

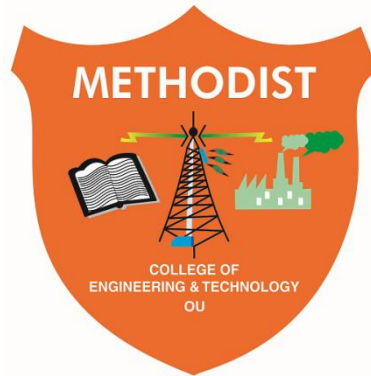


Estd:2008

# METHODIST

## COLLEGE OF ENGINEERING AND TECHNOLOGY

(Affiliated to Osmania University & Approved by AICTE, New Delhi)



## LABORATORY MANUAL

## SOFTWARE ENGINEERING LAB

BE V Semester (AICTE Modal Curriculum): 2020-21

NAME: \_\_\_\_\_

ROLL NO: \_\_\_\_\_

BRANCH: \_\_\_\_\_ SEM: \_\_\_\_\_

## DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

*Empower youth- Architects of Future World*



Estd:2008

# METHODIST

## COLLEGE OF ENGINEERING AND TECHNOLOGY

### **VISION**

To produce ethical, socially conscious and innovative professionals who would contribute to sustainable technological development of the society.

### **MISSION**

To impart quality engineering education with latest technological developments and interdisciplinary skills to make students succeed in professional practice.

To encourage research culture among faculty and students by establishing state of art laboratories and exposing them to modern industrial and organizational practices.

To inculcate humane qualities like environmental consciousness, leadership, social values, professional ethics and engage in independent and lifelong learning for sustainable contribution to the society.



Estd:2008

**METHODIST**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**DEPARTMENT  
OF  
COMPUTER SCIENCE AND ENGINEERING**

**LABORATORY MANUAL  
SOFTWARE ENGINEERING LAB**

**Prepared  
By  
Mrs. P M Tulasi,  
Assistant Professor.**



**METHODIST**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

## **VISION & MISSION**

### **VISION**

To become a leader in providing Computer Science & Engineering education with emphasis on knowledge and innovation.

### **MISSION**

- To offer flexible programs of study with collaborations to suit industry needs.
- To provide quality education and training through novel pedagogical practices.
- To expedite high performance of excellence in teaching, research and innovations.
- To impart moral, ethical values and education with social responsibility.



**METHODIST**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

## **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

### **PROGRAM EDUCATIONAL OBJECTIVES**

**After 3-5 years of graduation, the graduates will be able to**

**PE01:** Apply technical concepts, Analyze, Synthesize data to Design and create novel products and solutions for the real life problems.

**PE02:** Apply the knowledge of Computer Science Engineering to pursue higher education with due consideration to environment and society.

**PE03:** Promote collaborative learning and spirit of team work through multidisciplinary projects

**PE04:** Engage in life-long learning and develop entrepreneurial skills.



# METHODIST

## COLLEGE OF ENGINEERING AND TECHNOLOGY

### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

#### PROGRAM OUTCOMES

**Engineering graduates will be able to:**

- P01: Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- P02: Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
- P03: Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
- P04: Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
- P05: Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
- P06: The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
- P07: Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
- P08: Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- P09: Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

**P010: Communication:** Communicate effectively on complex engineering activities with the Engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

**P011: Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

**P012: Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

## **PROGRAM SPECIFIC OUTCOMES**

**At the end of 4 years, Computer Science and Engineering graduates at MCET will be able to:**

**PS01:** Apply the knowledge of Computer Science and Engineering in various domains like networking and data mining to manage projects in multidisciplinary environments.

**PS02:** Develop software applications with open-ended programming environments.

**PS03:** Design and develop solutions by following standard software engineering principles and implement by using suitable programming languages and platforms

Course Code	Course Title					Core/ Elective	
<b>PC 531 CS</b>	<b>SOFTWARE ENGINEERING LAB</b>					<b>CORE</b>	
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
-	-	-	-	2	25	50	2
<p><b>Course Objectives</b></p> <ul style="list-style-type: none"> <li>➤ To understand the software engineering methodologies for project development.</li> <li>➤ To gain knowledge about open source tools for Computer Aided Software Engineering(CASE).</li> <li>➤ To develop test plans and test cases to perform various testing.</li> </ul> <p><b>Course Outcomes</b></p> <p>Student will be able to:</p> <ul style="list-style-type: none"> <li>➤ Analyze and design software requirements in an efficient manner.</li> <li>➤ Use open source case tools to develop software</li> <li>➤ Implement the design , debug and test the code</li> </ul>							

## I. FORWARD ENGINEERING

Students have to form a team with a batch size of two or three and take up a case study based project to analyze, plan, design UML models and create a prototypical model (identifying deliverables) by coding the developed designs and finally documenting considering any one example of the following domains:-

1. Academics (Course Registration System, Student marks analyzing system)
2. Health Care ( Expert system to prescribe medicines for given symptoms, Remote Diagnostics, Patient/Hospital Management System)
3. Finance (Banking:ATM/NetBanking, UPI:PayTM/PhonePay,Stocks:Zerodha)
4. E-Commerce ( various online shopping portals like FlipKart/Amazon/Myntra)
5. Logistics (Postal/Courier:IndiaPost/DTDC/UPS/FedEx,Freight:Maersk)
6. Hospitality (Tourism Management:Telangana Tourism/Incredible India,Event anagement:MeraEvents/BookMyShow/Explara/EventBrite)
7. Social Networking ( LinkedIn, FaceBook, Shaadi.com, BharatMatrimony,Tinder)
8. Customer Support (Banking Ombudsman,Indian Consumer ComplaintsForum)
9. Booking/Ticketing(Food:Zomato/Swiggy/BigBasket/Grofers/JioMart, Hotel:OYO/Trivago or Travel:{ Cars:Uber/OLA/Zoom, Railways:IRCTC, Buses:OnlineTSRTC/RedBus/AbhiBus, Flights:MakeMyTrip/Goibibo, Ships:Lakport })

**II. REVERSE ENGINEERING:** Students have to refer any project repository: GitLab /GitHub,execute the code in order to observe its functionalities/features/requirements and by the help of any tool derive the designs from the code for understanding the relationships among various subsystems/classes/components and if the tool partially generates models then identify by associating elements to judge/mark the appropriate relationships.



**III. TESTING:** Prepare Test Plan and develop Test Case Hierarchy to monitor or uncover/report errors using manual/automated testing tools

**Software Required:** *StarUML/Umbrello, NetBeans/Eclipse IDE, XAMPP/MEAN stack, JUnit, JMeter, Selenium, Bugzilla*



Estd:2008

# METHODIST

## COLLEGE OF ENGINEERING AND TECHNOLOGY

### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**Course Outcomes (CO's):**

**SUBJECT NAME : SOFTWARE ENGINEERING LAB**

**CODE : PC531CS**

**SEMESTER : V**

<b>CO No.</b>	<b>Course Outcomes</b>	<b>Taxonomy Level</b>
<b>PC531CS.1</b>	Interpret a variety of approaches and perspectives of system development.	Understanding
<b>PC531CS.2</b>	Identify the requirements which are relevant to the design of a system.	Applying
<b>PC531CS.3</b>	Model software design with a set of objects and their relationships using structural modeling.	Applying
<b>PC531CS.4</b>	Take part in using advanced & behavioral modeling to develop a case study.	Analysing
<b>PC531CS.5</b>	Design the activities with the help of behavioral modeling.	Creating
<b>PC531CS.6</b>	Develop components through architectural modeling.	Creating



Estd:2008

# METHODIST

## COLLEGE OF ENGINEERING AND TECHNOLOGY

### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

#### GENERAL LABORATORY INSTRUCTIONS

1. Students are advised to come to the laboratory at least 5 minutes before (to starting time), those who come after 5 minutes will not be allowed into the lab.
2. Plan your task properly much before to the commencement, come prepared to the lab with the program / experiment details.
3. Student should enter into the laboratory with:
  - a. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure, Program, Expected Output, etc.,) filled in for the lab session.
  - b. Laboratory Record updated up to the last session experiments.
  - c. Formal dress code and Identity card.
4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.
5. Execute your task in the laboratory, and record the results / output in the lab observation note book, and get certified by the concerned faculty.
6. All the students should be polite and cooperative with the laboratory staff, must maintain the discipline and decency in the laboratory.
7. Computer labs are established with sophisticated and high end branded systems, which should be utilized properly.
8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment, misbehaviours with the staff and systems etc., will attract severe punishment.
9. Students must take the permission of the faculty in case of any urgency to go out. If anybody found loitering outside the lab / class without permission during working hours will be treated seriously and punished appropriately.
10. Students should SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly.

**Head of the Department**

**Principal**



# METHODIST

## COLLEGE OF ENGINEERING AND TECHNOLOGY

### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

#### **CODE OF CONDUCT FOR THE LABORATORY**

- All students must observe the dress code while in the laboratory
- Footwear is NOT allowed
- Foods, drinks and smoking are NOT allowed
- All bags must be left at the indicated place
- The lab timetable must be strictly followed
- Be PUNCTUAL for your laboratory session
- All programs must be completed within the given time
- Noise must be kept to a minimum
- Workspace must be kept clean and tidy at all time
- All students are liable for any damage to system due to their own negligence
- Students are strictly PROHIBITED from taking out any items from the laboratory
- Report immediately to the lab programmer if any damages to equipment

#### **BEFORE LEAVING LAB:**

- Arrange all the equipment and chairs properly.
- Turn off / shut down the systems before leaving.
- Please check the laboratory notice board regularly for updates.

**Lab In – charge**



Estd:2008

# METHODIST

## COLLEGE OF ENGINEERING AND TECHNOLOGY

### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

### LIST OF EXPERIMENTS

Sl.No.	Name of the Experiment	Date of Experiment	Date of Submission	Page No	Faculty Signature
1.	Academics (Course Registration System, Student marks analyzing system)			9	
2.	Health Care ( Expert system to prescribe medicines for given symptoms, Remote Diagnostics, Patient/Hospital Management System)			26	
3.	Finance (Banking: ATM/NetBanking, UPI: PayTM/ PhonePay,Stocks: Zerodha)			35	
4.	E-Commerce ( various online shopping portals like FlipKart /Amazon/Mynta)			46	
5.	Logistics (Postal /Courier: IndiaPost /DTDC/ UPS/ FedEx,Freight:Maersk)			50	
6.	Hospitality(Tourism Management : Telangana Tourism /Incredible India Event Management: MeraEvents BookMyShow /Explara/EventBrite)			57	
7.	Social Networking ( LinkedIn, FaceBook, Shaadi.com, BharatMatrimony,Tinder)			65	

8.	Customer Support (Banking Ombudsman, Indian Consumer Complaints Forum)			69	
9.	Booking/Ticketing (Food : Zomato/ Swiggy/BigBasket/Grofers/JioMart, Hotel: OYO/Trivago or Travel: { Cars: Uber/OLA/Zoom, Railways: IRCTC, Buses: Online TSRTC/RedBus/AbhiBus, Flights: MakeMyTrip/Goibibo, Ships: Lakport } ) Travel			72	
10.	<b>REVERSE ENGINEERING</b>			88	
11.	<b>TESTING</b>			91	

### **ADDITIONAL EXPERIMENTS**

<b>Sl.No.</b>	<b>Name of the Experiment</b>	<b>Date of Experiment</b>	<b>Date of Submission</b>	<b>Page No</b>	<b>Faculty Signature</b>
12.	Education system- online eamcet examination case study			99	
13.	Applying for online passport system back end verification process			103	
14.	Govt. Services: applying certificates 1. Birth certificate 2. caste certificate 3. migration certificate 4. Death certificate			108	

**PROGRAM I : FORWARD ENGINEERING****THEORY :****SOFTWARE DEVELOPMENT LIFE CYCLE**

SDLC, or Software Development Life Cycle, is a set of steps used to create software applications. These steps divide the development process into tasks that can then be assigned, completed, and measured.

**What is the Software Development Life Cycle?**

Software Development Life Cycle is the application of standard business practices to building software applications. It's typically divided into six to eight steps: Planning, Requirements, Design, Build, Document, Test, Deploy, and Maintain. Some project managers will combine, split, or omit steps, depending on the project's scope. These are the core components recommended for all software development projects.

SDLC is a way to measure and improve the development process. It allows a fine-grain analysis of each step of the process. This, in turn, helps companies maximize efficiency at each stage. As computing power increases, it places a higher demand on software and developers. Companies must reduce costs, deliver software faster, and meet or exceed their customers' needs. SDLC helps achieve these goals by identifying inefficiencies and higher costs and fixing them to run smoothly.

**How the Software Development Life Cycle Works?**

The Software Development Life Cycle simply outlines each task required to put together a software application. This helps to reduce waste and increase the efficiency of the development process. Monitoring also ensures the project stays on track, and continues to be a feasible investment for the company.

Many companies will subdivide these steps into smaller units. Planning might be broken into technology research, marketing research, and a cost-benefit analysis. Other steps can merge with each other. The Testing phase can run concurrently with the Development phase, since developers need to fix errors that occur during testing.

**The Seven Phases of the SDLC****1. Planning**

In the Planning phase, project leaders evaluate the terms of the project. This includes calculating labor and material costs, creating a timetable with target goals, and creating the project's teams and leadership structure.

Planning can also include feedback from stakeholders. Stakeholders are anyone who stands to benefit from the application. Try to get feedback from potential customers, developers, subject matter experts, and sales reps.

Planning should clearly define the scope and purpose of the application. It plots the course and provisions the team to effectively create the software. It also sets boundaries to help keep the project from expanding or shifting from its original purpose.

## 2. Define Requirements

Defining requirements is considered part of planning to determine what the application is supposed to do and its requirements. For example, a social media application would require the ability to connect with a friend. An inventory program might require a search feature.

Requirements also include defining the resources needed to build the project. For example, a team might develop software to control a custom manufacturing machine. The machine is a requirement in the process.

## 3. Design and Prototyping

The Design phase models the way a software application will work. Some aspects of the design include:

- **Architecture** – Specifies programming language, industry practices, overall design, and use of any templates or boilerplate
- **User Interface** – Defines the ways customers interact with the software, and how the software responds to input
- **Platforms** – Defines the platforms on which the software will run, such as Apple, Android, Windows version, Linux, or even gaming consoles
- **Programming** – Not just the programming language, but including methods of solving problems and performing tasks in the application
- **Communications** – Defines the methods that the application can communicate with other assets, such as a central server or other instances of the application\
- **Security** – Defines the measures taken to secure the application, and may include SSL traffic encryption, password protection, and secure storage of user credentials

Prototyping can be a part of the Design phase. A prototype is like one of the early versions of software in the Iterative software development model. It demonstrates a basic idea of how the application looks and works. This “hands-on” design can be shown to stakeholders. Use feedback to improve the application. It’s less expensive to change the Prototype phase than to rewrite code to make a change in the Development phase.

## 4. Software development

This is the actual writing of the program. A small project might be written by a single developer, while a large project might be broken up and worked by several teams. Use an Access Control or Source Code Management application in this phase. These systems help developers track changes to the code. They also help ensure compatibility between different team projects and to make sure target goals are being met.



The coding process includes many other tasks. Many developers need to brush up on skills or work as a team. Finding and fixing errors and glitches is critical. Tasks often hold up the development process, such as waiting for test results or compiling code so an application can run. SDLC can anticipate these delays so that developers can be tasked with other duties.

Software developers appreciate instructions and explanations. Documentation can be a formal process, including writing a user guide for the application. It can also be informal, like comments in the source code that explain why a developer used a certain procedure. Even companies that strive to create software that's easy and intuitive benefit from the documentation.

Documentation can be a quick guided tour of the application's basic features that display on the first launch. It can be video tutorials for complex tasks. Written documentation like user guides, troubleshooting guides, and FAQ's help users solve problems or technical questions.

## **5. Testing**

It's critical to test an application before making it available to users. Much of the testing can be automated, like security testing. Other testing can only be done in a specific environment – consider creating a simulated production environment for complex deployments. Testing should ensure that each function works correctly. Different parts of the application should also be tested to work seamlessly together—performance test, to reduce any hangs or lags in processing. The testing phase helps reduce the number of bugs and glitches that users encounter. This leads to a higher user satisfaction and a better usage rate.

## **6. Deployment**

In the deployment phase, the application is made available to users. Many companies prefer to automate the deployment phase. This can be as simple as a payment portal and download link on the company website. It could also be downloading an application on a smartphone.

Deployment can also be complex. Upgrading a company-wide database to a newly-developed application is one example. Because there are several other systems used by the database, integrating the upgrade can take more time and effort.

## **7. Operations and Maintenance**

At this point, the development cycle is almost finished. The application is done and being used in the field. The Operation and Maintenance phase is still important, though. In this phase, users discover bugs that weren't found during testing. These errors need to be resolved, which can spawn new development cycles.

In addition to bug fixes, models like Iterative development plan additional features in future releases. For each new release, a new Development Cycle can be launch

## **Unified Modeling Language (UML) :**

Unified Modeling Language (UML) is a standardized (ISO/IEC 19501:2005), general-purpose modeling language in the field of Software Engineering. The Unified Modeling Language includes

a set of graphic notation techniques to create visual models of object-oriented software-intensive systems. Unified Modeling Language (UML) combines techniques from data modeling (Entity Relationship Diagrams), business modeling (work flows), object modeling, and component modeling. It can be used with all processes, throughout the software development life cycle, and across different implementation technologies. The UML offers a standard way to visualize a system's architectural blueprints, including elements such as: activities, actors, business processes, database schemas, (logical) components, programming language statements, reusable software components.

### **1. History**

The Unified Modeling Language was developed by Grady Booch, Ivar Jacobson and James Rumbaugh at Rational Software in the 1990s. It was adopted by the Object Management Group (OMG) in 1997, and has been managed by this organization ever since. In 2000 the Unified Modeling Language was accepted by the International Organization for Standardization (ISO) as industry standard for modeling software-intensive systems. The current version of the UML is 2.4.1 published by the OMG in August 2011.

### **2. Introduction**

The Unified Modeling Language (UML) is a standard language for writing software blueprints. The UML may be used to visualize, specify, construct, and document the artefacts of a software intensive system. The UML is appropriate for modelling systems ranging from enterprise information systems to distributed Web-based applications and even to hard real time embedded systems.

Even though it is expressive, the UML is not difficult to understand and to use. Learning to apply the UML effectively starts with forming a conceptual model of the language, which requires learning three major elements: the UML's basic building blocks, the rules that dictate how these building blocks may be put together, and some common mechanisms that apply throughout the language. The UML is only a language and so is just one part of a software development method.

### **3. Where Can the UML Be Used?**

The UML is proposed primarily for software-intensive systems. It has been used effectively for such areas as

Enterprise information systems

Banking and financial services

Telecommunications

Transportation

Defence/aerospace

Retail

Medical electronics

Distributed Web-based services

The UML is not limited to modelling software. In fact, it is expressive enough to model non-software systems, such as workflow in the legal system, the structure and behaviour of a patient healthcare system, and the design of hardware.

#### **4. Building Blocks of the UML**

The vocabulary of the UML comprises of three kinds of building blocks

1. Things
2. Relationships
3. Diagrams

##### **1. Things in the UML**

There are four kinds of things in the UML

1. Structural things
2. Behavioural things
3. Grouping things
4. Annotational things

These things are the basic object-oriented building blocks of the UML. You use them to write well-formed models.

##### **2. Relationships in the UML**

There are four kinds of relationships in the UML

1. Dependency
2. Association
3. Generalization
4. Realization

These relationships are the basic relational building blocks of the UML. You use them to write well-formed models.

##### **3. UML Standard Diagrams**

In the previous chapters we have discussed about the building blocks and other necessary elements of UML. Now we need to understand where to use those elements.

The elements are like components which can be associated in different ways to make complete UML pictures which is known as diagram. So it is very important to understand the different diagrams to implement the knowledge in real life systems.

Any complex system is best understood by making some kind of diagrams or pictures. These diagrams have a better impact on our understanding. So if we look around then we will realize that

the diagrams are not a new concept but it is used widely in different form in different industries. We prepare UML diagrams to understand a system in better and simple way. A single diagram is not enough to cover all aspects of the system. So UML defines various kinds of diagrams to cover most of the aspects of a system.

You can also create your own set of diagrams to meet your requirements. There are two broad categories of diagrams and then are again divided into sub-categories,

### 3.1. Structural Diagrams

### 3.2. Behavioural Diagrams

#### 3.1 Structural Diagrams

The structural diagrams represent the static aspect of the system. These static aspects represent those parts of a diagram which forms the main structure and therefore stable. These static parts are represented by classes, interfaces, objects, components and nodes.

The four structural diagrams are:

Class diagram

Object diagram

Component diagram

Deployment diagram

##### 3.1.1 Class Diagrams

Class diagrams are the most common diagrams used in UML. Class diagram consists of classes, interfaces, associations and collaboration. Class diagrams basically represent the object oriented view of a system which is static in nature. Active class is used in a class diagram to represent the concurrency of the system. Class diagram represents the object orientation of a system. So it is generally used for development purpose. This is the most widely used diagram at the time of system construction.

##### 3.1.2 Object Diagram

Object diagrams can be described as an instance of class diagram. So these diagrams are more close to real life scenarios where we implement a system. Object diagrams are a set of objects and their relationships just like class diagrams and also represent the static view of the system. The usage of object diagrams is similar to class diagrams but they are used to build prototype of a system from practical perspective.

##### 3.1.3 Component Diagram

Component diagrams represent a set of components and their relationships. These components

consist of classes, interfaces or collaborations. So Component diagrams represent the implementation view of a system. During design phase software artifacts (classes, interfaces etc.) of a system are arranged in different groups depending upon their relationship. Now these groups are known as components. Finally, component diagrams are used to visualize the implementation.

### 3.1.4 Deployment Diagram

Deployment diagrams are a set of nodes and their relationships. These nodes are physical entities where the components are deployed. This is generally used by the deployment team.

Note: If the above descriptions and usages are observed carefully then it is very clear that all the diagrams are having some relationship with one another. Component diagrams are dependent upon the classes, interfaces etc. which are part of class/object diagram. Again the deployment diagram is dependent upon the components which are used to make a component diagrams.

## 3.2. Behavioral Diagrams

Any system can have two aspects, static and dynamic. So a model is considered as complete when both the aspects are covered fully.

Behavioral diagrams basically capture the dynamic aspect of a system. Dynamic aspect can be further described as the changing/moving parts of a system.

UML has the following five types of behavioral diagrams,

Use case diagram

Sequence diagram

Collaboration diagram

State-chart diagram

Activity diagram

### 3.2.1 Use case Diagram

Use case diagrams are a set of use cases, actors and their relationships. They represent the use case view of a system.

A use case represents a particular functionality of a system. So use case diagram is used to describe the relationships among the functionalities and their internal/external controllers. These controllers are known as actors.

### 3.2.2 Sequence Diagram

A sequence diagram is an interaction diagram. From the name it is clear that the diagram deals with some sequences, which are the sequence of messages flowing from one object to another.

Interaction among the components of a system is very important from implementation and

execution perspective.

### 3.2.3 Collaboration Diagram

Collaboration diagram is another form of interaction diagram. It represents the structural organization of a system and the messages sent/received. Structural organization consists of objects and links.

The purpose of collaboration diagram is similar to sequence diagram. But the specific purpose of collaboration diagram is to visualize the organization of objects and their interaction.

### 3.2.4 State-chart Diagram

Any real time system is expected to be reacted by some kind of internal/external events. These events are responsible for state change of the system.

State-chart diagram is used to represent the event driven state change of a system. It basically describes the state change of a class, interface etc. State-chart diagram is used to visualize the reaction of a system by internal/external factors.

### 3.2.5 Activity Diagram

Activity diagram describes the flow of control in a system. So it consists of activities and links. The flow can be sequential, concurrent or branched.

Activities are nothing but the functions of a system. Numbers of activity diagrams are prepared to capture the entire flow in a system. Activity diagrams are used to visualize the flow of controls in a system. This is prepared to have an idea of how the system will work when executed.

Note: Dynamic nature of a system is very difficult to capture. So UML has provided features to capture the dynamics of a system from different angles. Sequence diagrams and collaboration diagrams are isomorphic so they can be converted from one another without losing any information. This is also true for state-chart and activity diagram.

**PROGRAM 1: Academics (Course Registration System, Student marks analyzing system)****1.1 AIM : Course Registration System****1.PROBLEM STATEMENT:**

The software which displays the list of courses available for the mark that the student get and the student can able to allocate best course from the choice available. It displays and allocates courses based on student ranking. The student ranking is based on marks, caste and community. Based on caste and community the ranking may varied. This software allows the student to choose the best college for the available courses.

**2.OVERALL DESCRIPTIONS:****2.1MODULES:**

- 1.Login
- 2.student\_info
- 3.course\_details
- 4.course\_registration
- 5.confirmation

**2.2MODULE DELIVARABLES:****Login:**

Basicflow:To authenticate the user,the student has to enter username and password

Alternateflow:If the password is wrong,it will ask the student to answer security question and retrieve the password

Precondition:The system asks the student to enter the password

Postcondition:On success,the student displays the student information

**Student\_info:**

Basicflow:The system displays the basic information about the student.It should be verified by the user.

