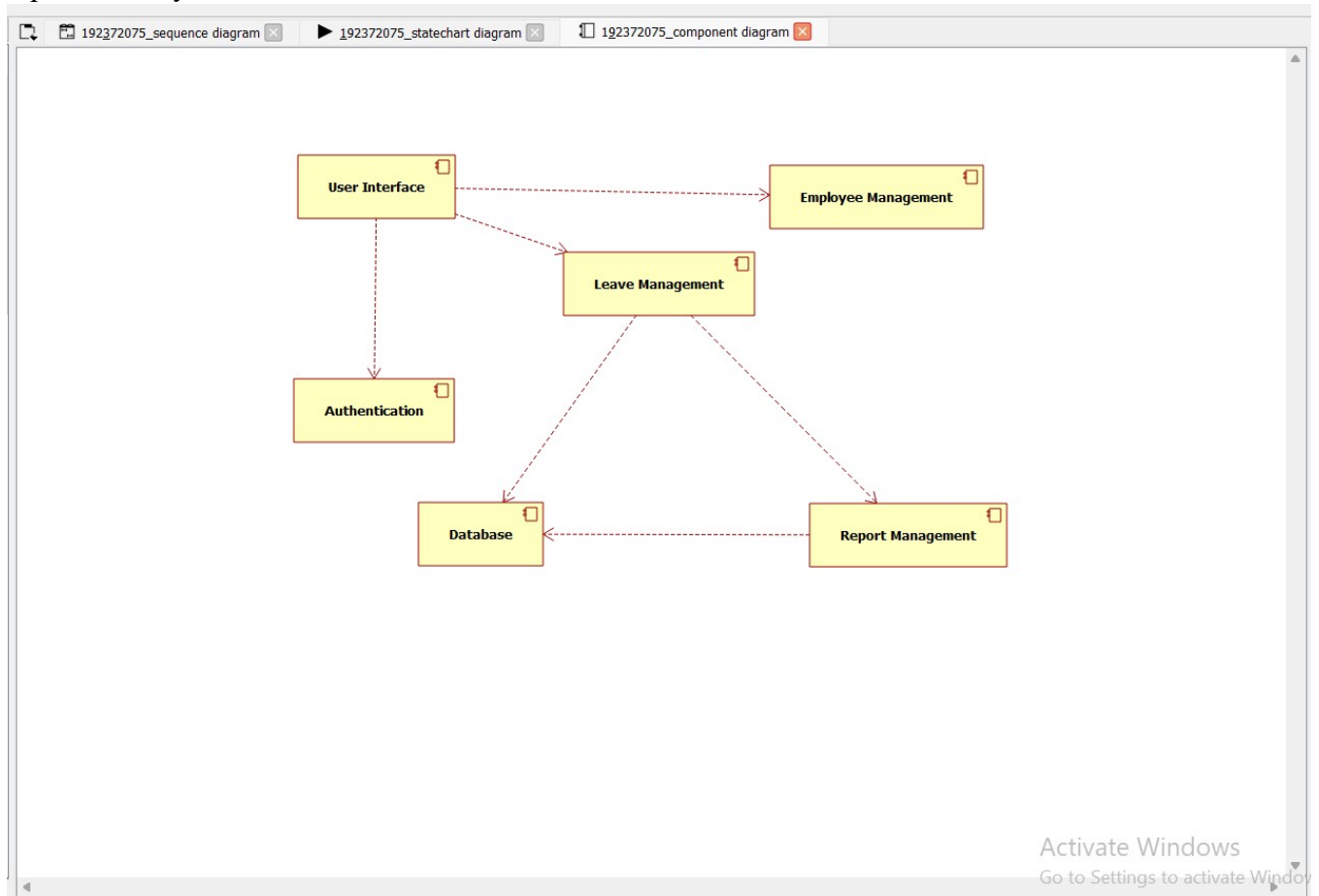
STATECHART DIAGRAM:

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 diagram are drawn for a single class to show the lifetime behaviour of a single objects.



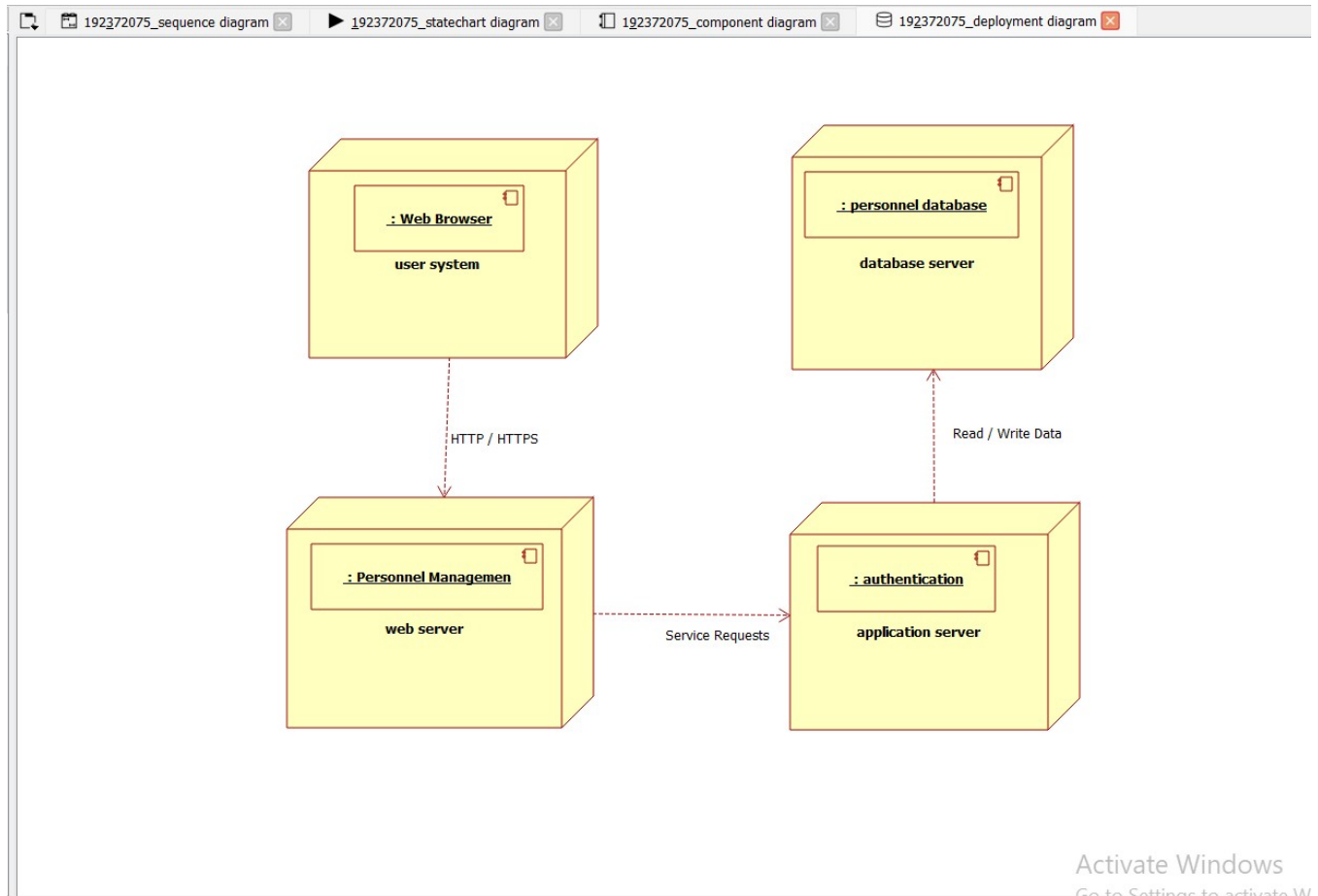
COMPONENT DIAGRAM:

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 systems. It is represented by boxed figure. Dependencies are represented by 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 3- dimensional box. Dependencies are represented by communication association.