Alternateflow:If there is any error in displayed information ,the student can do changes.

Precondition:The student checks the basic details displayed.

Postcondition:After verifying all the details,it should be updated and moves on to the next state.

**Course\_details:**

Basicflow:The courses that are available for the student are displayed and the student should select one.

Alternateflow:If the student finds that the courses available for him are not displayed,he can report it.

Precondition:The details of the corses available for his mark should be known by the user.

Postcondition:After selecting the course,it will moves on to the next stage.

### Course\_registration:

Basicflow:All the colleges that are available for the courses are displayed.

Alternateflow:If the student finds that the colleges are displayed in error,he can report it.

Precondition:The student should have to select the colleges available.

Postcondition:After registering the course,it will move on to the next stage.

### Confirmation:

Basicflow:The college and the course that the student choose is displayed.

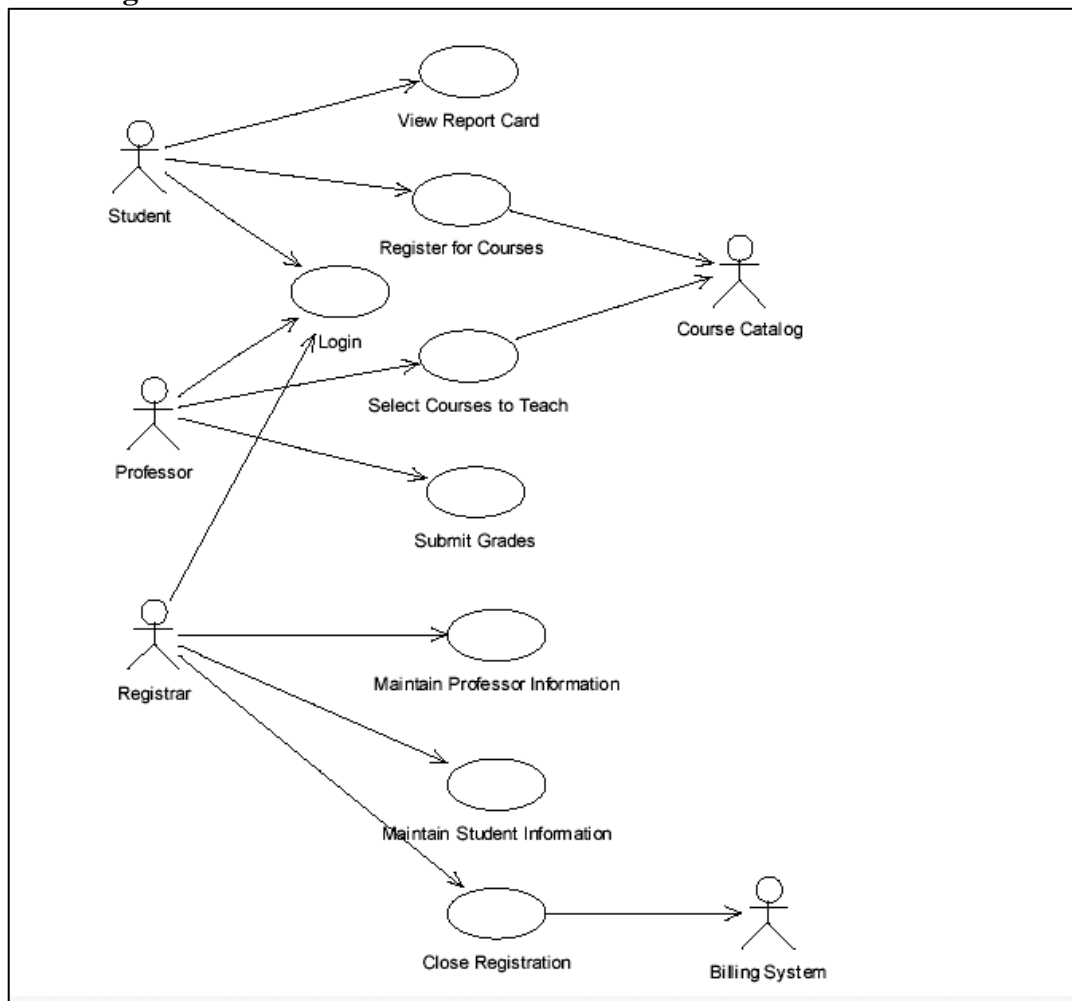
Alternateflow:If the student finds any fault regarding his course/college,he can report it.

Precondition:The student should have to confirm the college and the course.

Postcondition:The course and the college is confirmed and the student confirms it by choosing confirm button.

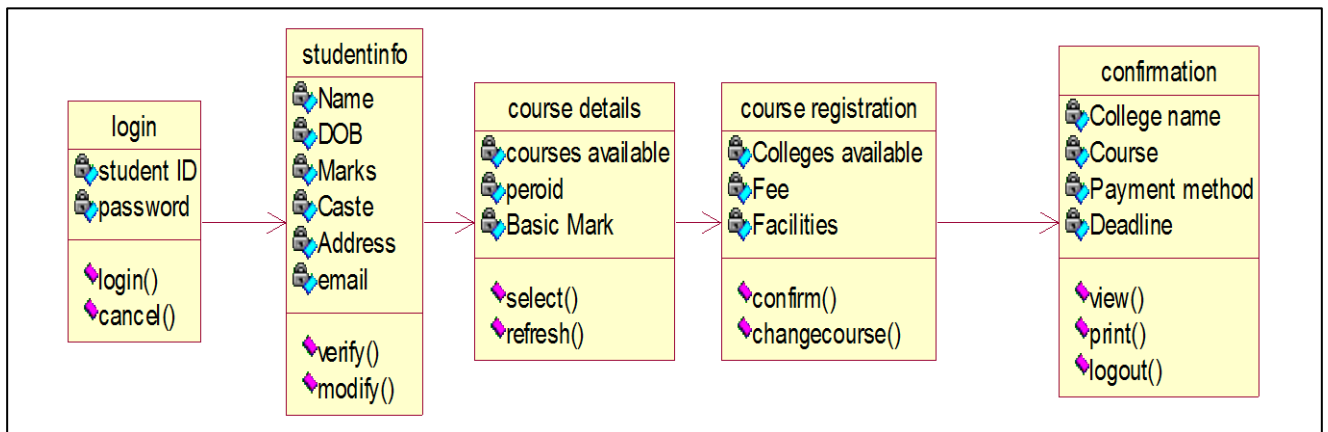
## 3.UML DIAGRAMS:

### 3.1.UseCase Diagram:

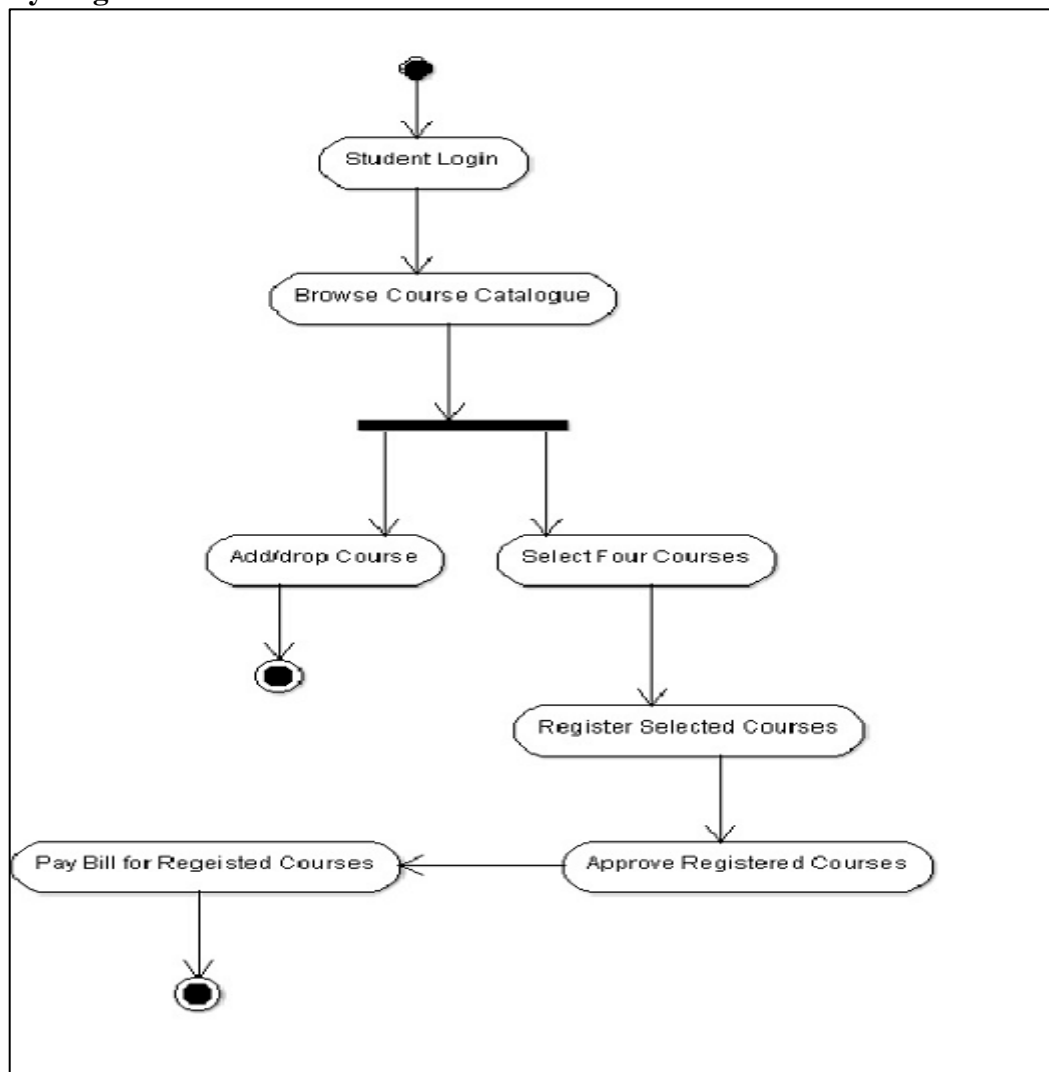




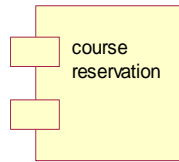
3.2. Class Diagram:



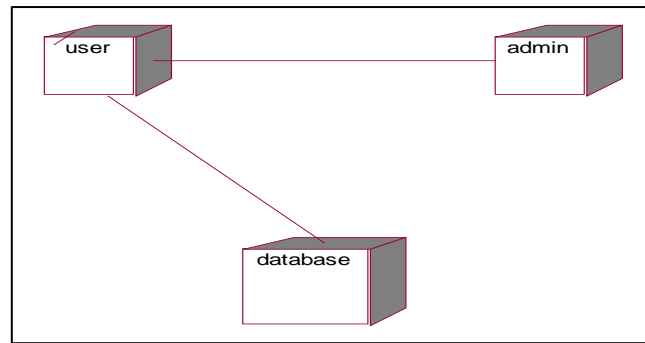
3.3. Activity Diagram:



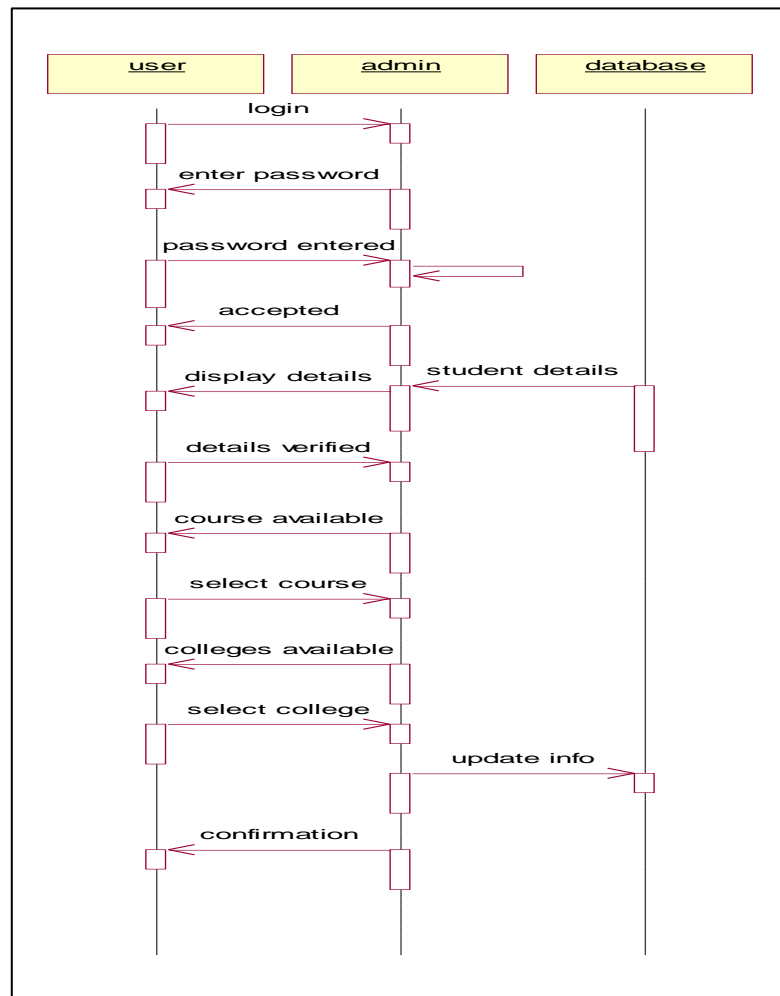
3.4.Component Diagram:



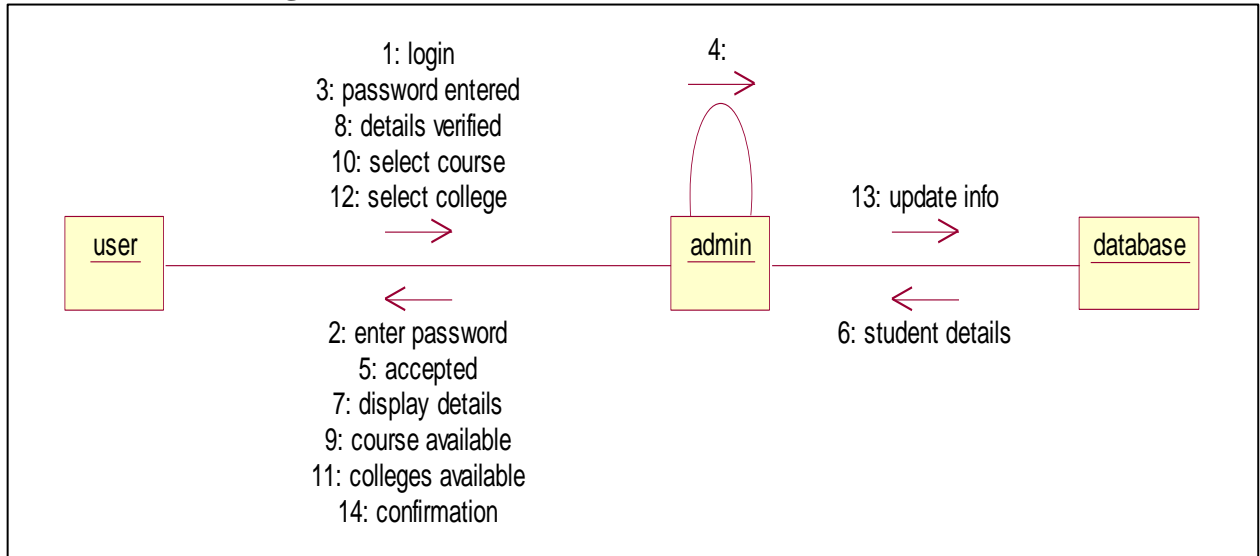
3.5.Deployment Diagram:



3.6.Sequence Diagram:



**3.7.Collaboration Diagram:**



**4. DATABASE DESIGN**

Database name: student

Table name : studentdetails

Fields	Data type
NAME	text
Dob	text
MARK	integer
Caste	text
ADDRESS	text
Email	text
Gender	text

Table name : coursedetails

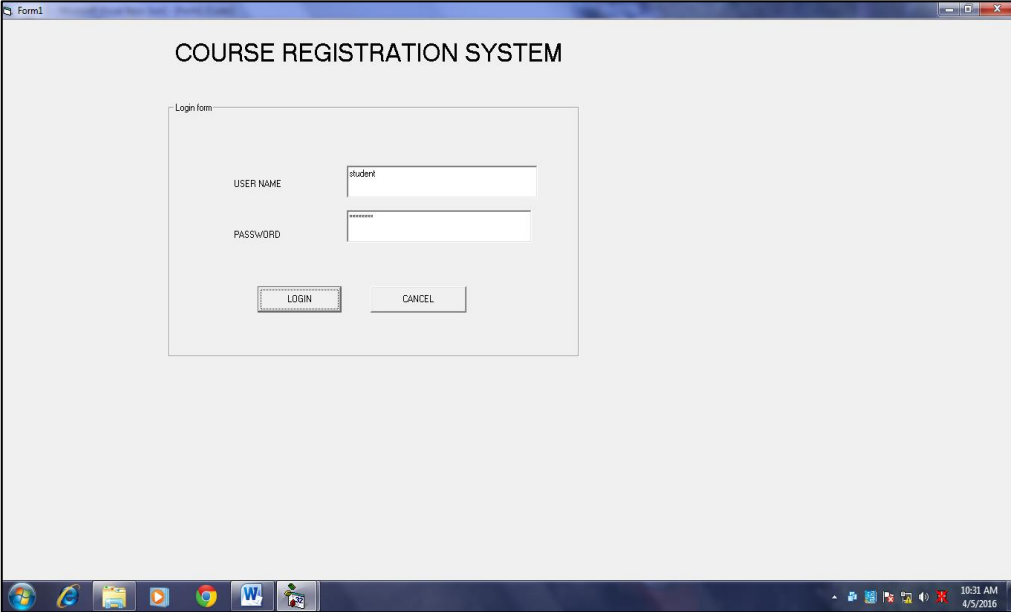
Fields	Data type
Courseavailable	text
Timeperiod	text
basicmark	integer

Table name : conformation

Fields	Data type
Availablecolleges	text
Paymentmethod	text

## 5. IMPLEMENTATION:

Form1(student login)



### Coding:

```
Private Sub Command1_Click()
```

```
If Text1.Text = "student" and Text2.Text = "password" Then  
form2.Show
```

```
else
```

```
msgBox ("incorrect username or password")
```

```
End If
```

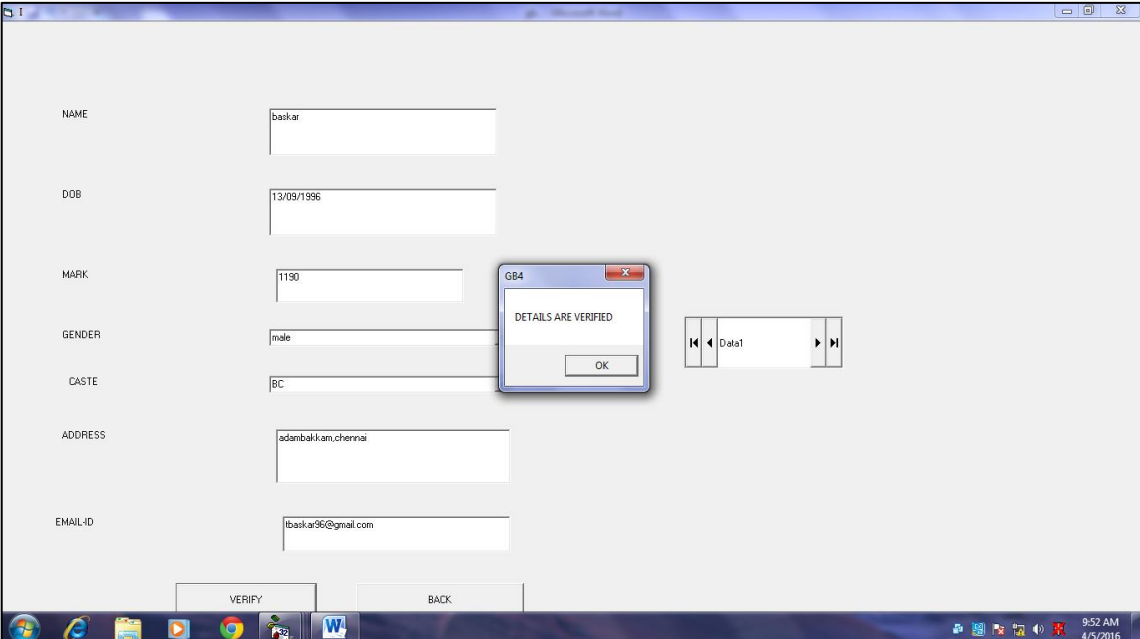
```
End Sub
```

```
Private Sub Command2_Click()
```

```
End
```

```
End Sub
```

Form2(Student details)



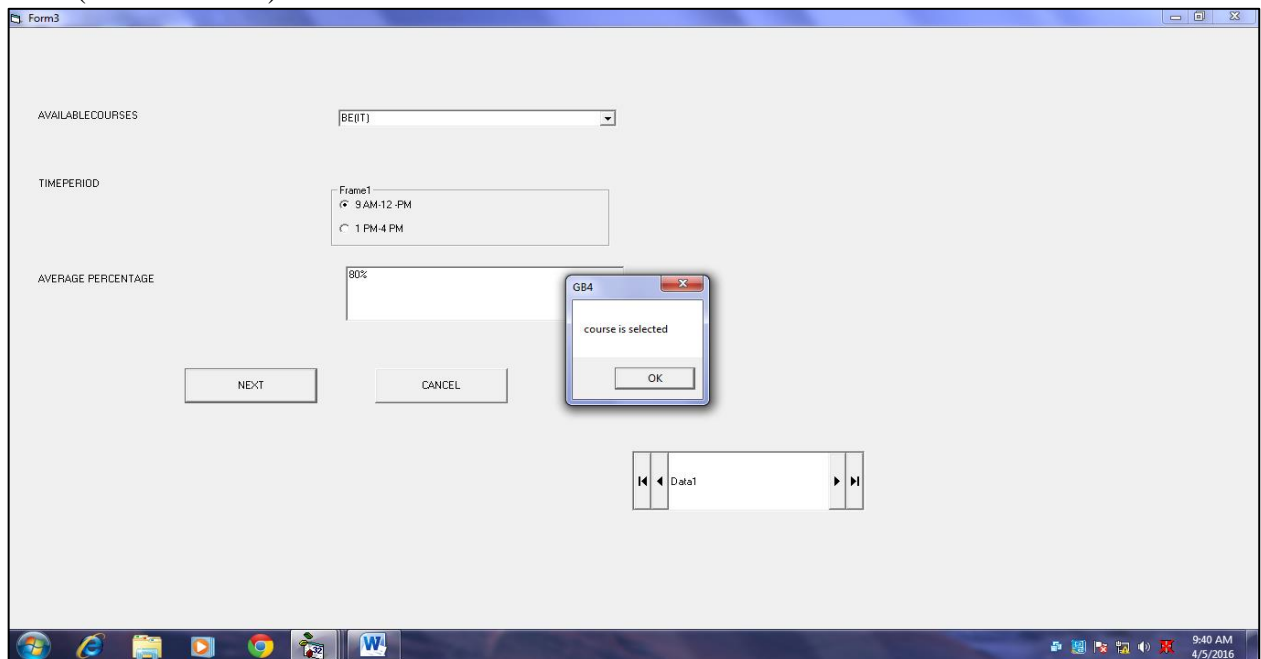
Coding:

```

Private Sub Command1_Click()
form2.Show
Data1.Recordset.AddNew
Data1.Recordset.Fields("NAME") = Text5.Text
Data1.Recordset.Fields("dob") = Text4.Text
Data1.Recordset.Fields("MARK") = Text3.Text
Data1.Recordset.Fields("caste") = Combo1.Text
Data1.Recordset.Fields("ADDRESS") = Text2.Text
Data1.Recordset.Fields("email") = Text1.Text
Data1.Recordset.Fields("gender") = Combo2.Text
MsgBox ("DETAILS ARE VERIFIED")
Form3.Show
Data1.Recordset.Update
End Sub
Private Sub Command2_Click()
Form1.Show
End Sub
Private Sub Form_Load()
Combo1.AddItem ("BC")
Combo1.AddItem ("SC")
Combo2.AddItem ("male")
Combo2.AddItem ("female")
Combo2.AddItem ("other")
End Sub

```

## Form3(Course details)

Coding:

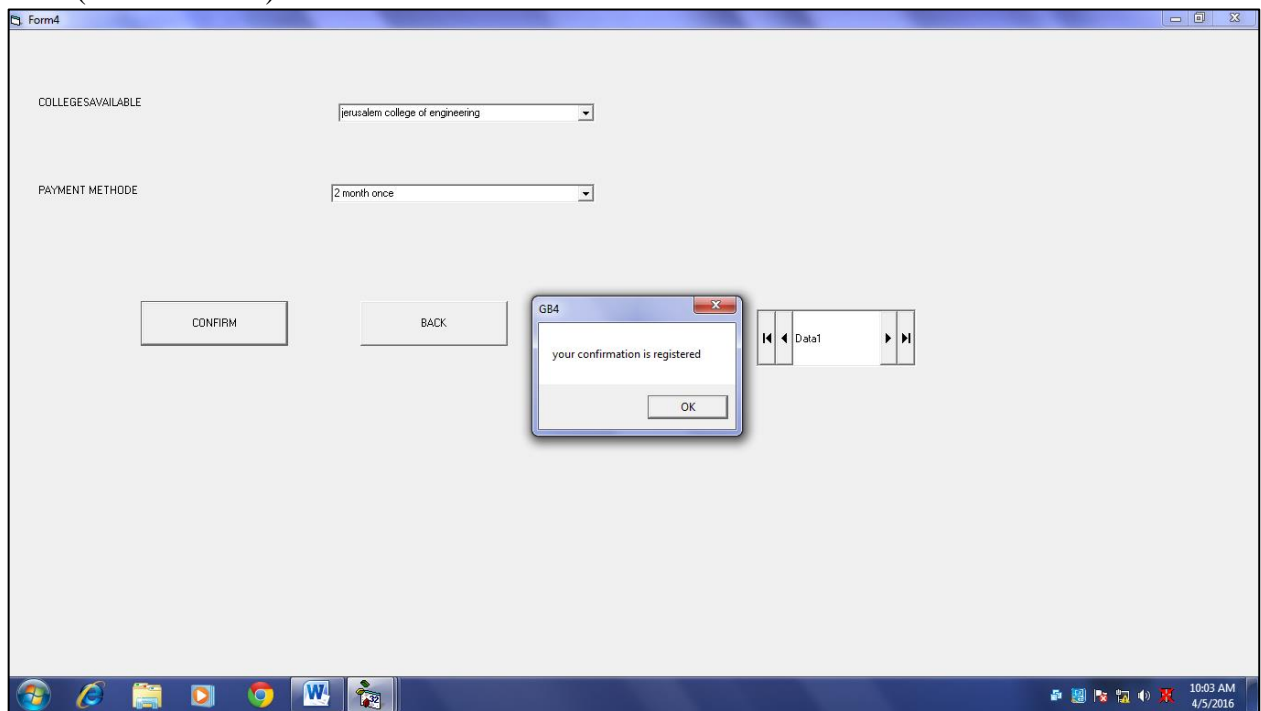
```

Private Sub Command1_Click()
Form3.Show
Data1.Recordset.AddNew
Data1.Recordset.Fields("courseavailable") = Combo1.Text
If Option1.Value = True Then
Data1.Recordset.Fields("timeperiod") = "9am-12pm"
Else

```

```
Data1.Recordset.Fields("timeperiod") = "1 am-4pm"
End If
Data1.Recordset.Fields("basicmark") = Val(Text1.Text)
MsgBox ("course is selected")
Form4.Show
Data1.Recordset.Update
End Sub
```

```
Private Sub Command2_Click()
Form2.Show
End Sub
Private Sub Form_Load()
Combo1.AddItem ("BE(ECE)")
Combo1.AddItem ("BE(IT)")
Combo1.AddItem ("BE(CIVIL)")
Combo1.AddItem ("BE(E&I)")
End Sub
Form4(Conformation)
```



#### Coding:

```
Private Sub Command1_Click()
Form4.Show
Data1.Recordset.AddNew
Data1.Recordset.Fields("availablecolleges") = Combo1.Text
Data1.Recordset.Fields("paymentmethod") = Combo2.Text
MsgBox (" your confirmation is registered")
Form1.Show
End Sub
Private Sub Command2_Click()
Form3.Show
End Sub
Private Sub Form_Load()
Combo1.AddItem ("jerusalem college of engineering")
Combo1.AddItem ("prathushya engineering college")
```

```

Combo1.AddItem ("tagore engineering college")
Combo2.AddItem ("single payment")
Combo2.AddItem ("2 month once")
Combo2.AddItem ("3 month once")
End Sub

```

**6. TESTING:**

<b>Test case ID: Test_01</b>					
Test priority (Low/Medium/High):Medium					
Module name: login					
Test title :verify login with valid username and password					
Precondition: user has invalid username and password					
S.NO	TEST STEPS	EXPECTED RESULTS	ACTUAL RESULTS	STATUS	NOTES
1	Provide valid User name	User should Be able to login	The user is able to move to next Entry	Success	-
2	Provide valid password	User should be Able to Login	The user is able To login Successfully	Success	In case of wrong Password was given an error Message box was displayed
3	Click login	User should be able to navigate to next page after validation	User name and password is validated and next page is displayed	Success	Incase user gives wrong entry the sign in page remains active
4	Click signup	User should be able to navigate to next page where user enters his credentials	User navigates to the signup page where his user name and password is validated	success	-

**RESULT:** The Course registration system was designed and implemented successfully.

## 1.2 AIM :Student marks analyzing system

### 1.PROBLEM STATEMENT:

Student marks analyzing system has to be developed for analyzing obtained by the students who scored in Semester Examination The System should provide following functionalities

1. The System obtains following information's from the faculty generates report Roll No, Name, Department, Semester, Marks obtained in each subject.
2. The total for each student should be calculated and ranked based on total and pass in all the subject appeared.
3. The Final report should display rank, percentage, Class, Pass/Fail Status for each student.

### 2.OVERALL DESCRIPTIONS:

#### 2.1MODULES:

- 1.Login
- 2.student\_mark
- 3.View report

#### 2.2MODULE DELIVARABLES:

##### 1. Login

Basic Flow: This use case starts when the Faculty wishes to Login to the Student Marks Analyzing System.

1. The System requests that the Faculty enter his/her name and password
2. The Faculty enters his/her name and password
3. The System validates the entered name and password and logs the Faculty into the System

Alternative Flows: Invalid Name/Password

If, in the Basic flow, the Faculty enters an invalid name and/or password, the system displays an error message. The Faculty chooses to either return to the beginning of the Basic flow or cancel the login, at which point the use case ends.

Pre-Conditions: None

Post-Conditions: If the use case was successful, the Faculty is now logged into the system. If not, the system State is unchanged.

##### 2. Student\_Mark

Basic flow: The Faculty uses this usecase to enter marks for each student. The faculty enters the following details namely Roll No, Student Name, Department, Marks for each student.

Alternative Flows: If faculty not entered any details or invalid marks then gives error Message

Pre-Conditions: The Faculty must logged into the system

Post-Conditions: If this Use case was successful, Student Mark Analysis Report will be generated for the Student.



### 3.View Report

Basic flow: The Actor uses this usecase view the Report .The report contains the following details Namely Roll No, Student Name, Marks in each subject, total, class, Pass/Fail Status, No of subjects failed, Rank.

Apart from this there is a separate report Overall Pass percentage of class, No of students cleared in First class, Overall Top 3 persons of the class.

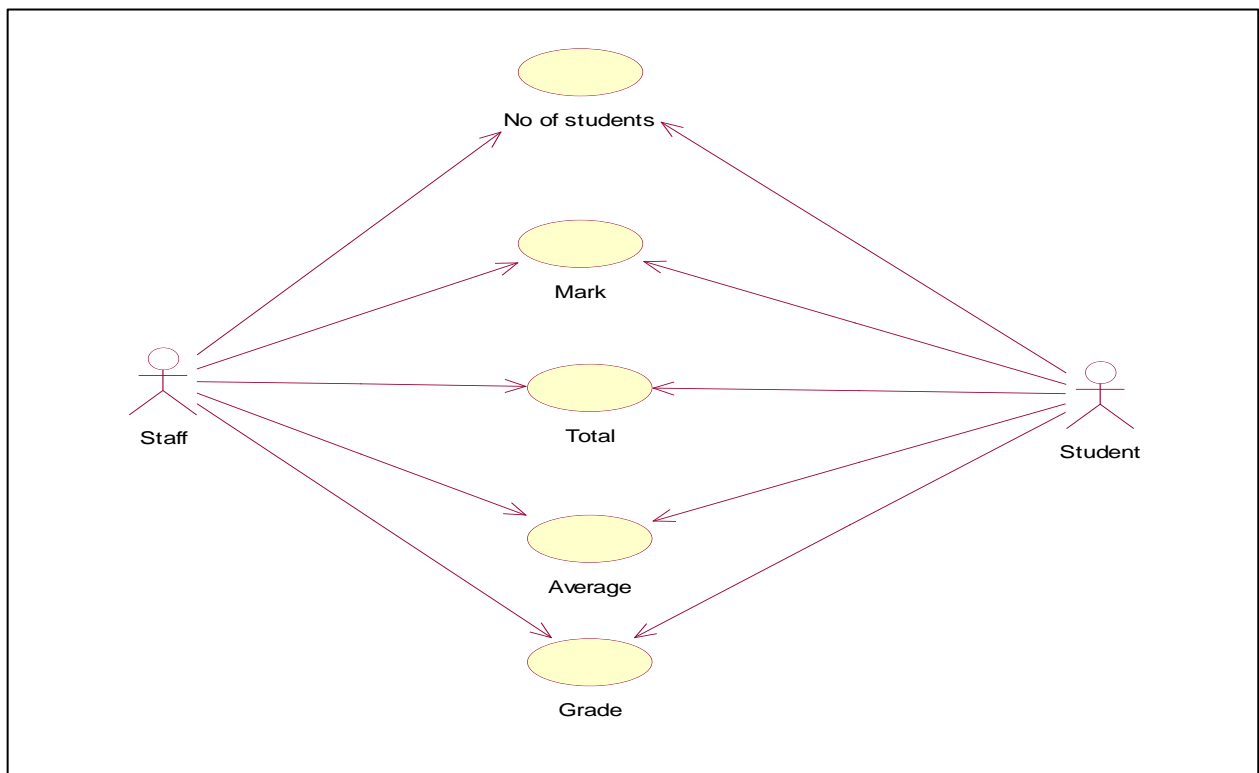
Alternative Flow: If the Marks is not entered for all the students the use case will ask the faculty to the enter the marks.

Pre-Conditions: The Faculty must entered marks for all the students in a class.

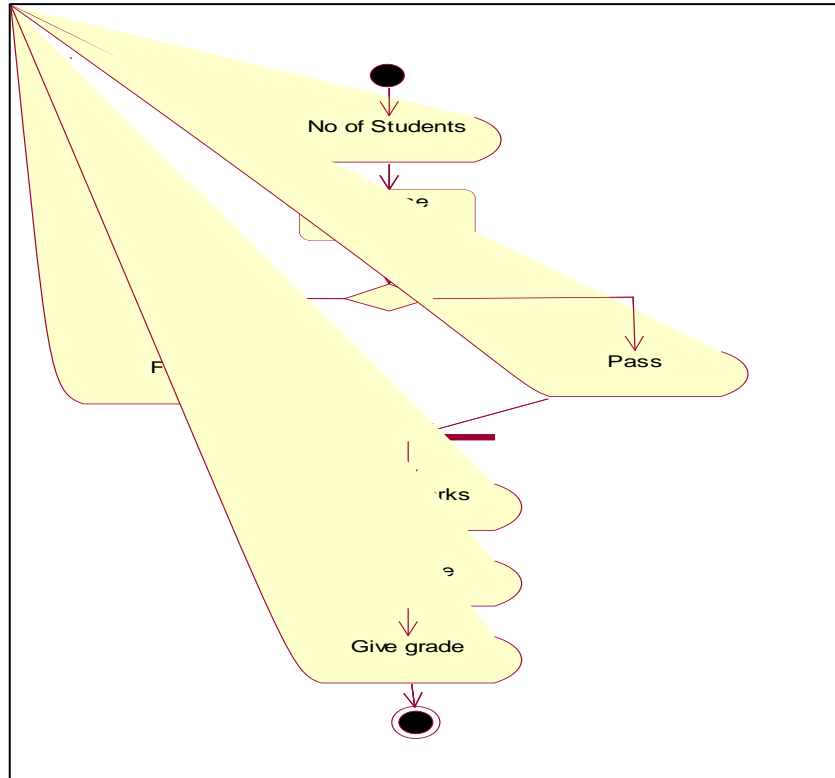
Post-Conditions: None

### 3.UML DIAGRAMS:

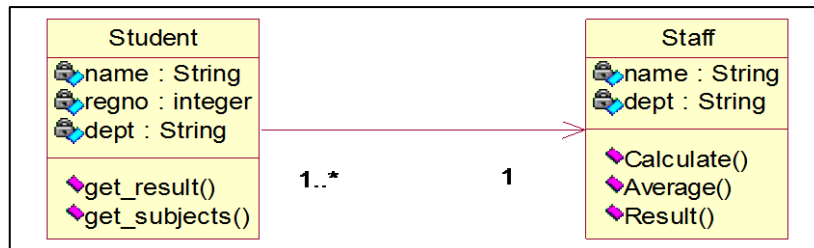
#### 3.1.UseCase Diagram:



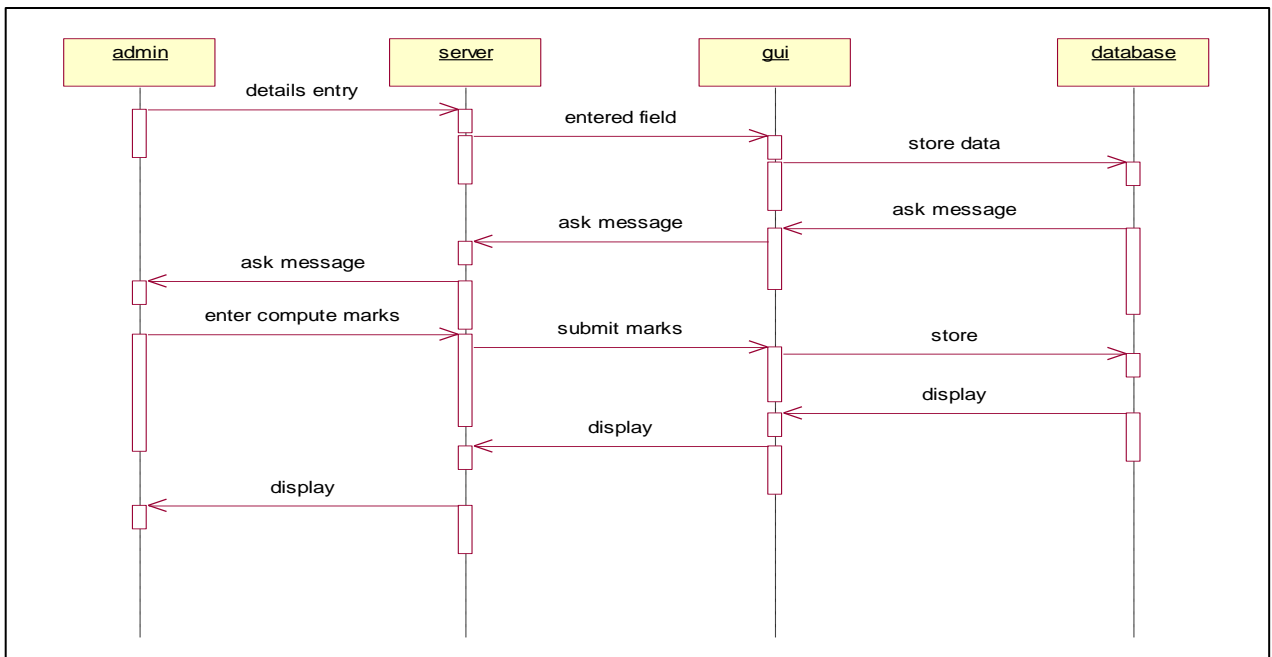
3.2 Activity Diagram



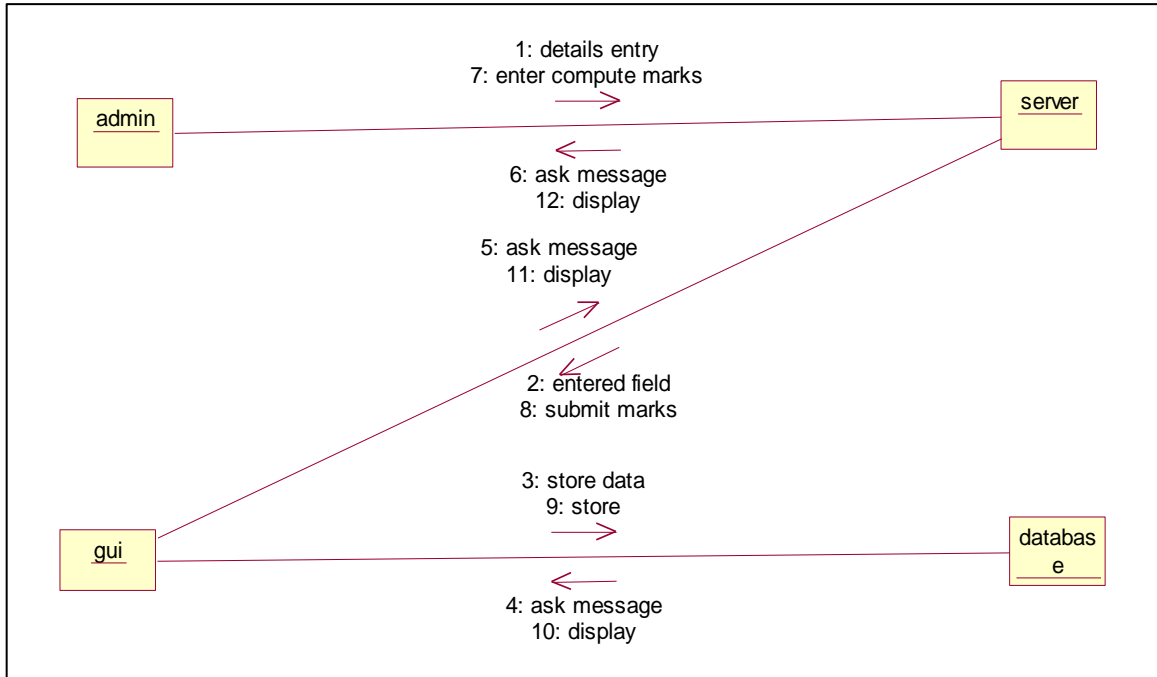
3.3 Class Diagram



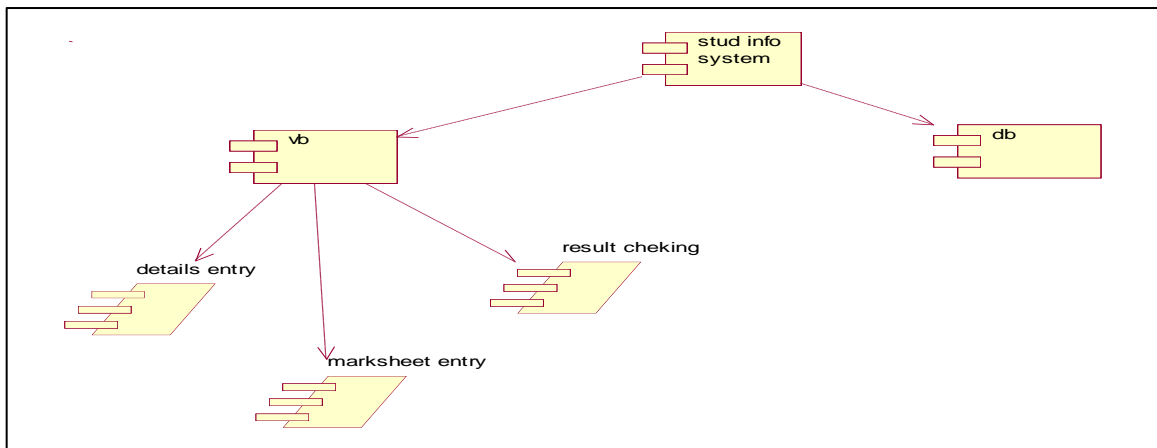
3.4 Sequential Diagram



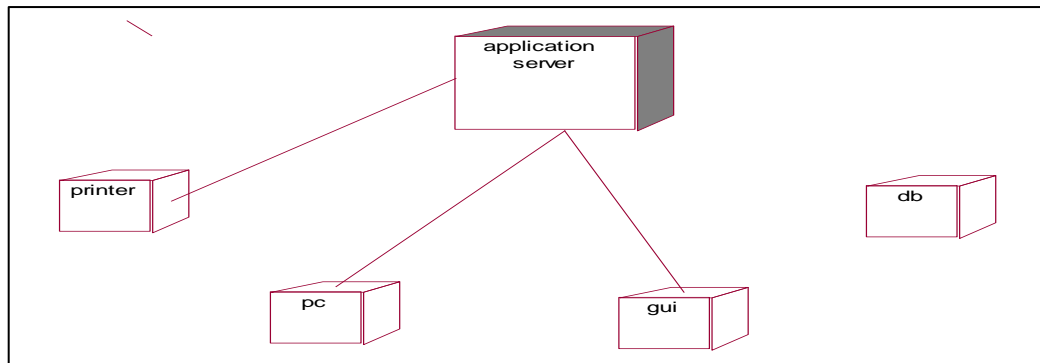
3.5 Collaboration diagram



3.6 Component Diagram of student info System



3.7 Deployment Diagram of Student info System



#### 4. DATABASE DESIGN

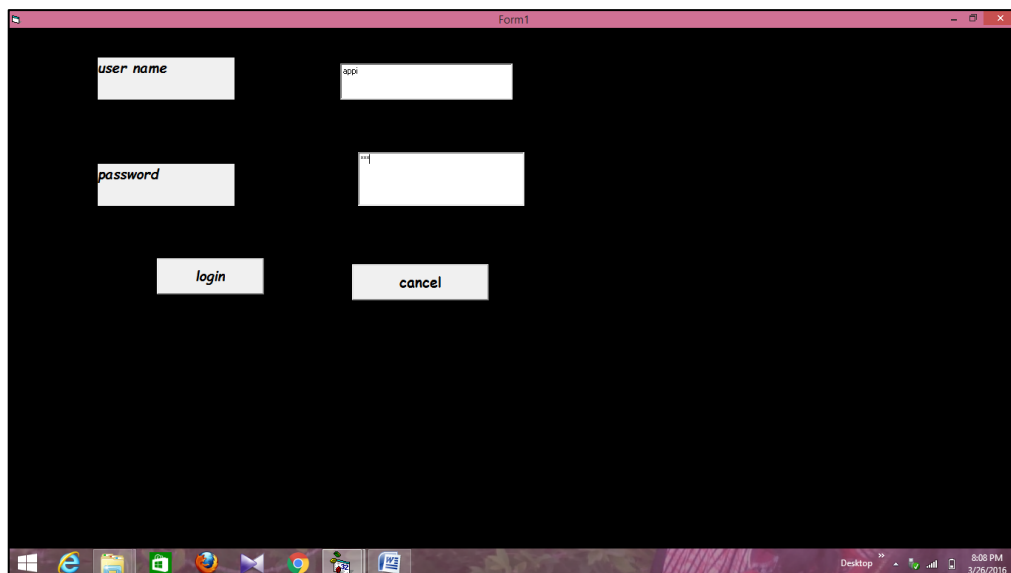
Database name: student

Table name:marks

Fields	Data type
student name	Text
register number	Integer
Pqt	Integer
Daa	Integer
Mup	Integer
Se	Integer
Os	Integer
Total	Integer
Percentage	Double
Result	Text

#### 5. IMPLEMENTATION:

##### FORM1 (Login form)



##### Coding:

```
Private Sub Command1_Click()  
if text1.text="student" and text2.text="itdept" then  
form2.Show  
Else  
msgbox(("Please enter correct user name and password"))  
end if  
End Sub
```

```
Private Sub Command2_Click()  
End  
End Sub
```

## FORM2 Entry form

Coding:

```

Private Sub Command1_Click()
Dim total, per, result
Data1.Recordset.AddNew
Data1.Recordset.Fields("sname") = Text1.Text
Data1.Recordset.Fields("regno") = Text2.Text
Data1.Recordset.Fields("m1") = Text3.Text
Data1.Recordset.Fields("m2") = Text4.Text
Data1.Recordset.Fields("m3") = Text5.Text
Data1.Recordset.Fields("m4") = Text6.Text
Data1.Recordset.Fields("m5") = Text7.Text
total = Text3.Text + Text4.Text + Text5.Text + Text6.Text + Text7.Text
Data1.Recordset.Fields("total") = total
per = (total / 5) * 100
Data1.Recordset.Fields("per") = per
If (Text3.Text > 49 And Text4.Text > 49 And Text5.Text > 49 And Text6.Text > 49 And
Text7.Text > 49) Then result = "pass"
Data1.Recordset.Fields("result") = result
Data1.Recordset.Update
MsgBox ("record saved")