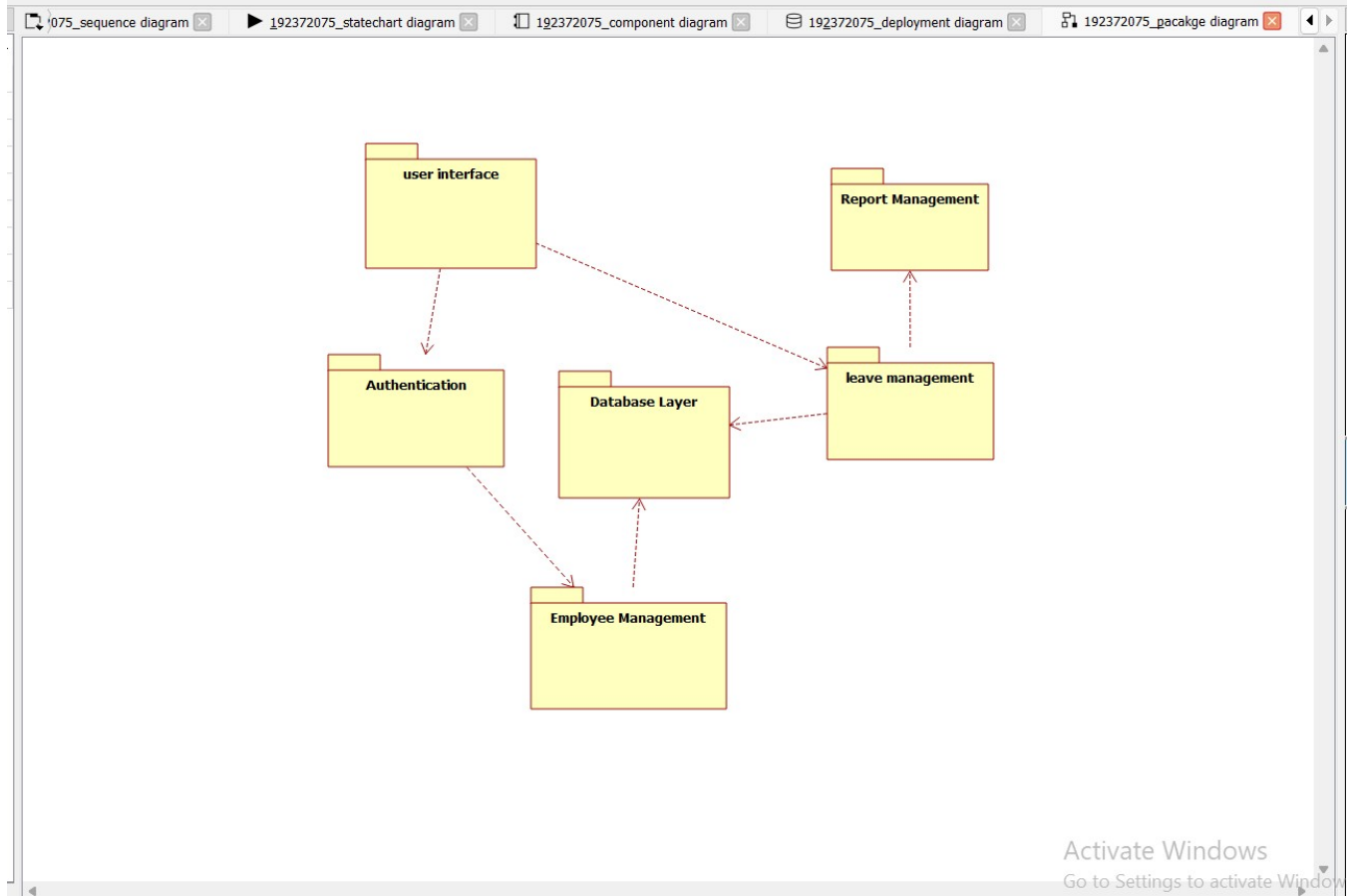


PACKAGE DIAGRAM:

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 layer. They are

- o User interface layer
- o Domain layer
- o Technical services layer



PROGRAM CODING:

EMPLOYEE:

Public class employee

```
{  
    Public integer employee details;  
    Public integer salary;  
    Public void leave taken()  
    {  
    }  
    Public void employee()  
    {  
    }  
}
```

HUMAN RESOURCES:

Public class HR

```
{  
    Public integer check details;  
    Public void loss of pay()  
    {  
    }  
    Public void tax()  
    {  
    }  
  
    Public void HR()
```