```

```
Form3.Show
```

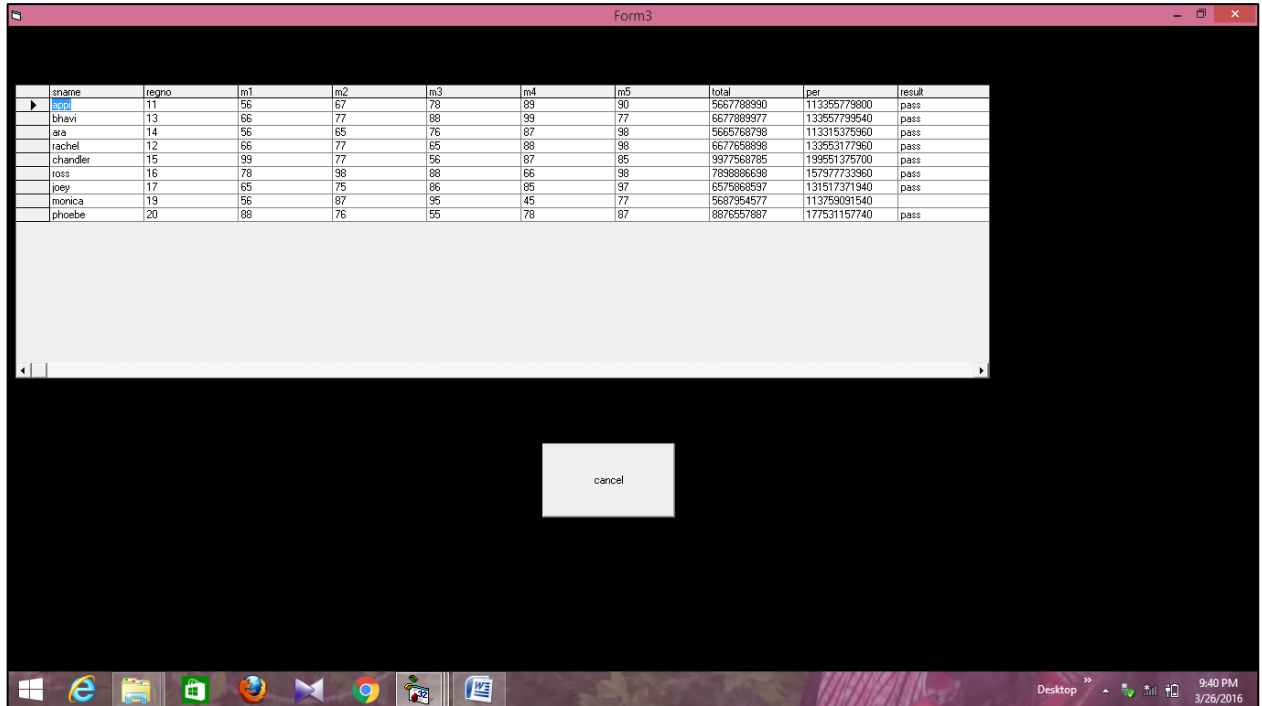
```
End Sub
```

```
Private Sub Command2_Click()
```

```
End
```

```
End Sub
```

## FORM3 (Display Students)



The screenshot shows a Windows desktop environment. A window titled 'Form3' is open, displaying a table of student data. The table has columns for 'sname', 'regno', 'm1', 'm2', 'm3', 'm4', 'm5', 'total', 'per', and 'result'. The data is as follows:

sname	regno	m1	m2	m3	m4	m5	total	per	result
ecpi	11	56	67	78	89	90	5667788990	113355779800	pass
bhavi	13	66	77	88	99	77	6677889977	133557799540	pass
ala	14	56	65	76	87	98	5665766798	113313375960	pass
rachel	12	66	77	88	99	88	6677889988	133653177960	pass
chandler	15	99	77	56	87	85	9977568785	199551375700	pass
ross	16	78	98	88	66	98	7898886698	157977733960	pass
roey	17	65	75	86	85	97	6575868597	131517371940	pass
monica	19	56	87	95	45	77	5687954577	113759091540	pass
phoebe	20	88	76	55	78	87	8876557887	177531157740	pass

Below the table, there is a large empty rectangular area. At the bottom center of the form, there is a button labeled 'cancel'. The Windows taskbar is visible at the bottom of the screen, showing the Start button, several application icons, and the system tray with the date and time (9:40 PM, 3/26/2016).

Coding:

```
Private Sub Command1_Click()
```

```
End
```

```
End Sub
```

**6.TESTING:**

Test case ID: Test_01					
Test priority (Low/Medium/High):Medium					
Module name: login					
Test title :verify login with valid username and password					
Precondition: user has invalid username and password					
S.NO	TEST STEPS	EXPECTED RESULTS	ACTUAL RESULTS	STATUS	NOTES
1	Provide valid User name	User should Be able to login	The user is able to move to next Entry	Success	-
2	Provide valid password	User should be Able to Login	The user is able To login Successfully	Success	Incase of wrong Password was given an error Message box was Displayed
3	Click login	User should be able to navigate to next page after validation	User name and password is validated and next page is displayed	Success	Incase user gives wrong entry the sign in page remains active
4	Click signup	User should be able to navigate to next page where user enters his credentials	User navigates to the signup page where his user name and password is validated	success	-

**RESULT:**The Student marks analyzing system was designed and implemented successfully.

**PROGRAM 2 : Health Care**

**AIM : Health Care ( Expert system to prescribe medicines for given symptoms, Remote Diagnostics, Patient/Hospital Management System)**

**1.PROBLEM STATEMENT:**

The project is mainly focused on medical field in expert system where the person or patient login to the system and must select what are the symptoms for him that field information will be given to expert system. Expert system will diagnose what type of disease using medical database. The medical database will get all the information about medicine from pharmacy module and generate the prescription for the right symptoms and it gives to expert system. The medical database gives types of disease information to expert system.

**2.OVERALL DESCRIPTION:****2.1 MODULES:**

Symptoms

Expert System

Medical DB

Prescription

Pharmacy

**2.2 MODULE DELIVERABLES:****SYMPTOMS:**

Basic Flow:

Patient must enter the symptoms that occurs for him. The symptoms must be exact so that it can be verified.

Alternate Flow:

If there are 4 symptoms then the patient is having big problem disease so,exit. If there are more than 4 then it is a serious problem

Precondition:

All the symptoms related to the disease are entered. Patient must enter the symptoms that occurs correctly.

Postcondition:

If the symptoms are corresponding to particular disease they are entered in expert system.

**PRESCRIPTION:**

Basic Flow:

The medicine is prescribed based on the symptoms. Medical prescription is given by expert system.

Alternate Flow:

If medicine is not available then exit. If there is no medicine then it comes under stock unavailable



condition so update information in medical DB.

Precondition:

If medicine is only available in pharmacy it will generate prescription. If medicine is not available then store the details in expert system.

Postcondition:

If medicine are prescribed it will give to patient. The patient then gets the prescribed medicine in the shop.

### **PHARMACY:**

#### **Basic Flow:**

Based on disease it will give medicine. Pharmacy gets the disease of prescribed medicine from the expert system.

Alternate Flow:

If medicine are not available it will pass the information as no medicine available

Precondition:

If prescribed medicine is available in pharmacy expert system must generate the required code.

Postcondition:

If prescribed medicine is available in pharmacy then expert system must load the data that is available.

### **EXPERT SYSTEM:**

Basic Flow:

It will get symptoms and information from medical database to diagnose disease

Alternate Flow:

If any of the condition fails then system will exit.

Precondition:

Medical database and symptoms of patient should be present

Postcondition:

Prescribe the medicine to the patient from expert system to the patient.

### **MODEL DATABASE:**

Basic Flow:

Medical database contains the backup and additional details which is not there in expert system.

Alternate Flow:

If medical database from expert system is not validated in medical database then first we have to update them

Precondition:

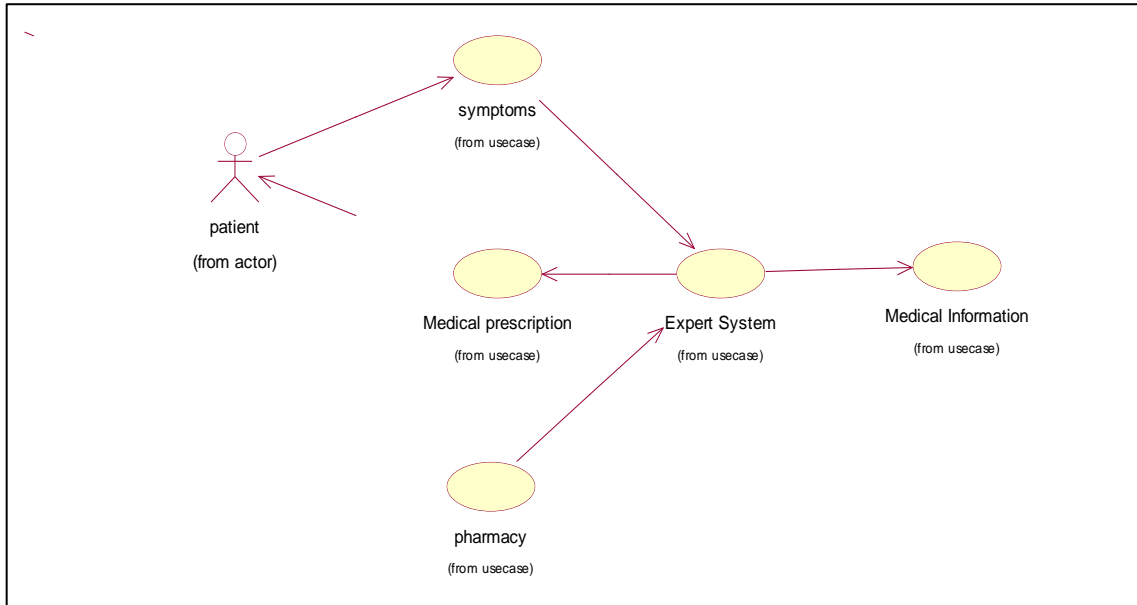
Medical database must contain some basic medicine stored in the memory of the system or computer

Postcondition:

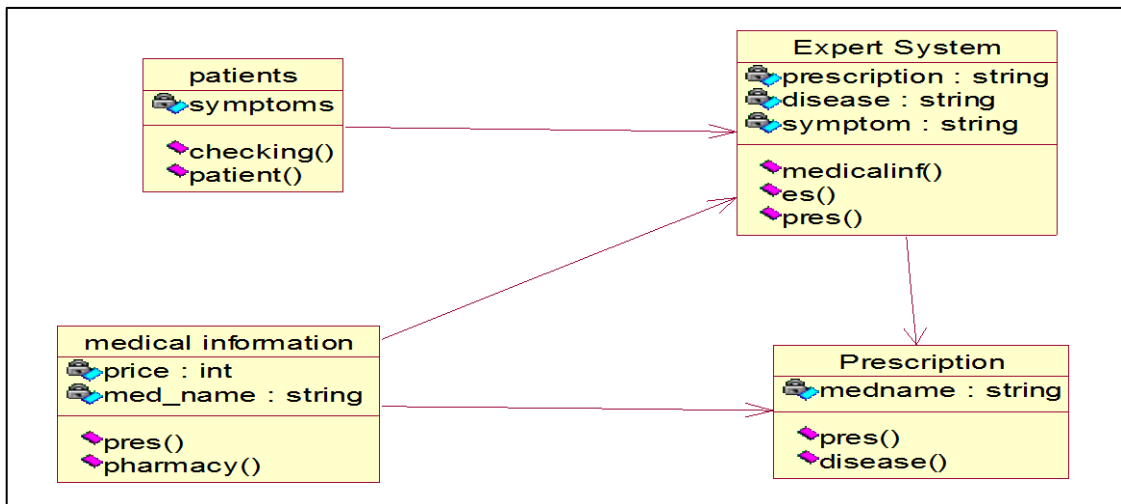
If the details which is not present in expert system contained in database then we can extract information.

**3.UML DIAGRAM:**

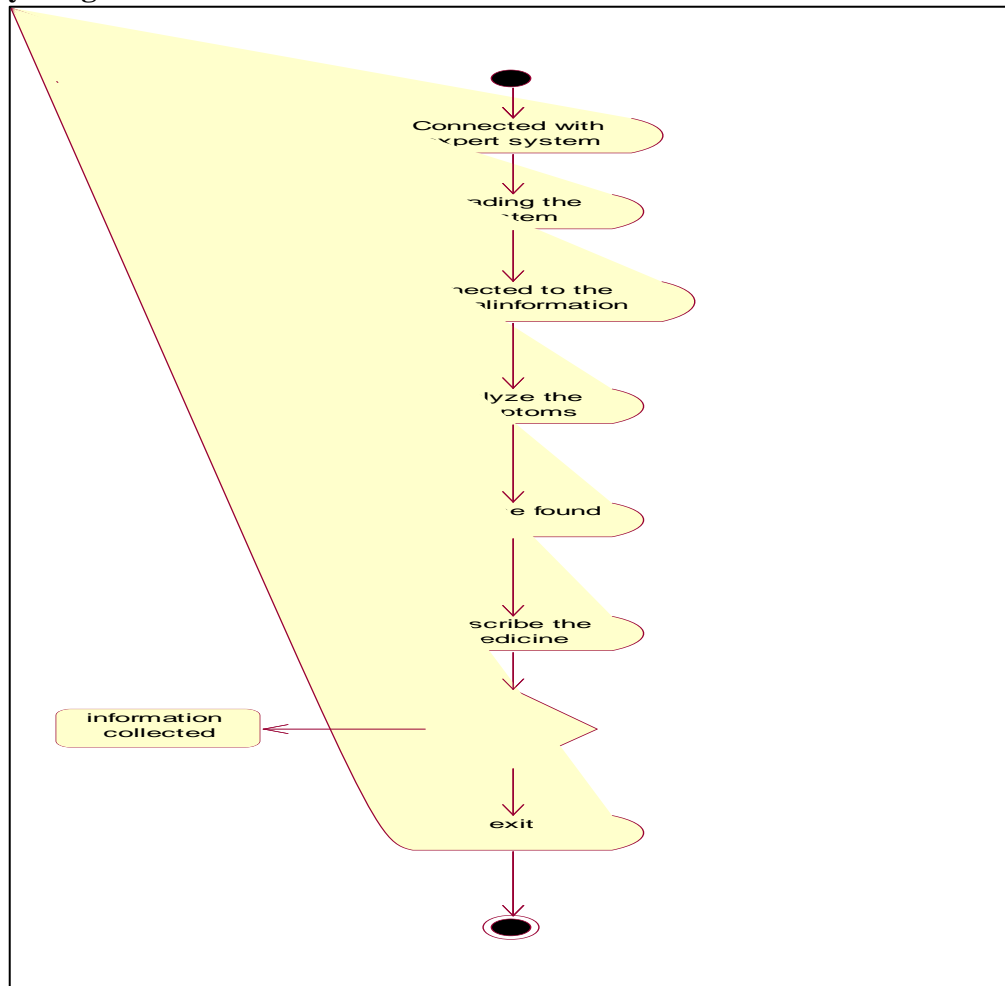
**3.1 Usecasediagram:**



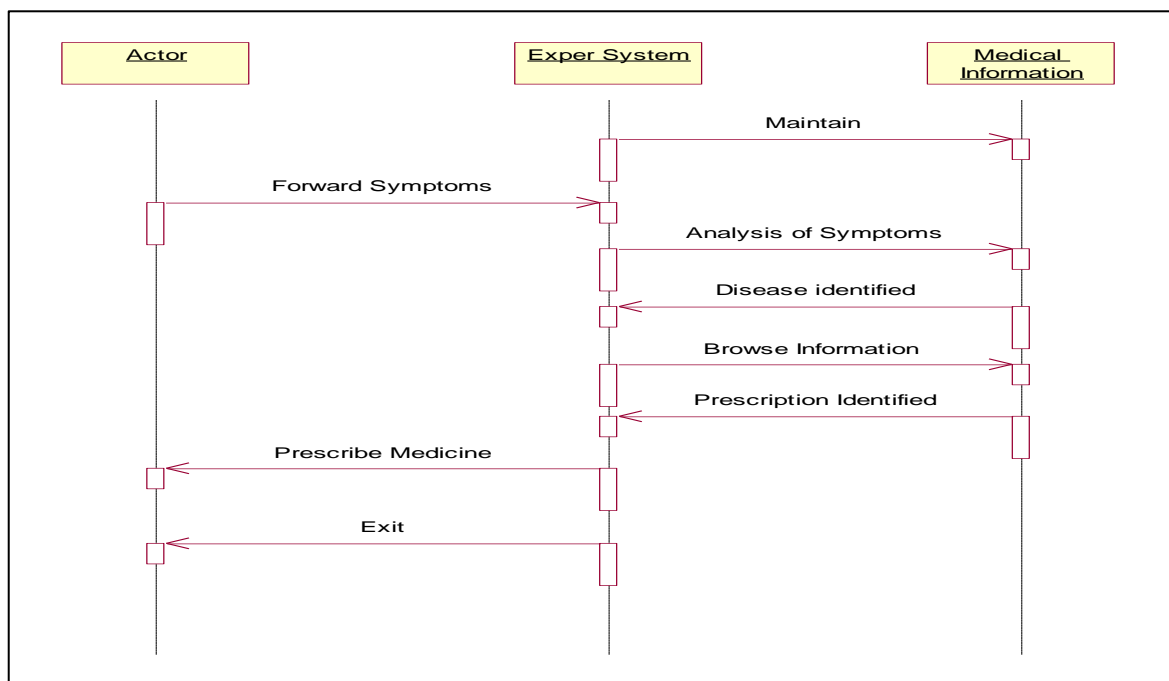
**3.2 Class Diagram:**



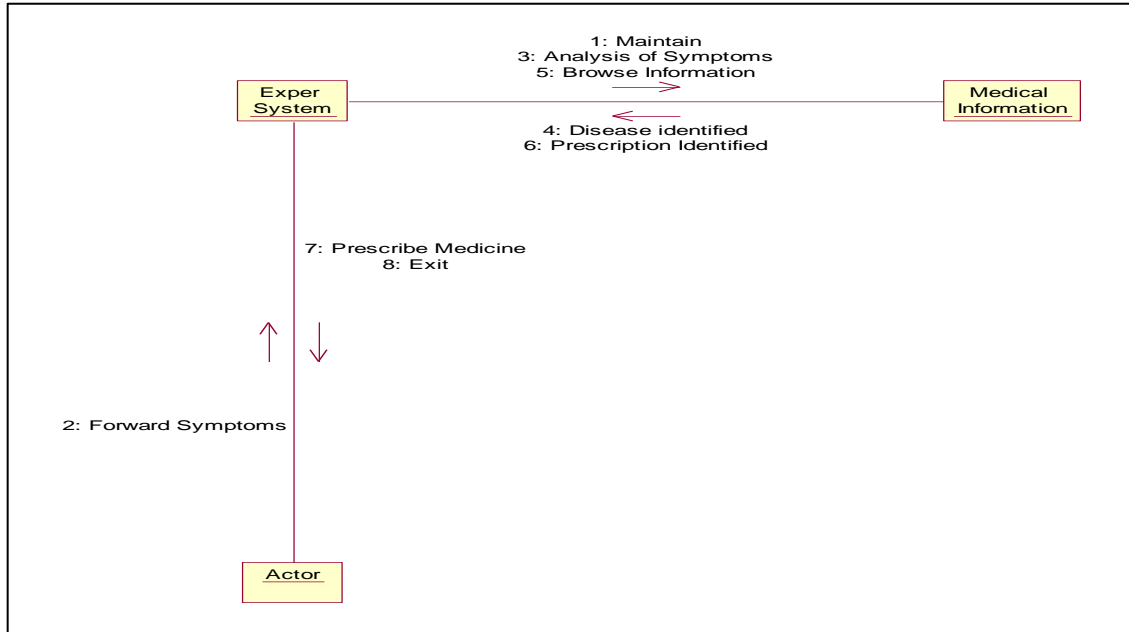
3.3 Activity Diagram:



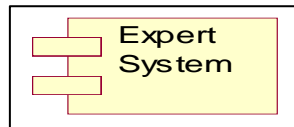
3.4 Sequence Diagram:



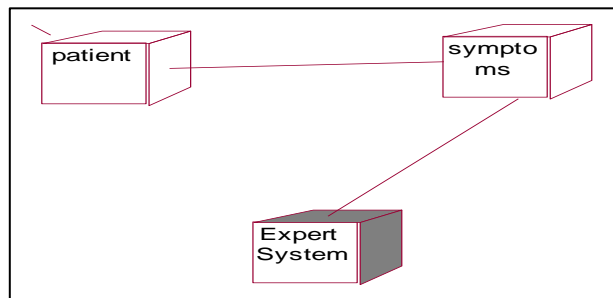
**3.5 Collaboration Diagram:**



**3.6 Component Diagram:**



**3.7 Deployment Diagram**



**4. DATABASE DESIGN**

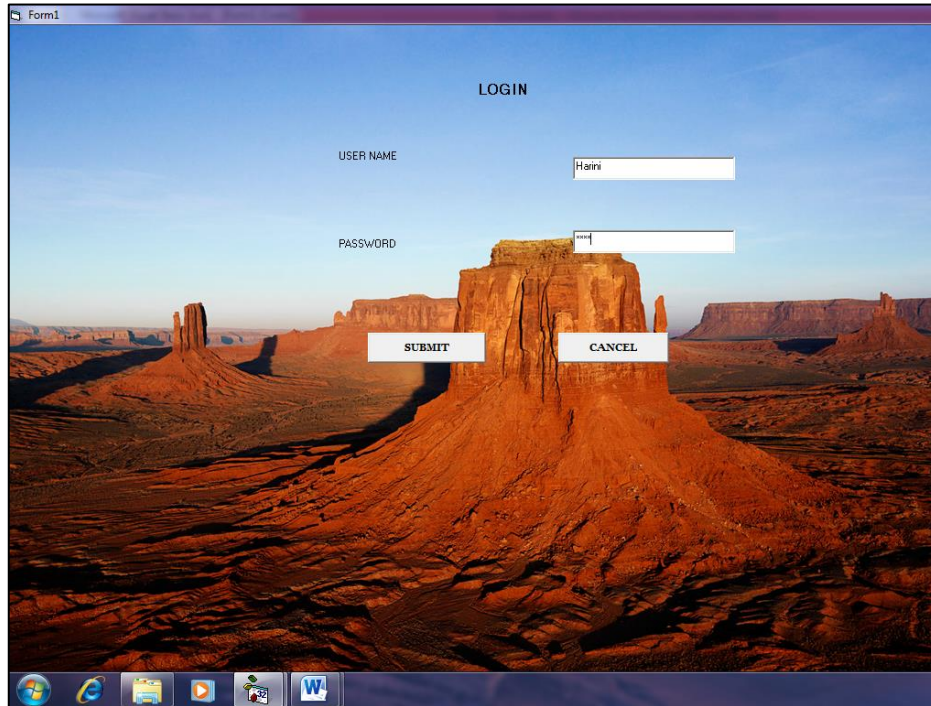
Database Name: Medical

Table Name : Patient

Fields	Data type
pid	Integer
pname	Text
age	Integer
gender	Text
symptoms	Text
drug	Text

## 5. IMPLEMENTATION:

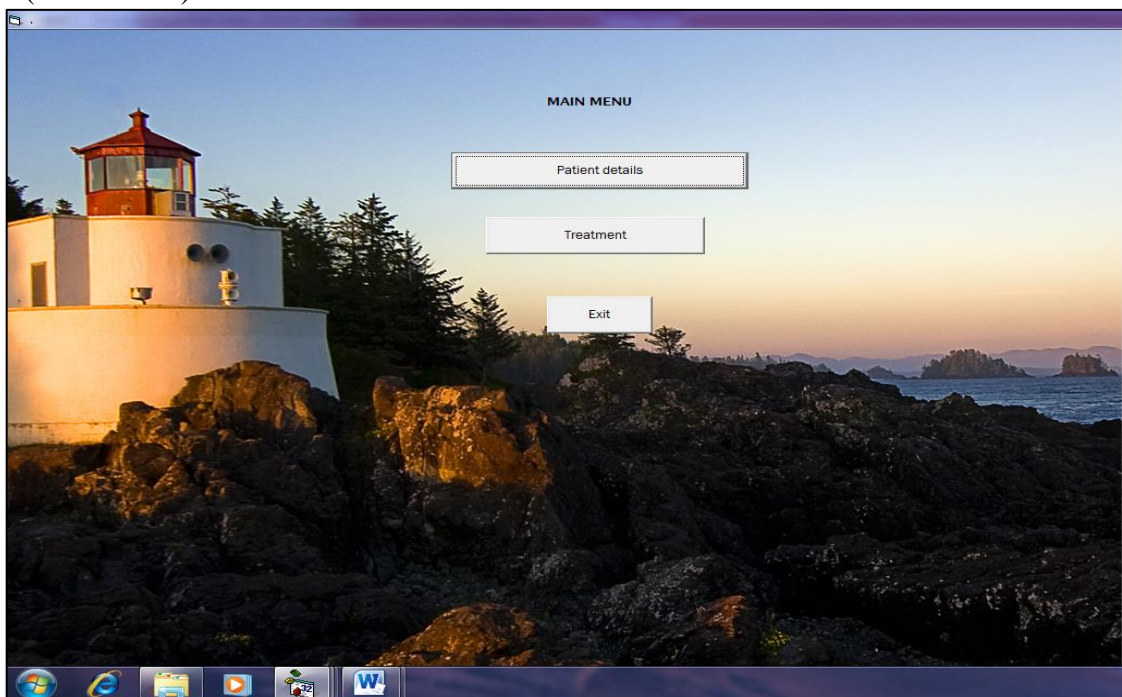
### Form1 (Login form)



#### Coding:

```
Private Sub Command1_Click()  
Form2.Show  
End Sub  
Private Sub Command2_Click()  
End  
End Sub
```

### Form 2 (Main menu)



Coding:

```
Private Sub Command1_Click()
Form3.Show
End Sub
```

```
Private Sub Command2_Click()
Form4.Show
End Sub
```

```
Private Sub Command3_Click()
Form1.Show
End Sub
```

FORM 3 (Patient data entry)

Coding:

```
Private Sub Command1_Click()
Data1.Recordset.AddNew
Data1.Recordset.Fields("pID") = Text1.Text
Data1.Recordset.Fields("pName") = Text2.Text
Data1.Recordset.Fields("age") = Text3.Text
If Option1.Value = True Then
Data1.Recordset.Fields("Gender") = "Male"
Else
Data1.Recordset.Fields("Gender") = "Female"
End If
Data1.Recordset.Fields("Symptoms") = Combo2.Text
Data1.Recordset.Update
MsgBox ("Data Added")
End Sub
```

```
Private Sub Command2_Click()
Form2.Show
End Sub
```



```

Private Sub Form_Load()
Combo2.AddItem ("Fever")
Combo2.AddItem ("Cough")
Combo2.AddItem ("Cold")
End Sub

```

#### FORM 4 (Prescription form)

#### Coding:

```

Private Sub Command1_Click()
Data1.Recordset.Edit
Data1.Recordset.Fields("Drug") = drug.Text
Data1.Recordset.Update
MsgBox ("Data Added")
End Sub

```

```

Private Sub Command2_Click()
Form2.Show
End Sub

```

```

Private Sub Command3_Click()
Data1.Recordset.MoveFirst
Do While Not Data1.Recordset.EOF
If Data1.Recordset.Fields("pid") = pid.Text Then
pid.Text = Data1.Recordset.Fields("pid")
pname.Text = Data1.Recordset.Fields("pname")
age.Text = Data1.Recordset.Fields("age")
gender.Text = Data1.Recordset.Fields("Gender")
symptoms.Text = Data1.Recordset.Fields("Symptoms")
GoTo out
End If
Data1.Recordset.MoveNext
Loop
MsgBox ("not found")

```

out:

End Sub

### 6. TESTING:

Test case ID: Test_01					
Test priority (Low/Medium/High):Medium					
Module name: login					
Test title :verify login with valid username and password					
Precondition: user has invalid username and password					
S.NO	TEST STEPS	EXPECTED RESULTS	ACTUAL RESULTS	STATUS	NOTES
1	Provide valid User name	User should Be able to login	The user is able to move to next Entry	Success	-
2	Provide valid password	User should be Able to Login	The user is able To login Successfully	Success	Incase of wrong Password was given an error Message box was Displayed
3	Click login	User should be able to navigate to next page after validation	User name and password is validated and next page is displayed	Success	Incase user gives wrong entry the sign in page remains active
4	Click signup	User should be able to navigate to next page where user enters his credentials	User navigates to the signup page where his user name and password is validated	success	-

**RESULT:**The Expert system to prescribe the medicines for the given symptoms was designed and implemented successfully.



**PROGRAM 3: Finance (Banking:ATM/NetBanking, UPI:PayTM/PhonePay,Stocks:Zerodha)****AIM : Banking:ATM/NetBanking****1.PROBLEM STATEMENT:**

This project on atm is based on the processing of debit cards in atm. There are two main sections and the administrator section. In the withdraw module the requested amount is deducted from the user's account if sufficient balance is available. If the balance in the user's account is insufficient the service is denied. In the deposit module the deposited amount is added to the user's balance. In the balance enquiry module the amount that is present in the user's account is displayed. The user must enter his pin and account number to perform any of the transactions.

**2.OVERALL DESCRIPTION:****2.1 MODULES:**

Login

Deposit

Withdraw

Changing pin number

Balance enquiry

**2.1MODULE DELIVERABLES:****2.1.1 LOGIN:****Basic Flow :**

User enters correct pin and logs into the machine

**Alternative Flow :**

User enters wrong pin and the card gets rejected

**Pre-condition :**

User must know the correct pin number.

**2.1.2DEPOSIT AMOUNT:****Basic Flow:**

User enters the amount and feeds cash to machine.

**Alternative Flow:**

User cancels transaction.

**Pre condition:**

User should know the amount to be deposited.

**Post condition:**

User balance gets updated in the account.

### 2.1.3 WITHDRAWAL:

**Basic Flow:**

User enters amount and gets cash from machine.

**Alternative Flow:**

Not enough cash in account and so the card gets rejected.

**Pre condition:**

User knows the amount to withdraw.

**Post condition:**

Balance is updated in account.

### 2.1.4 BALANCE ENQUIRY:

**Basic Flow:**

User enquires about balance and the balance is displayed.

**Alternative Flow:**

User cancels transaction.

### 2.1.5 DELETE ACCOUNT:

**Basic Flow:**

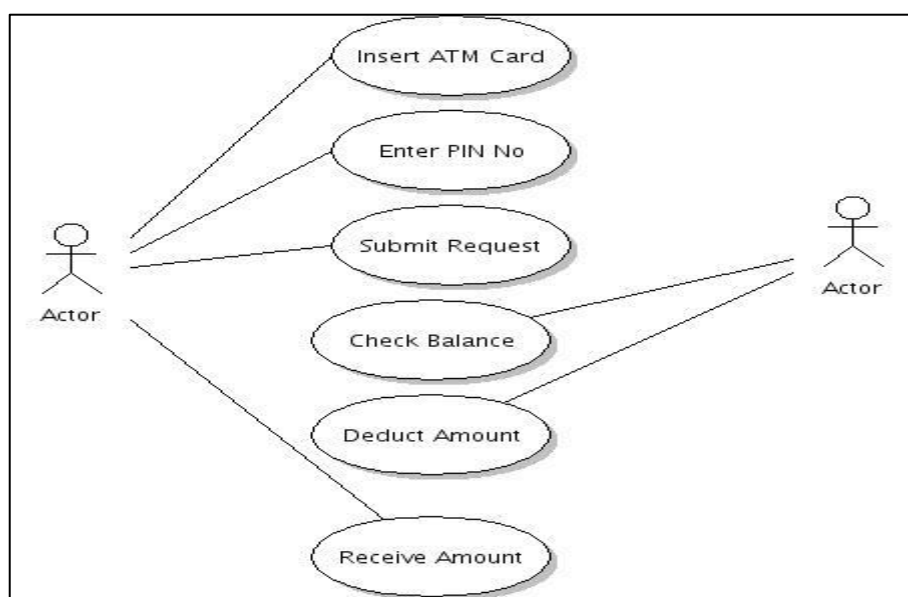
User confirms deletion and account gets deleted.

**Alternative Flow:**

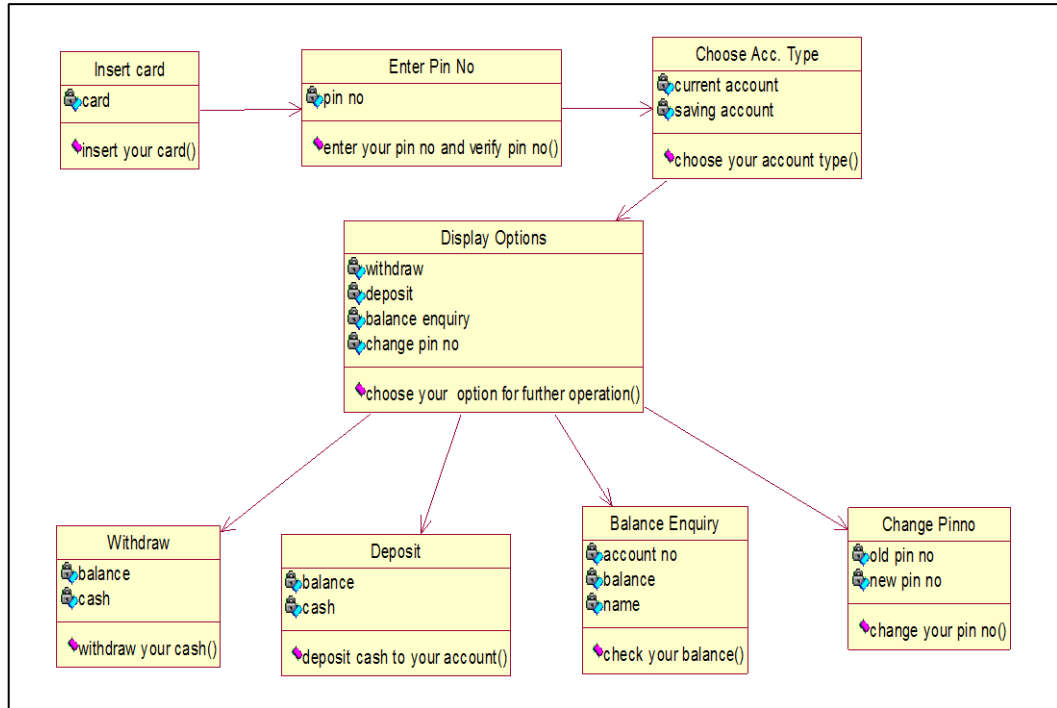
User doesn't confirm and resumes transaction.

## 3. DESIGN

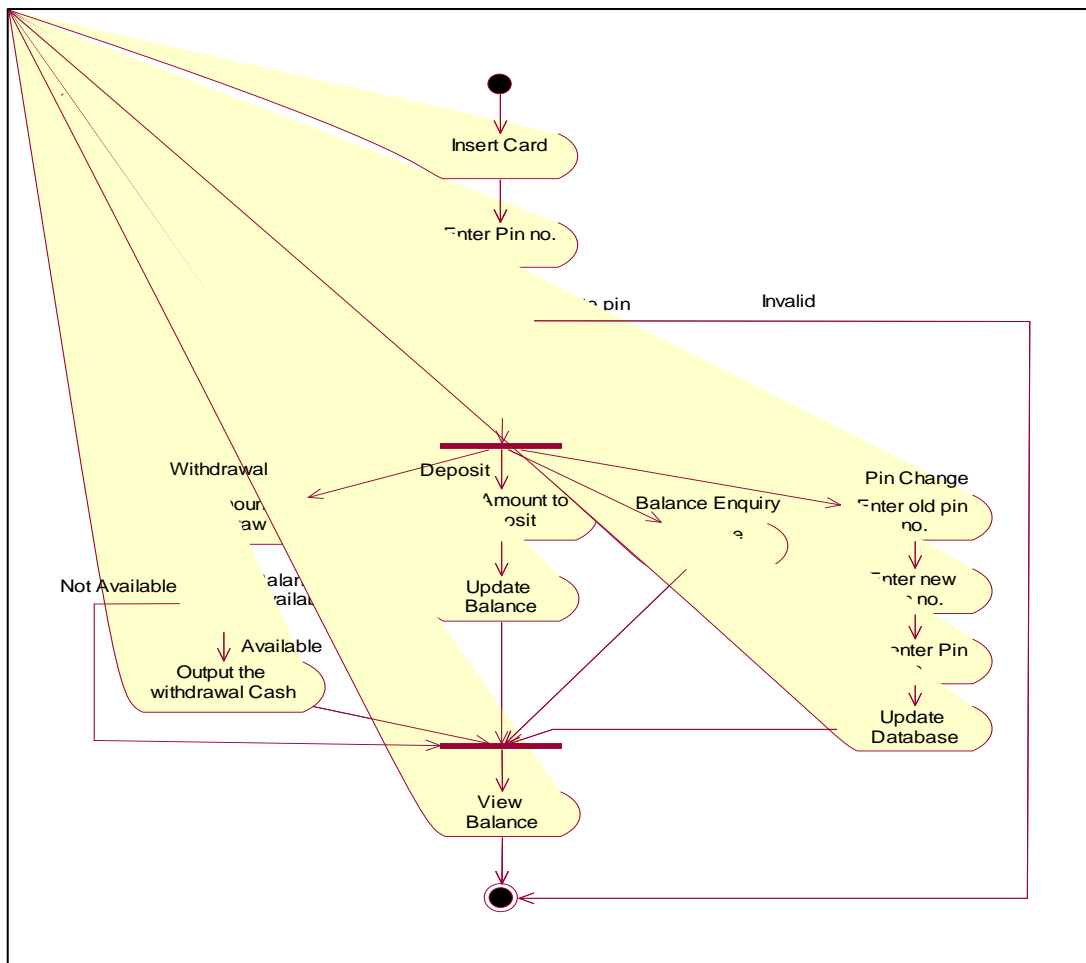
### 3.1 USECASE DIAGRAM



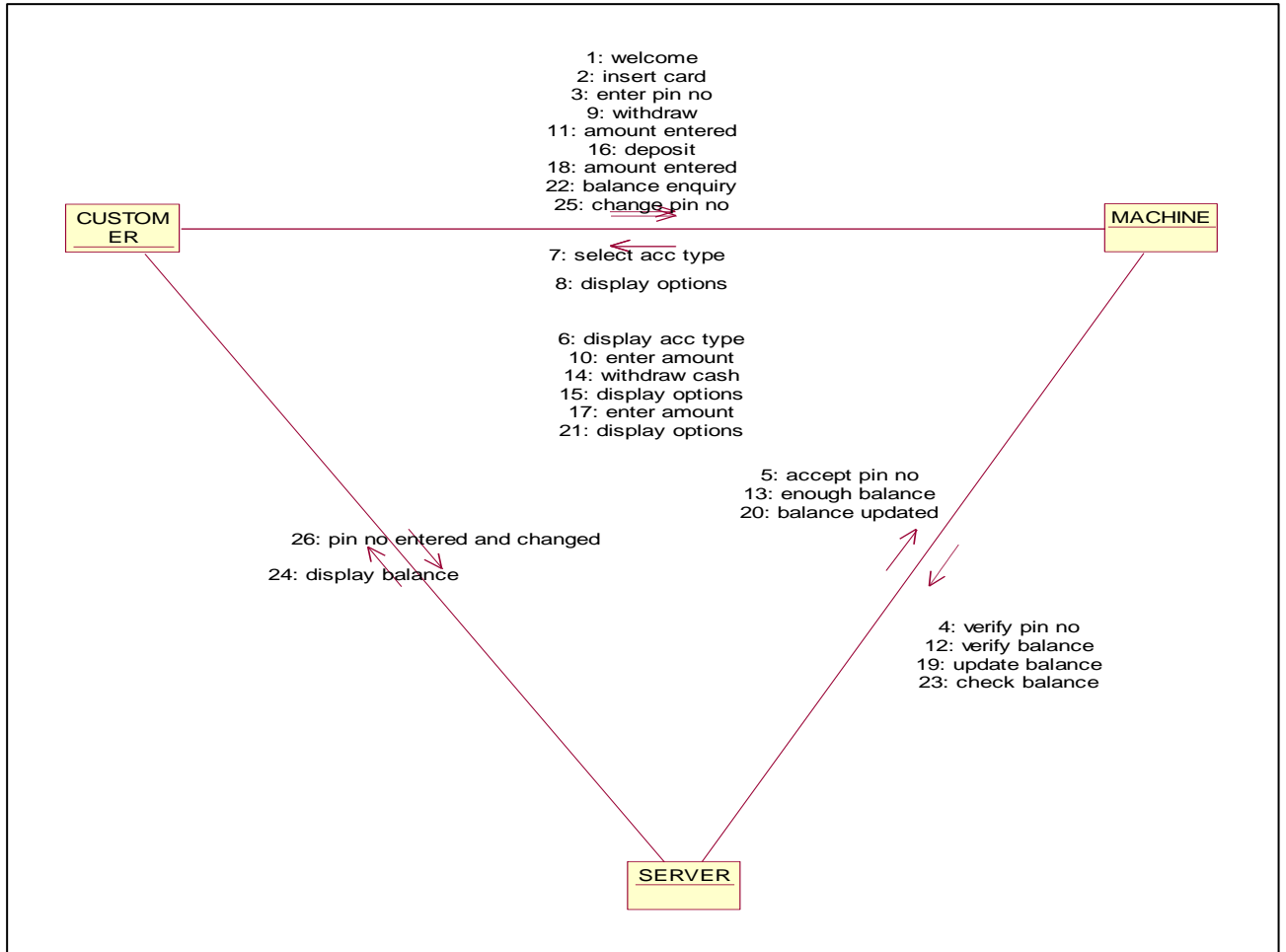
3.2 CLASS DIAGRAM



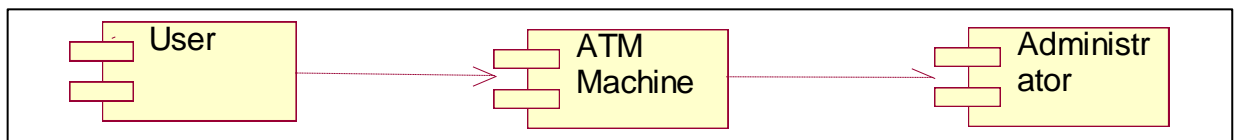
3.3 ACTIVITY DIAGRAM



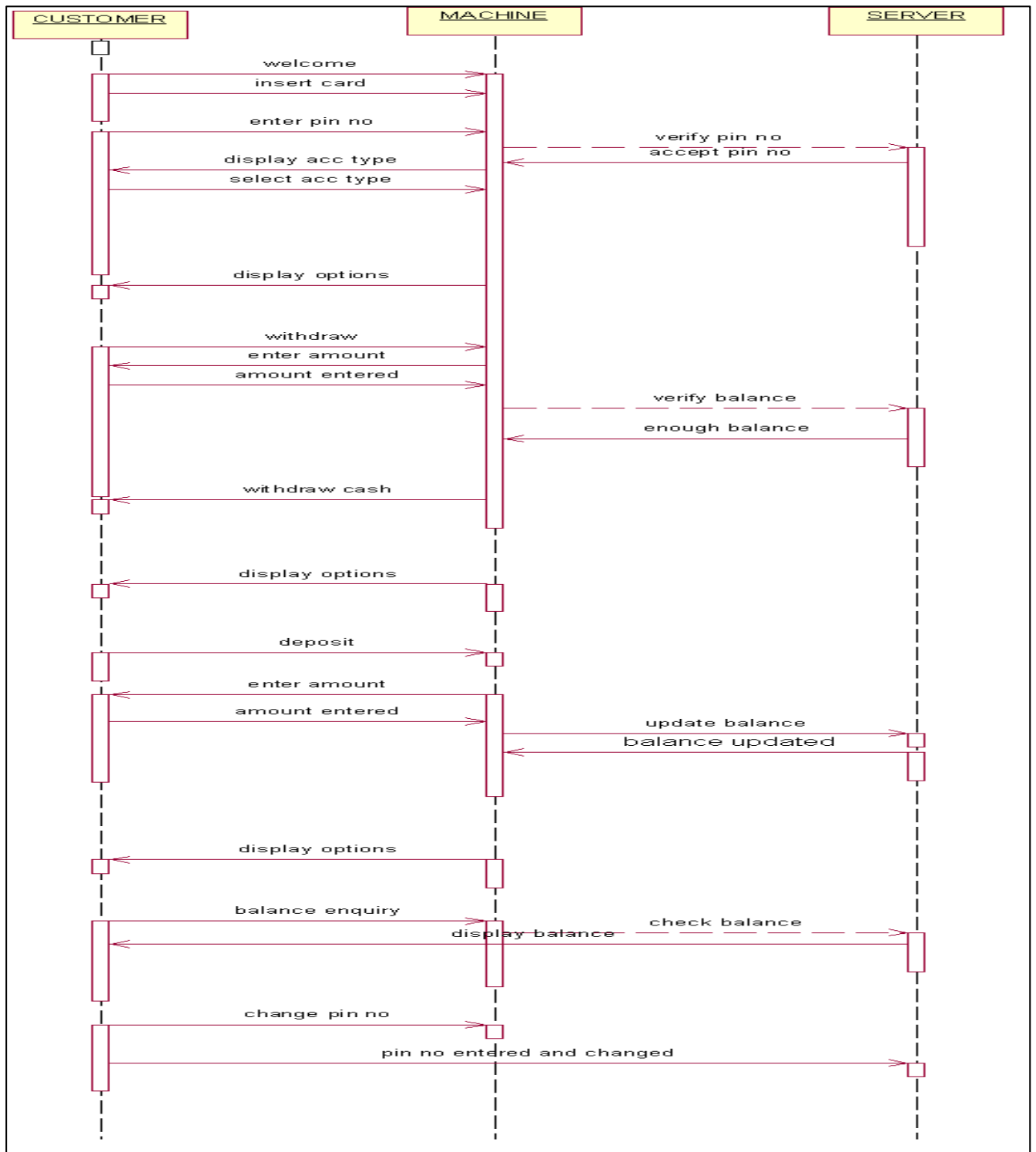
3.4 COLLABORATION DIAGRAM



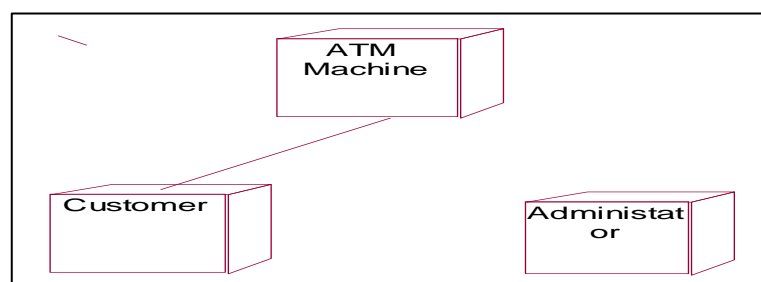
3.5 COMPONENT DIAGRAM



3.6 SEQUENCE DIAGRAM



3.7 DEPLOYMENT DIAGRAM



#### 4. DATABASE DESIGN

Database Name: ATM

Table Name : Infor atm

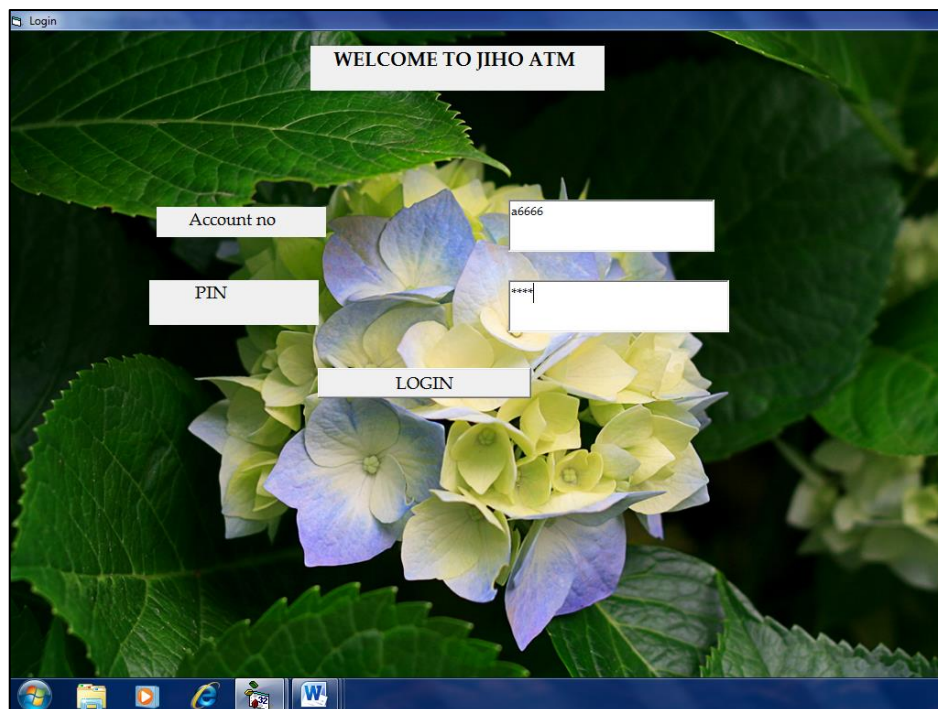
Fields	Data type
ACC NO	Text
USER NAME	Text
PIN NO	Integer
ACCOUNT BALANCE	Double(8)

Table data:

ACC NO	PIN NO	USER NAME	ACCOUNT BALANCE
a6666	9999	Magdalene	Rs.19000
b7777	7777	hhhh	Rs.35000

#### 5. IMPLEMENTATION:

Form1 (Login form)



#### Coding:

```
Private Sub Command1_Click()
Data1.Recordset.MoveFirst
While Not Data1.Recordset.EOF
If ((Text1.Text = Data1.Recordset.Fields("AccNO")) And (Text2.Text =
Data1.Recordset.Fields("PIN"))) Then
Welcome.Text1.Text = Data1.Recordset.Fields("UserName")
Welcome.Text2.Text = Data1.Recordset.Fields("AccNO")
Welcome.Show
GoTo out

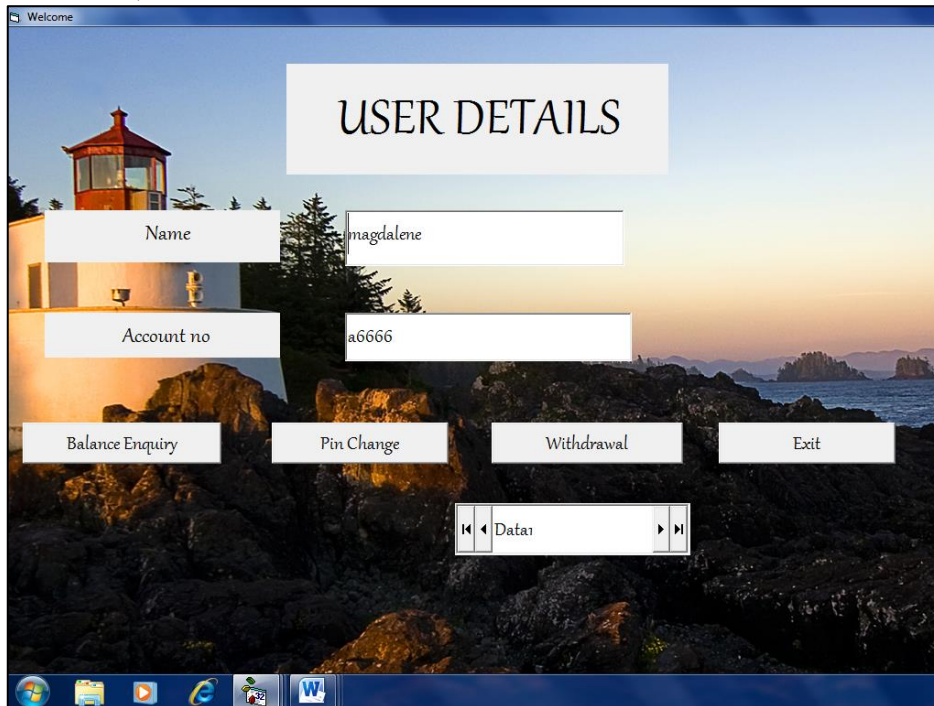
```

```

Else
Data1.Recordset.MoveNext
End If
MsgBox ("Enter correct AccNO and PIN")
Wend
out:
End Sub

```

Form2 (Welcome Form)



Coding:

```

Private Sub Command1_Click()
Data1.Recordset.MoveFirst
While Not Data1.Recordset.EOF
If ((Text1.Text = Data1.Recordset.Fields("UserName")) And (Text2.Text =
Data1.Recordset.Fields("accno"))) Then
MsgBox (Data1.Recordset.Fields("Account Balance"))
Balanceenquiry.Text1.Text = Data1.Recordset.Fields("Account Balance")
Balanceenquiry.Show
GoTo out
Else
Data1.Recordset.MoveNext
End If
MsgBox ("check user name and account number")
Wend
out:
End Sub

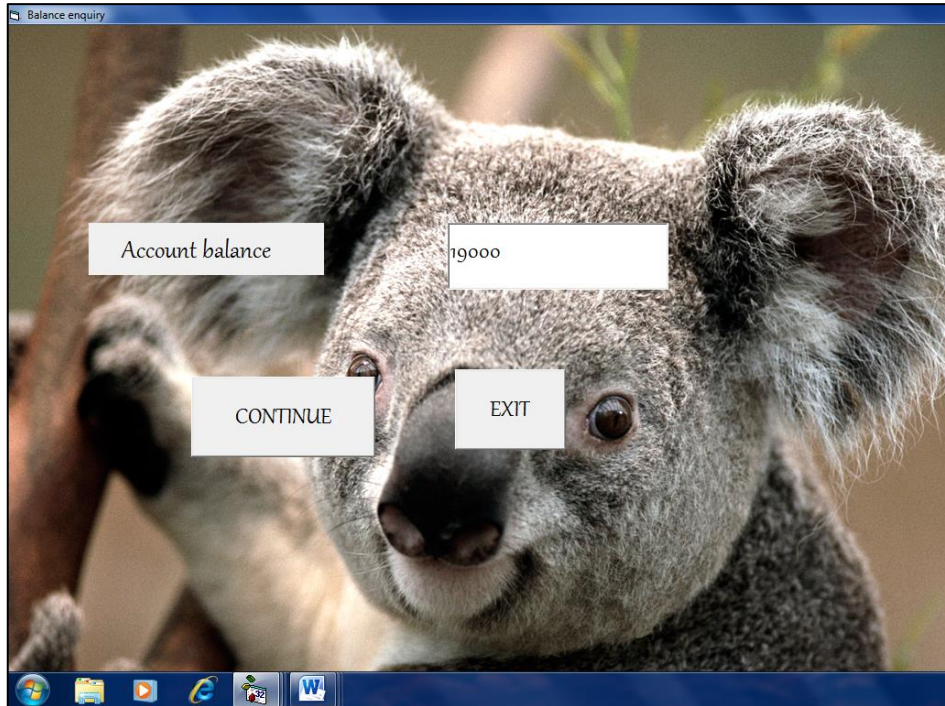
Private Sub Command2_Click()
Pinchange.Show
End Sub

Private Sub Command3_Click()
Withdrawal.Show
End Sub

```



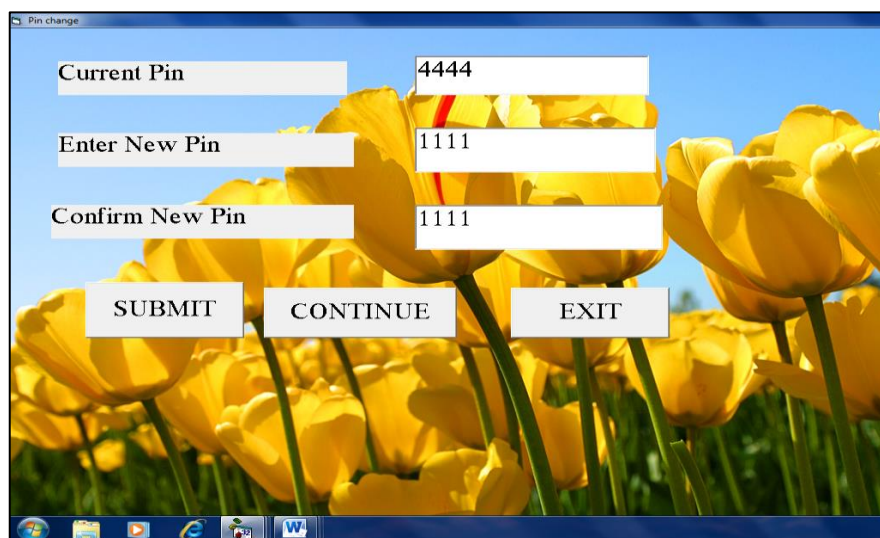
## Form3(Balanceenquiry form)

Coding:

```
Private Sub Command1_Click()  
Welcome.Show  
End Sub
```

```
Private Sub Command2_Click()  
End  
End Sub
```

## Form4 (Pinchange form)

Coding:

```
Private Sub Command1_Click()  
Data1.Recordset.MoveFirst  
Data1.Recordset.Edit
```

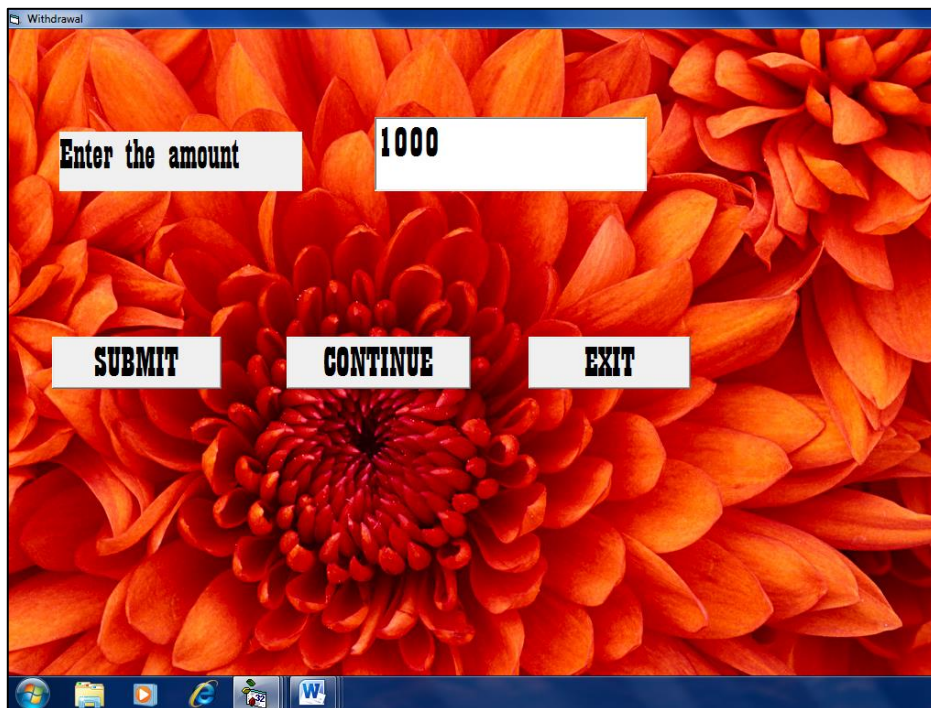


```
While Not Data1.Recordset.EOF
If (Text1.Text = Data1.Recordset.Fields("PIN")) Then
Data1.Recordset.Fields("PIN") = Text3.Text
MsgBox ("Pin Change Success")
Data1.Recordset.Update
GoTo out
Else
Data1.Recordset.MoveNext
End If
MsgBox ("Pin change not succesful")
Wend
out:
End Sub
```

```
Private Sub Command2_Click()
Welcome.Show
End Sub
```

```
Private Sub Command3_Click()
End
End Sub
```

Form5 (Withdrawal form)



Coding:

```
Private Sub Command1_Click()
Dim s
Welcome.Data1.Recordset.Edit
If Welcome.Data1.Recordset.Fields("Account Balance") > Text1.text then
s=val(Welcome.Text2.text) – val(Text1.text)
Welcome.Data1.Recordset.Fields("Account Balance") = s
```

```
Welcome.Data1.Recordset.Update
Else
Msgbox("Insufficient amount")
End if
End Sub
```

```
Private Sub Command2_Click()
Welcome.Show
End Sub
```

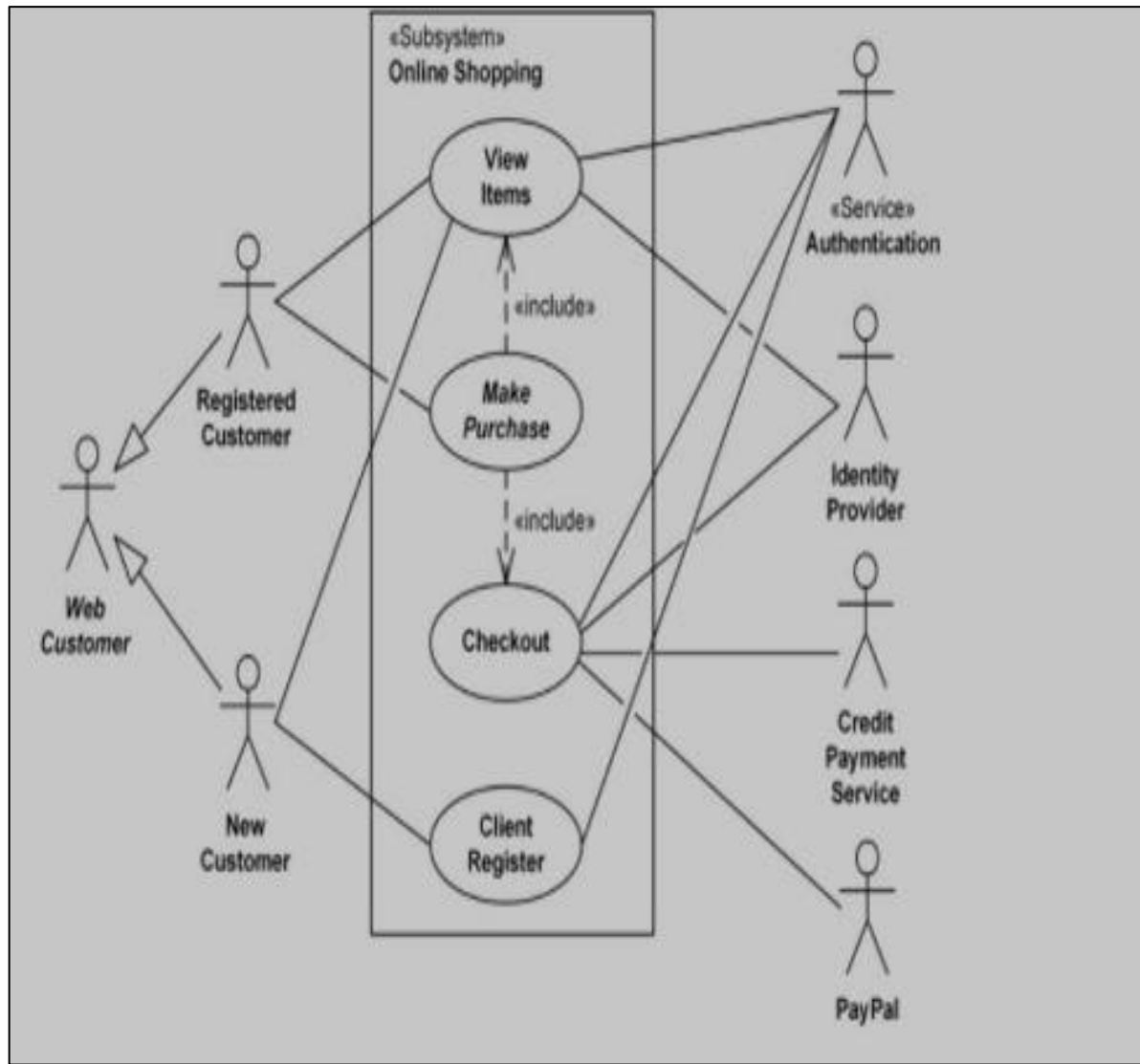
```
Private Sub Command3_Click()
End
End Sub
```

## 6. TESTING:

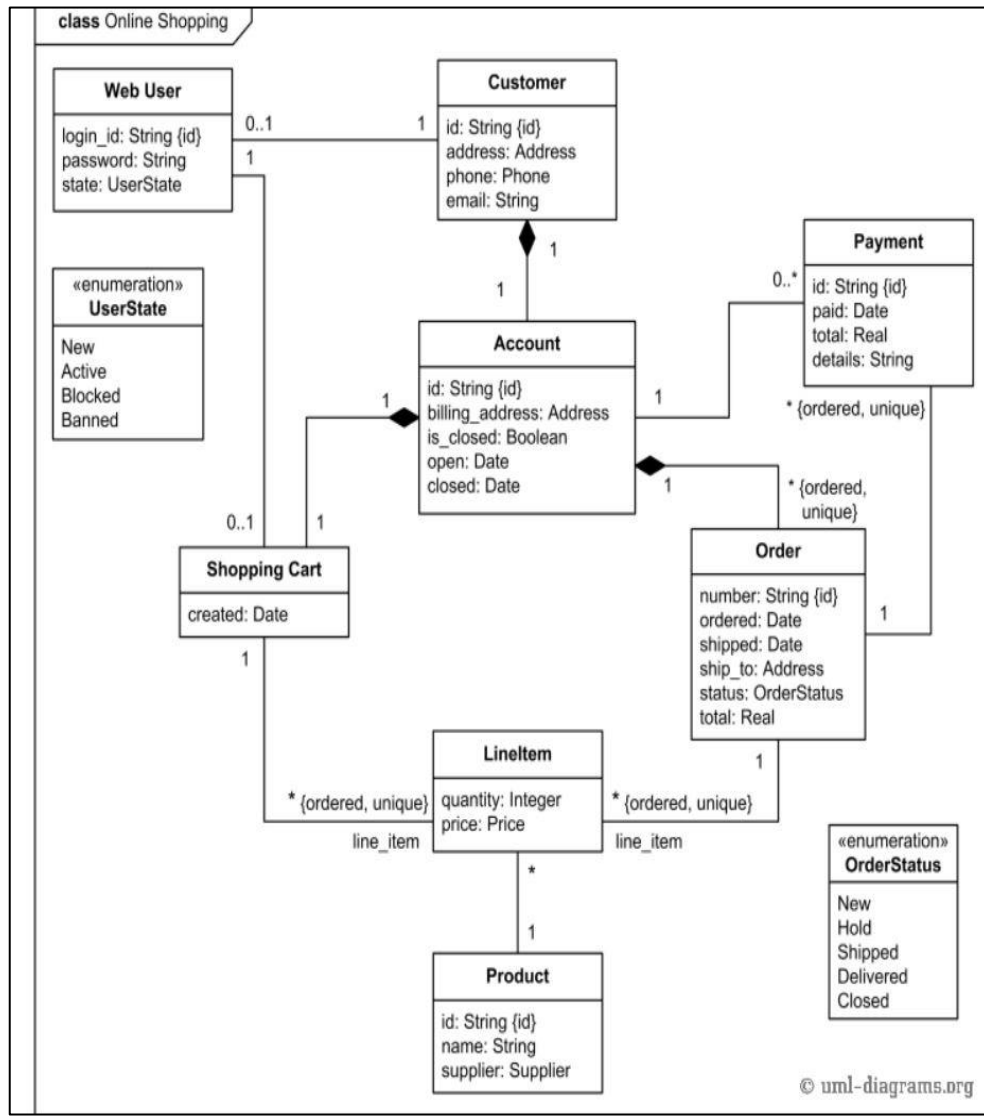
<b>Test case ID: Test_01</b>					
Test priority (Low/Medium/High):Medium					
Module name: login					
Test title :verify login with valid username and password					
Precondition: user has invalid username and password					
S.NO	TEST STEPS	EXPECTED RESULTS	ACTUAL RESULTS	STATUS	NOTES
1	Provide valid User name	User should Be able to login	The user is able to move to next Entry	Success	-
2	Provide valid password	User should be Able to Login	The user is able To login Successfully	Success	In case of wrong Password was given an error Message box was displayed
3	Click login	User should be able to navigate to next page after validation	User name and password is validated and next page is displayed	Success	Incase user gives wrong entry the sign in page remains active

4	Click signup	User should be able to navigate to next page where user enters his credentials	User navigates to the signup page where his user name and password is validated	success	-
---	--------------	--	---	---------	---

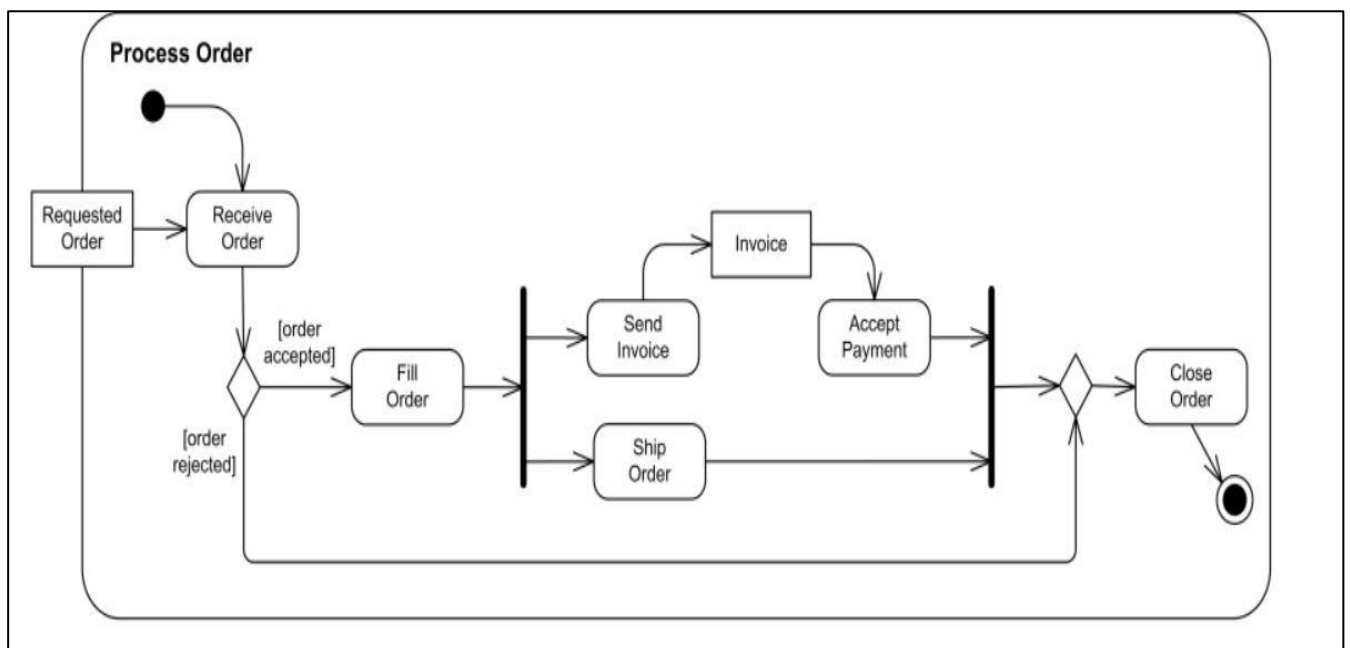
**RESULT:**The ATM system was designed and implemented successfully.

**PROGRAM 4 : E-Commerce ( various online shopping portals like FlipKart / Amazon /Myntra)****AIM : E-Commerce ( various online shopping portals like FlipKart/Amazon/Myntra)****3. DESIGN****3.1 USECASE DIAGRAM**

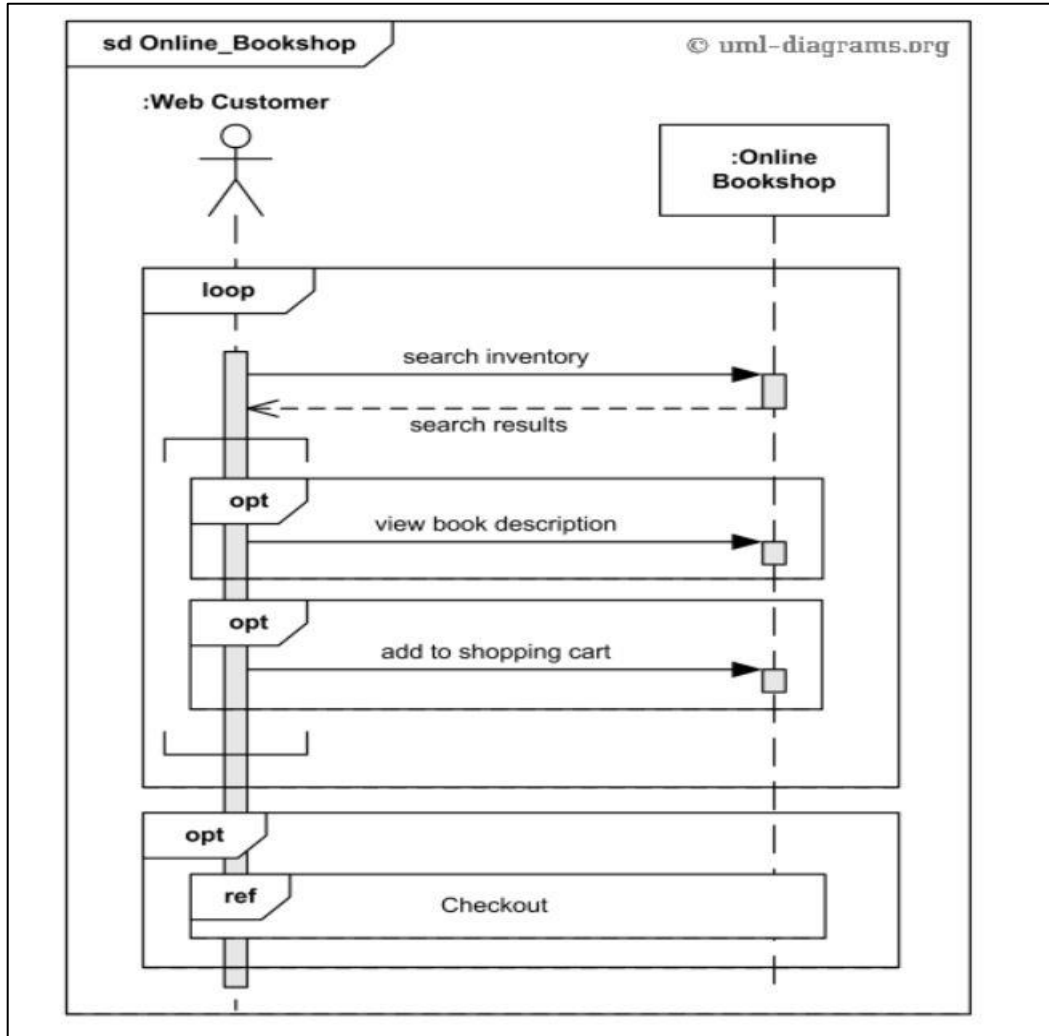
CLASS DIAGRAM



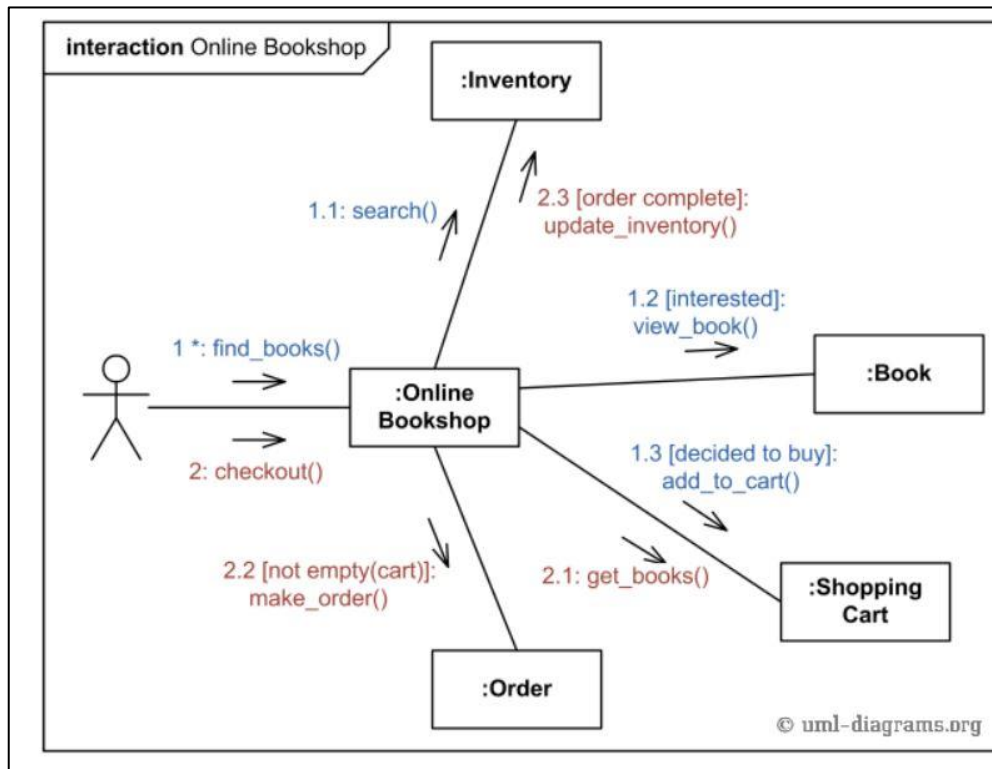
ACTIVITY DIAGRAM



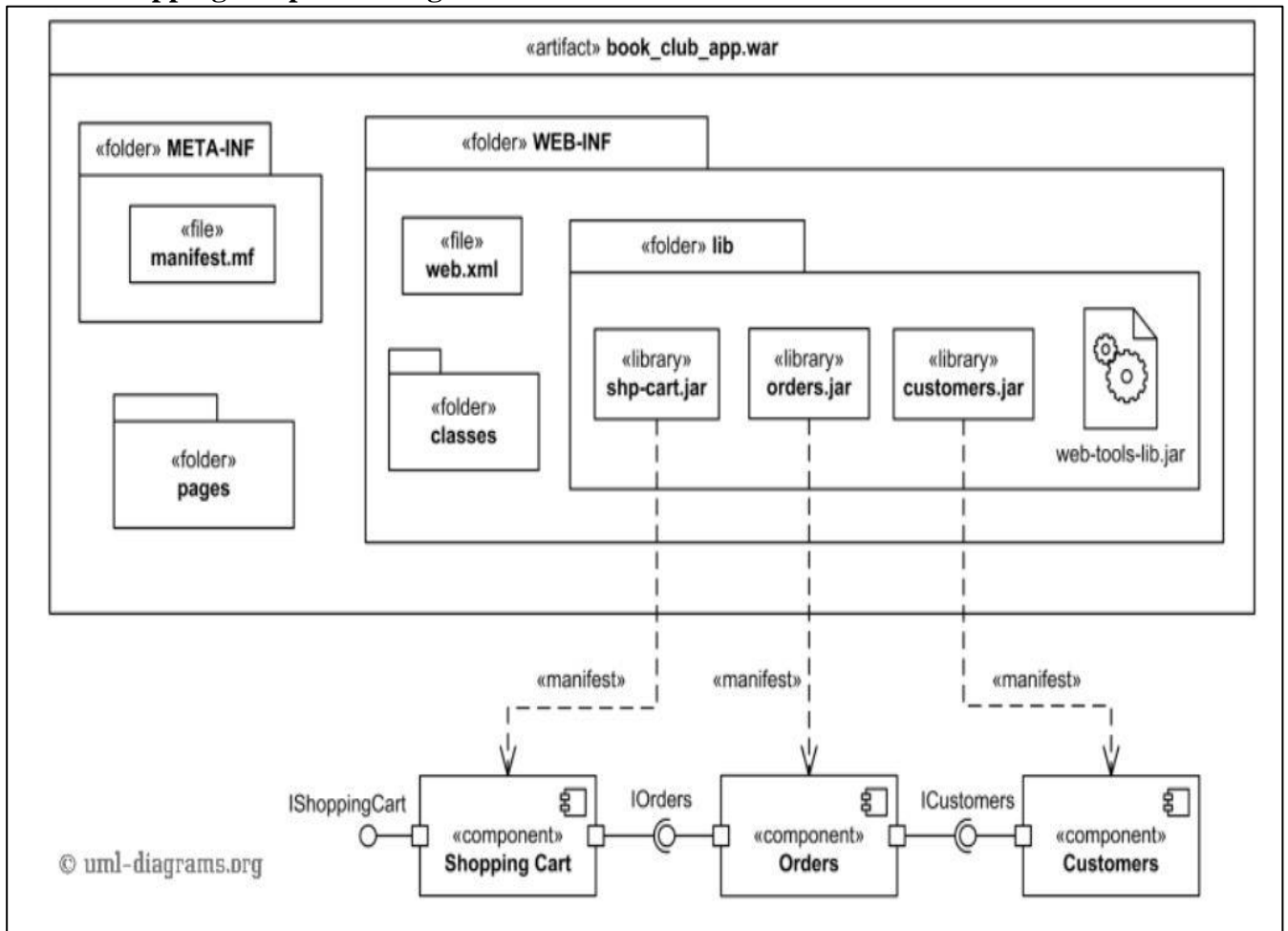
SEQUENCE DIAGRAM



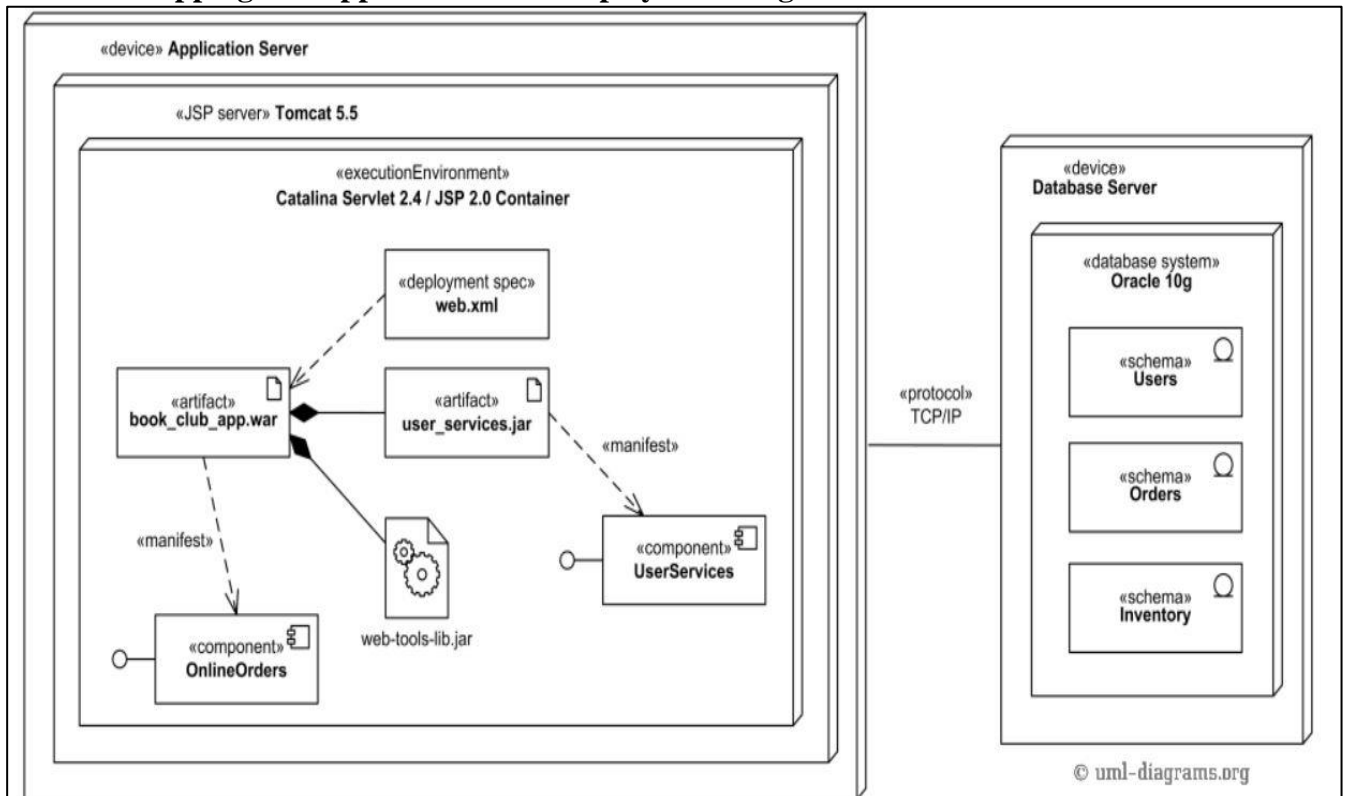
COLLABORATION DIAGRAM:



Online shopping component diagram



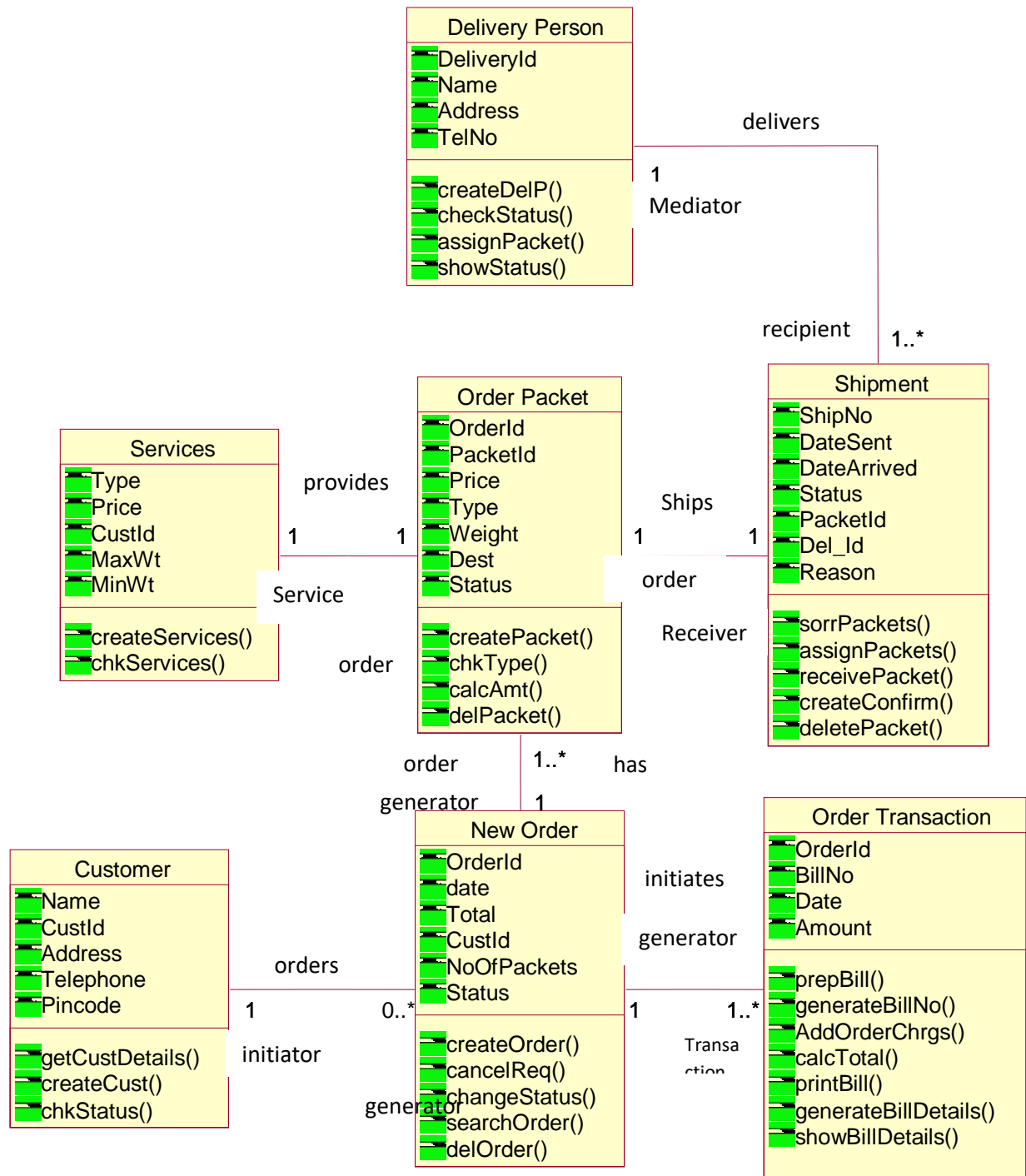
Online shopping web application UML deployment diagram



**PROGRAM 5: Logistics (Postal/Courier:IndiaPost/DTDC/UPS/FedEx,Freight:Maersk)**

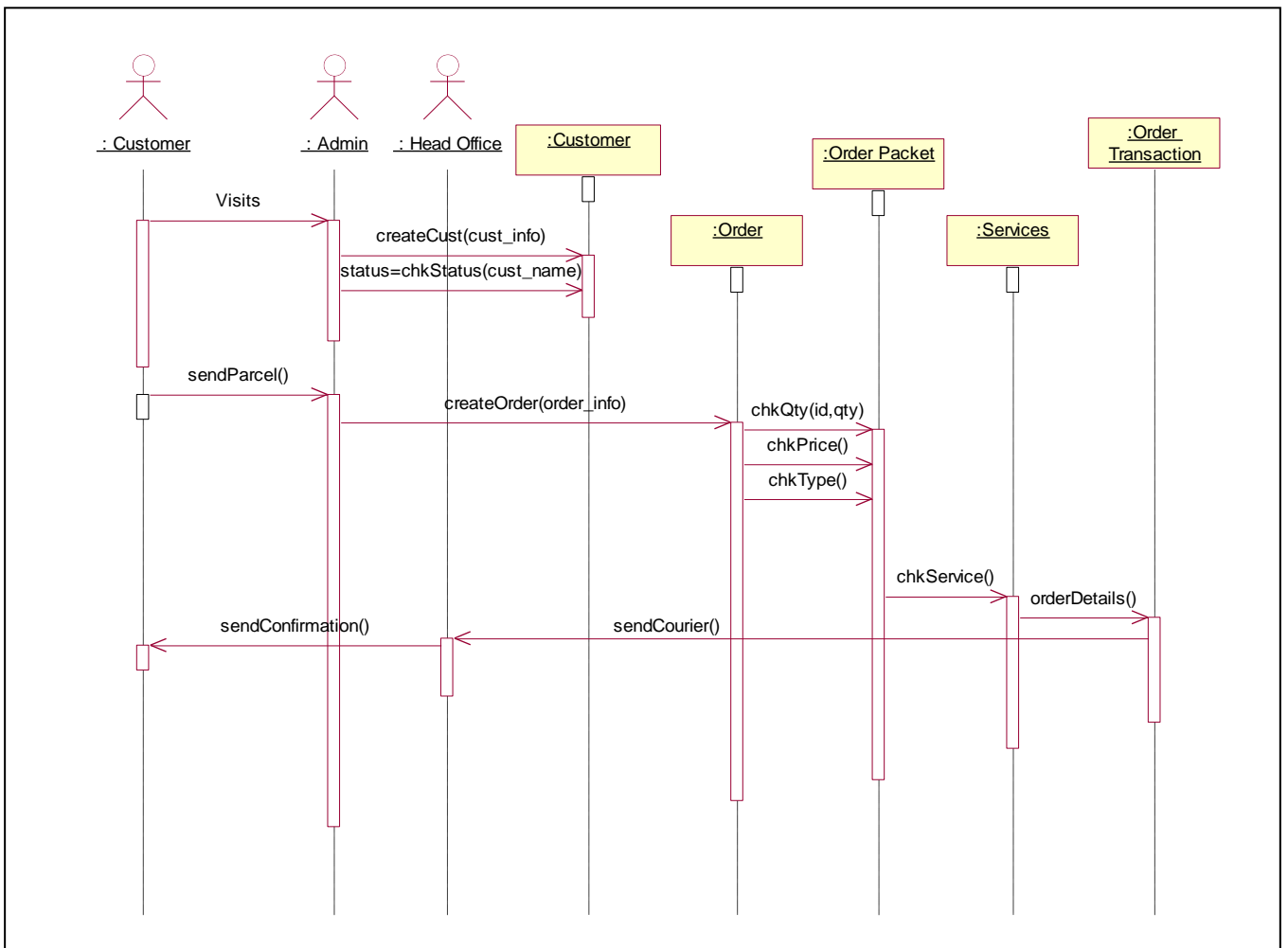
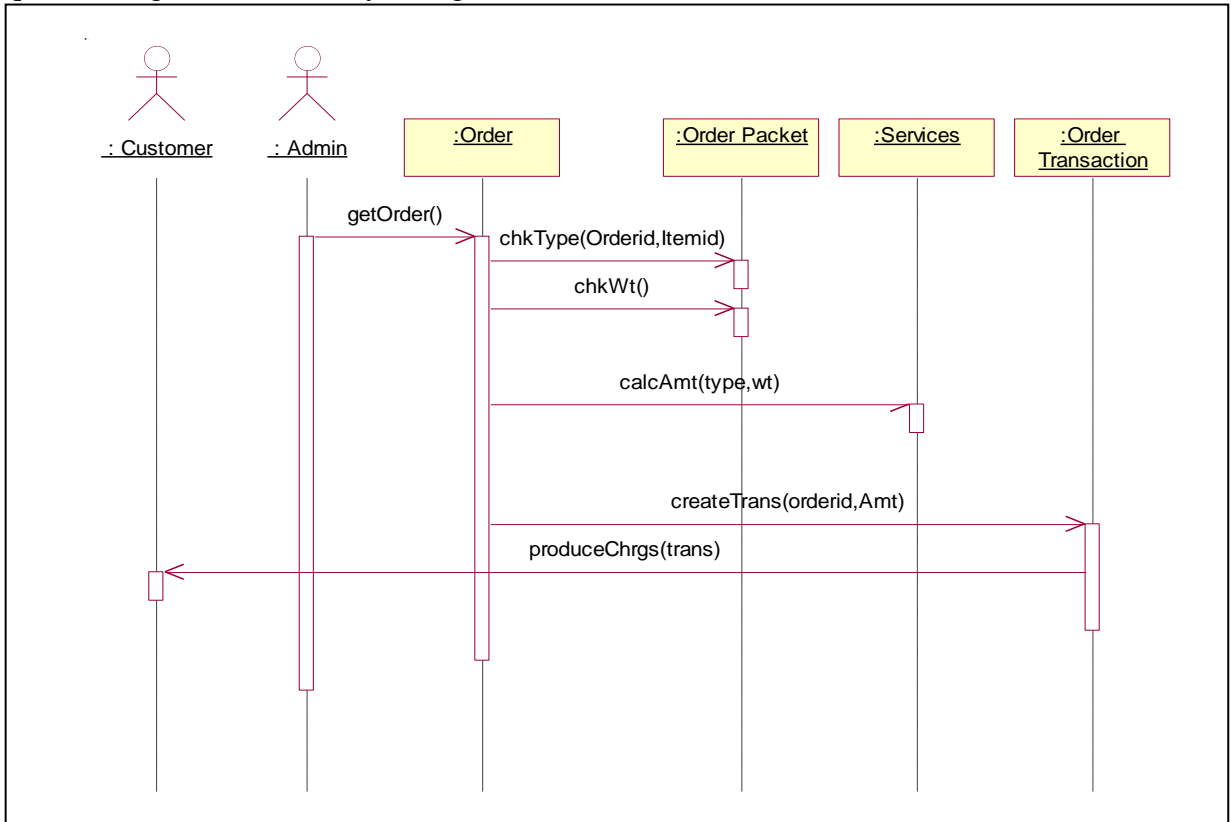
**AIM : Logistics (Postal/Courier:IndiaPost/DTDC/UPS/FedEx,Freight:Maersk)**

**CLASS DIAGRAM**

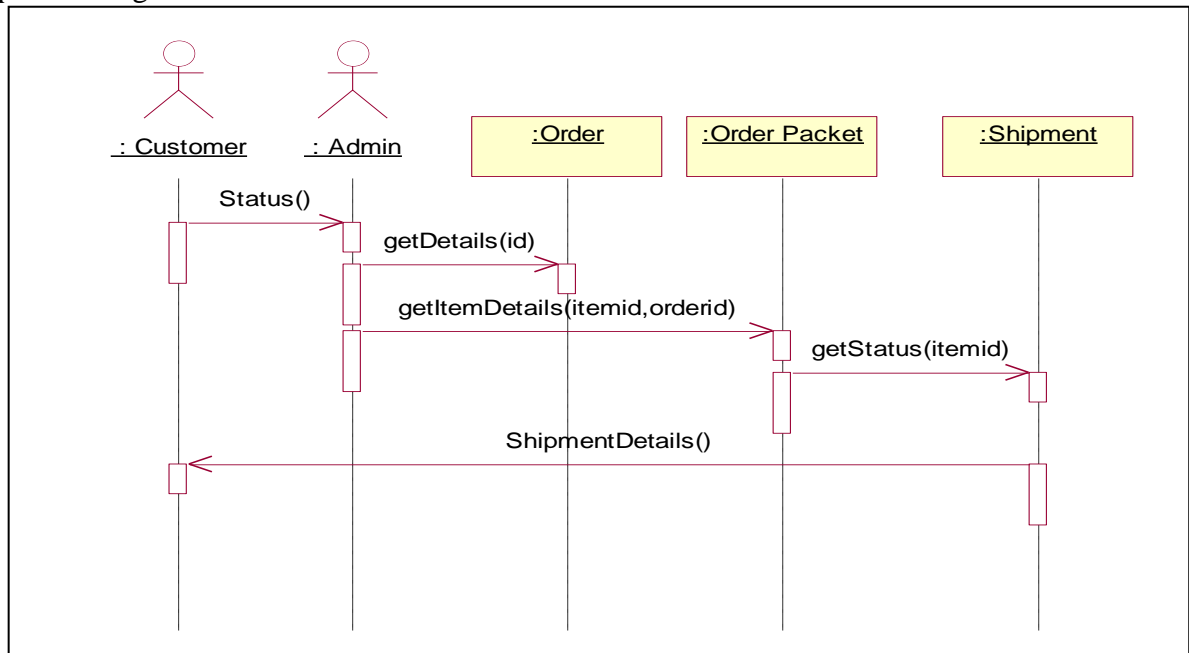




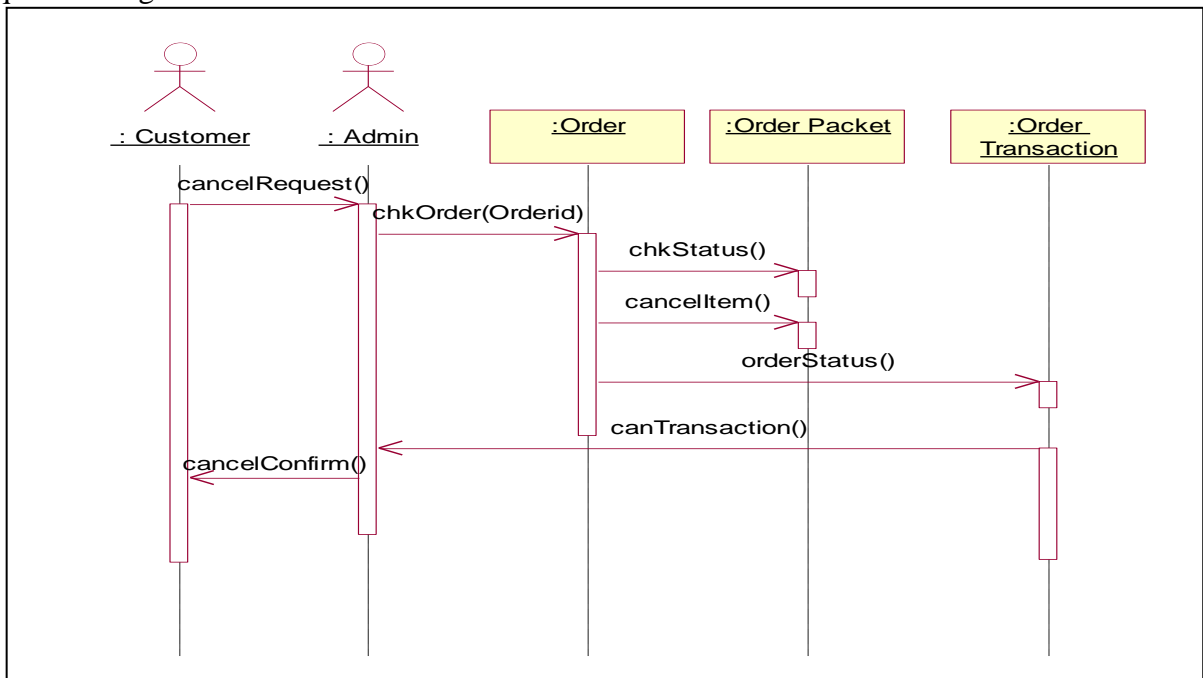
Sequence Diagram for Create New Order  
Sequence Diagram for Delivery Charges



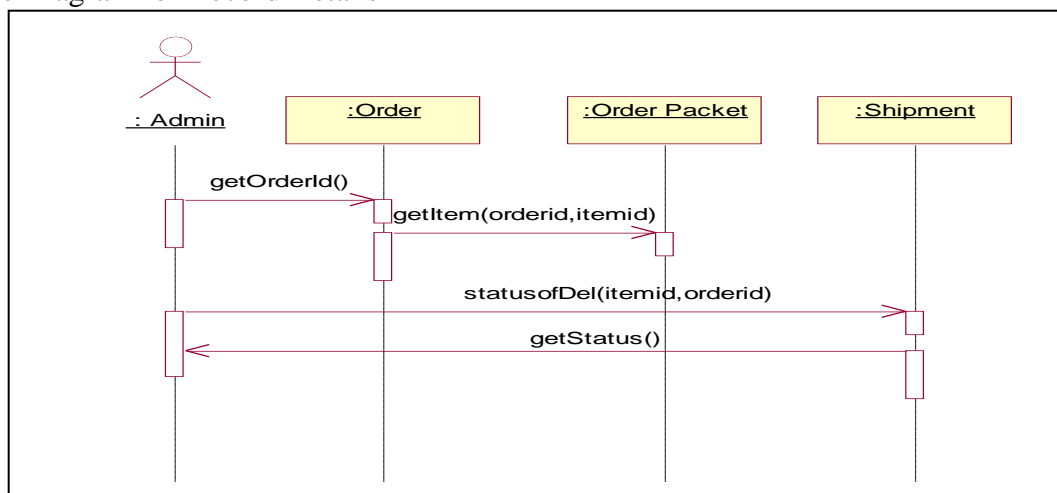
Sequence Diagram for Status



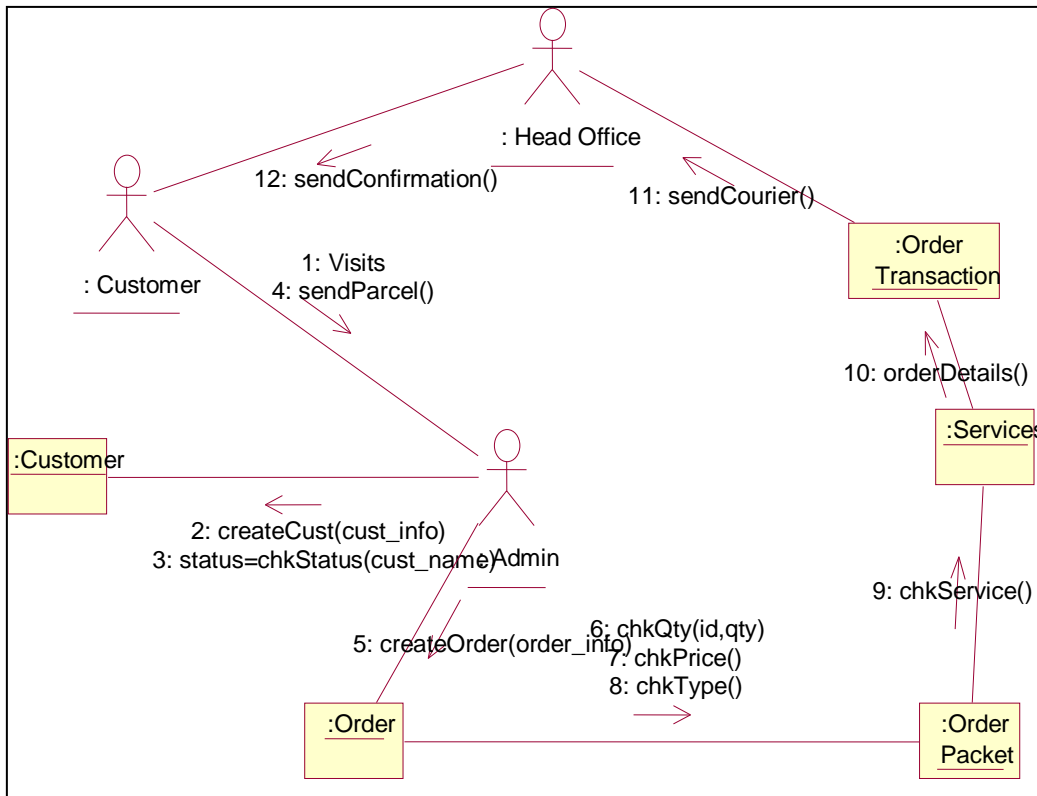
Sequence Diagram for Cancel Order



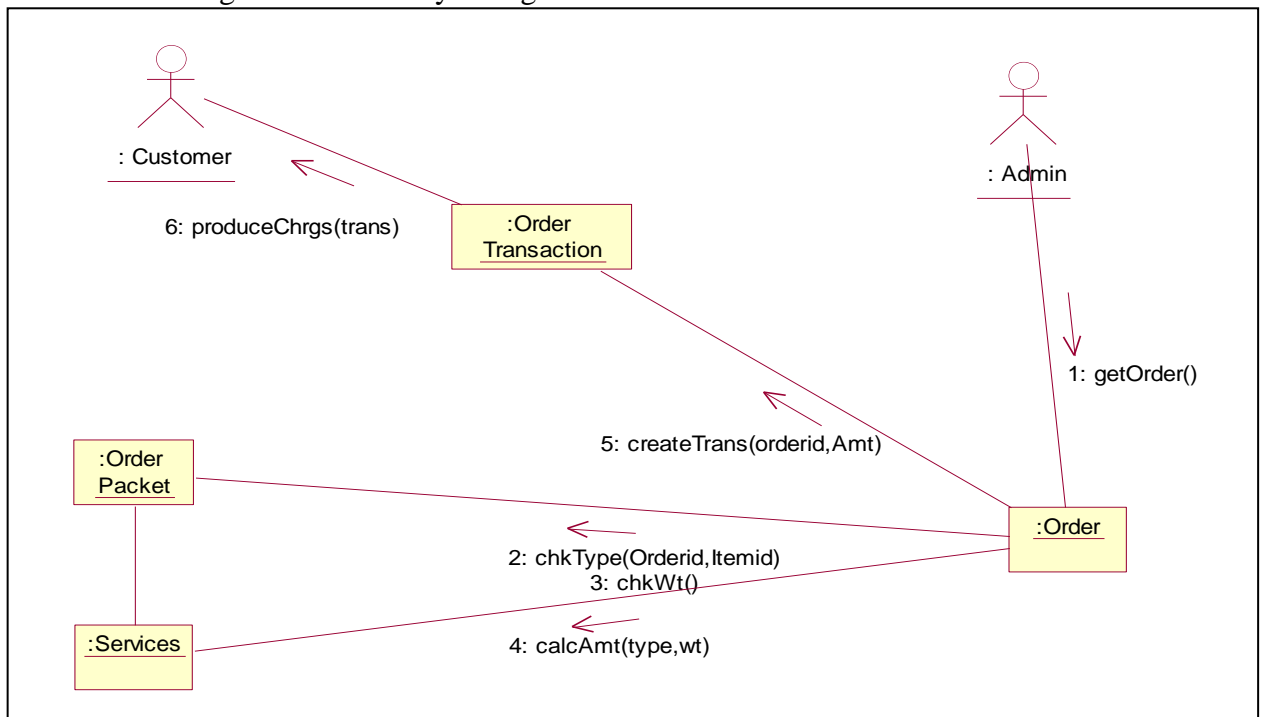
Sequence Diagram for Record Details



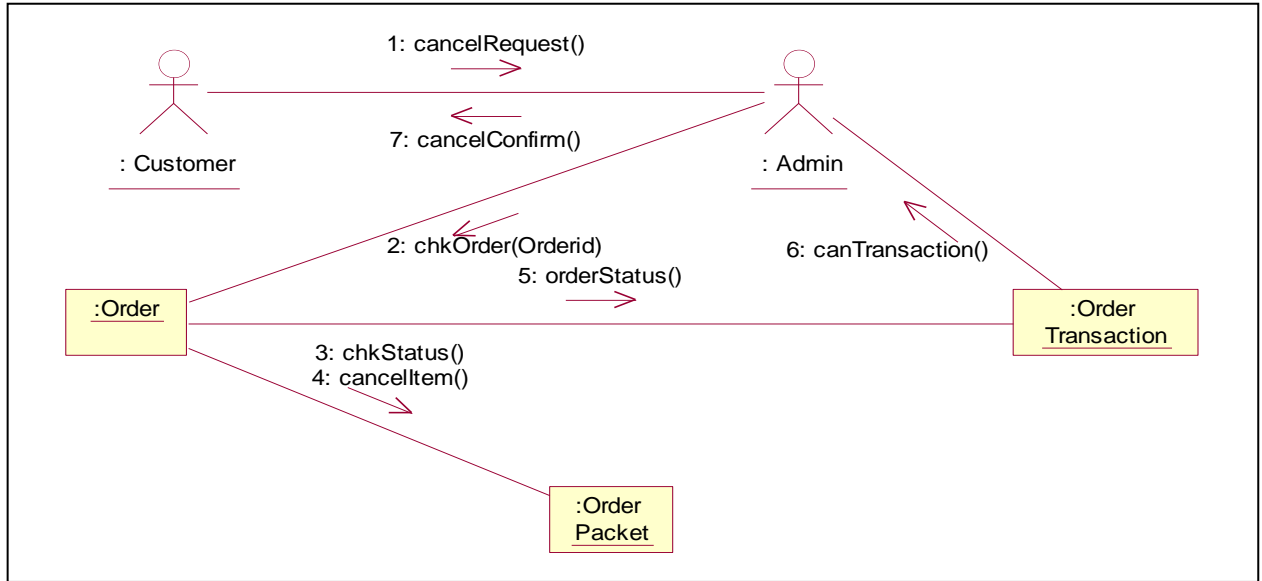
Collaboration Diagram for Create New Order



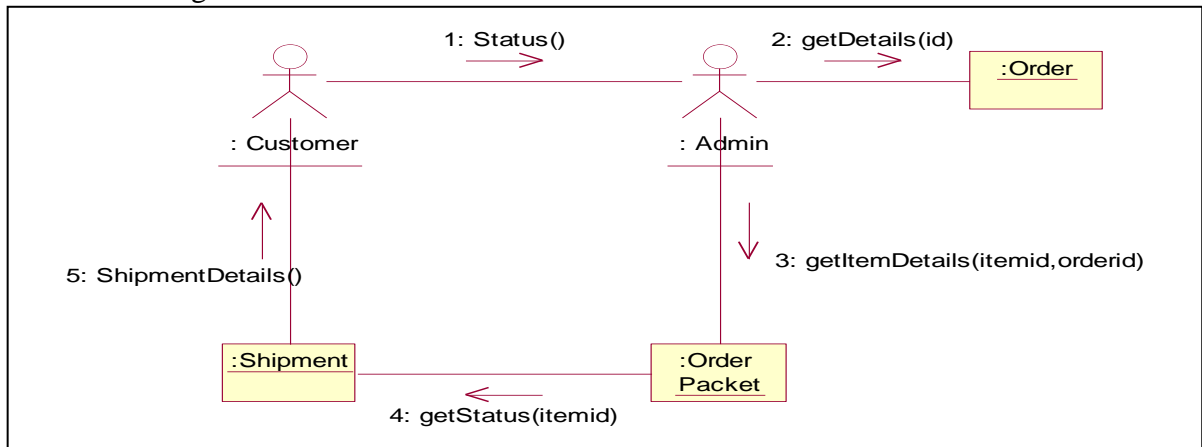
Collaboration Diagram for Delivery Charges



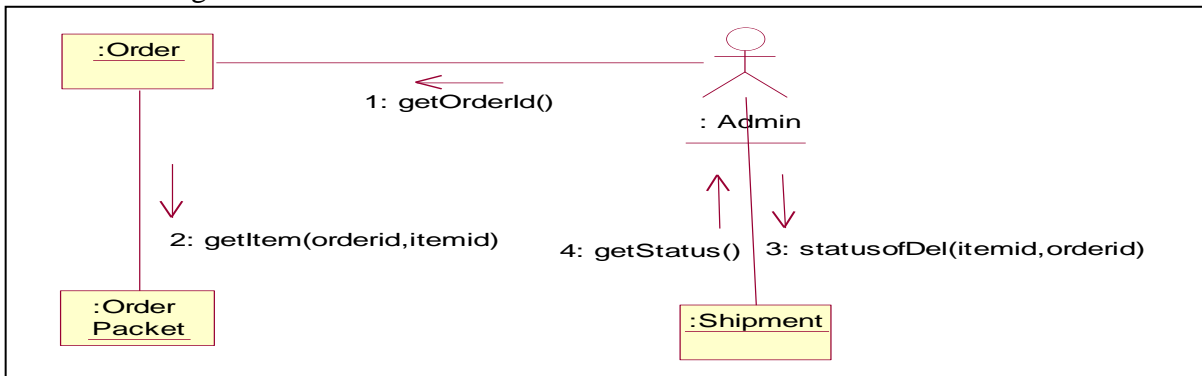
Collaboration Diagram for Cancel Order



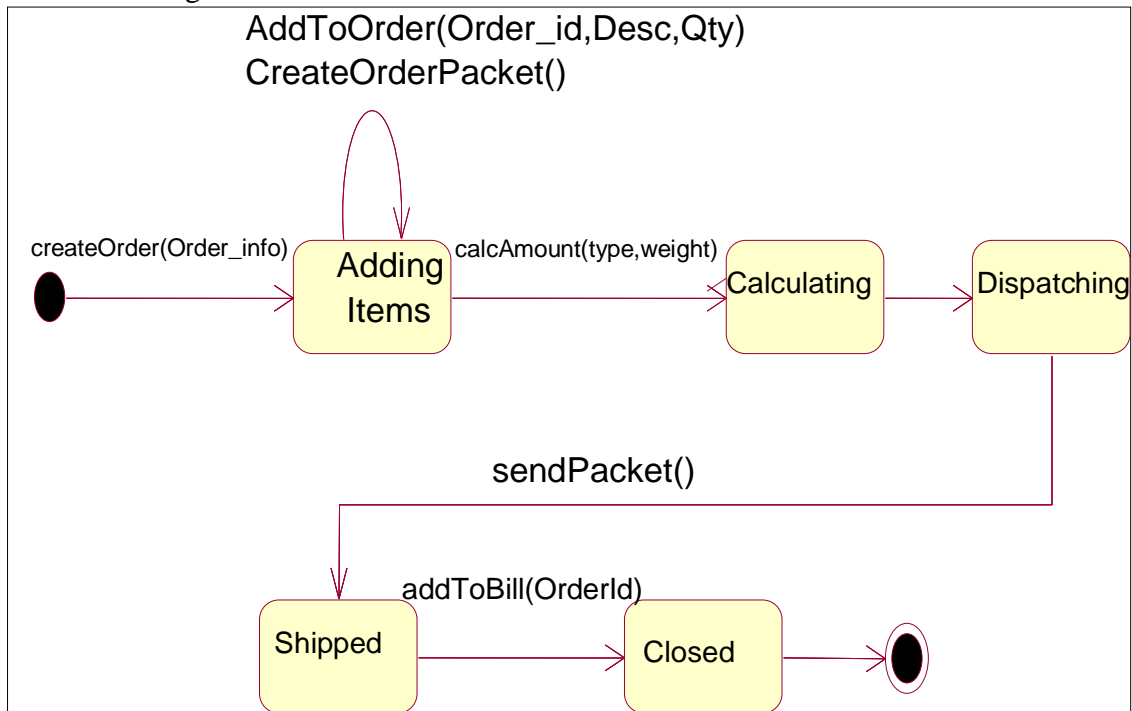
Collaboration Diagram for Status



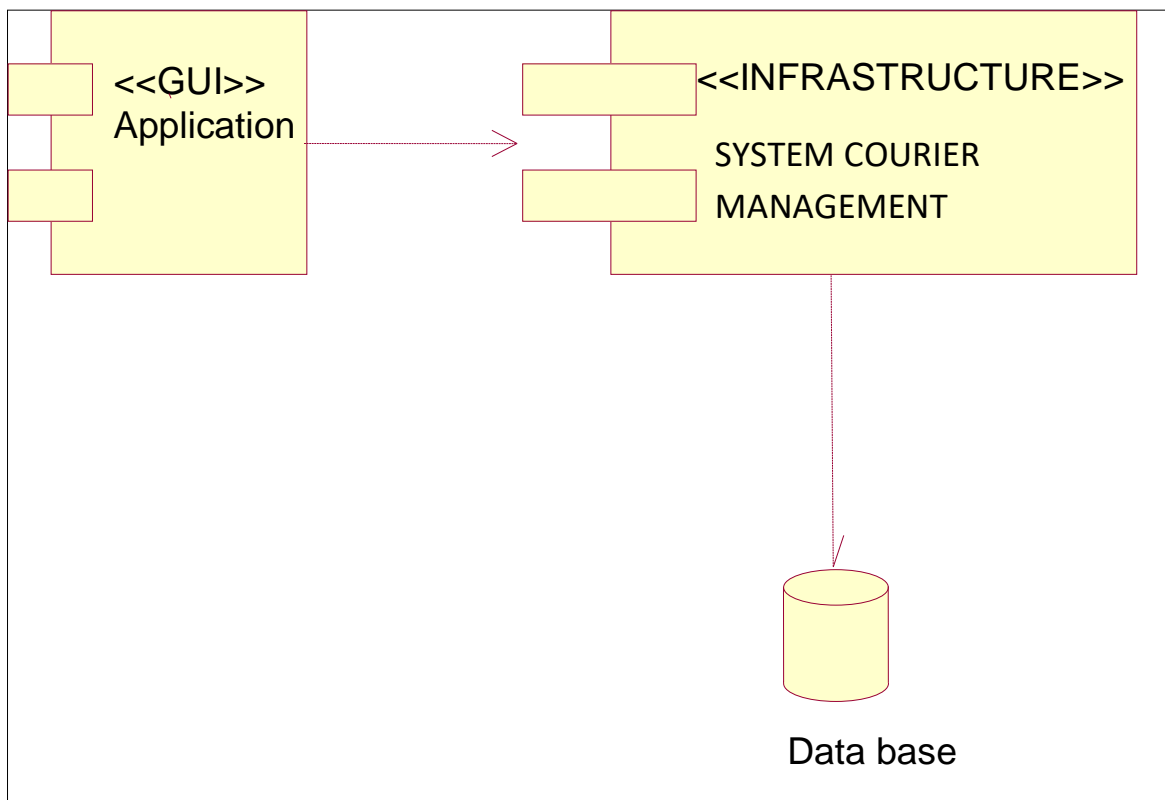
Collaboration Diagram for Record Details



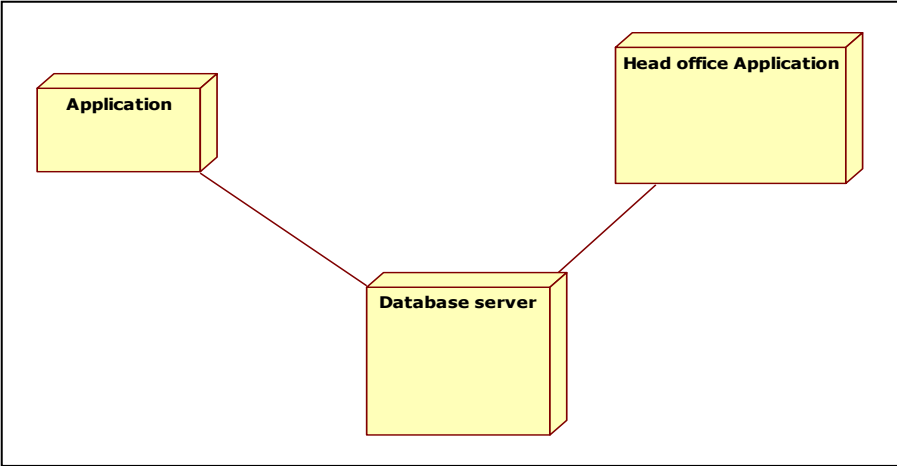
Statechart Diagram



COMPONENT Diagram Courier Management System



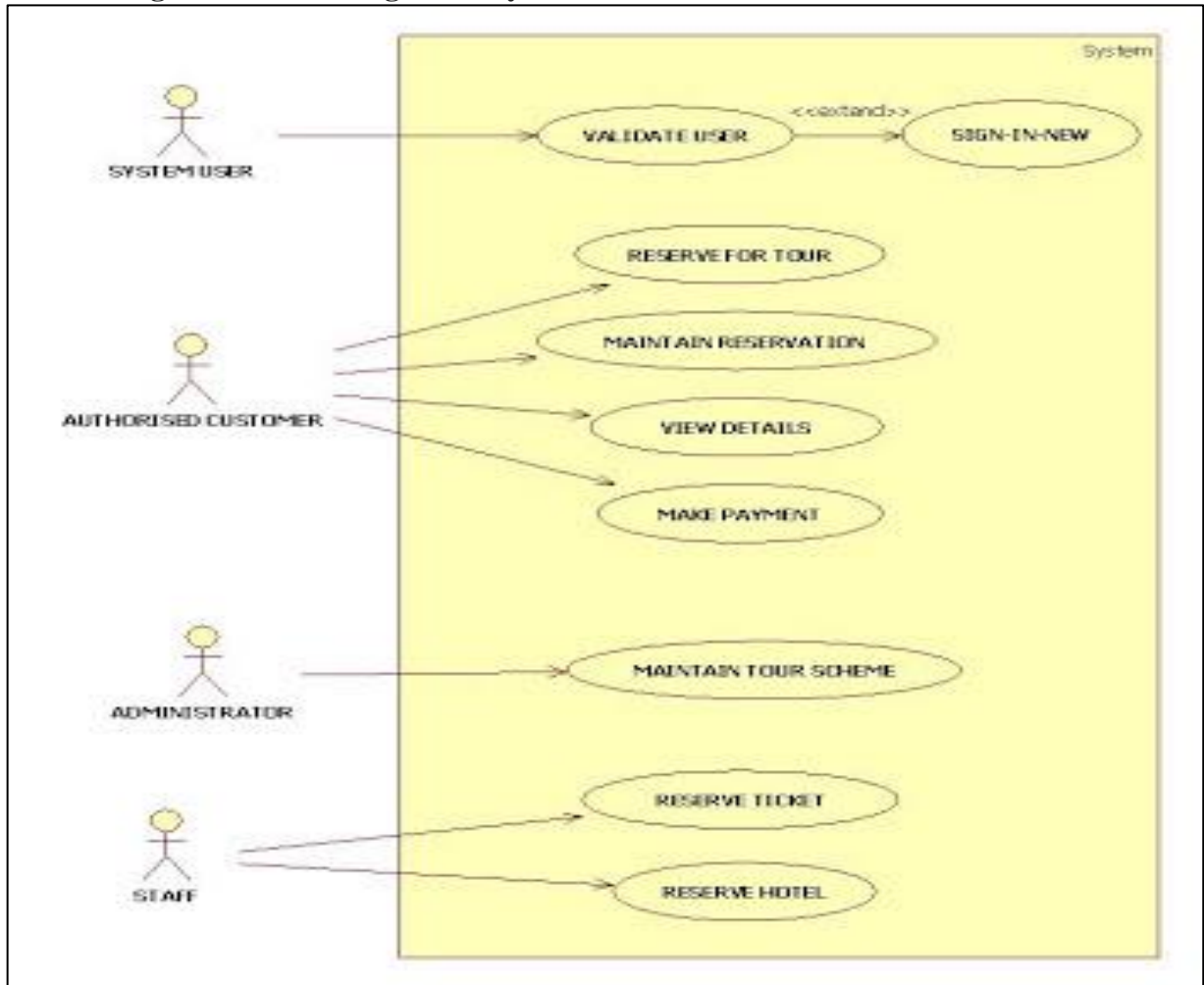
Deployment Diagram Courier Management System



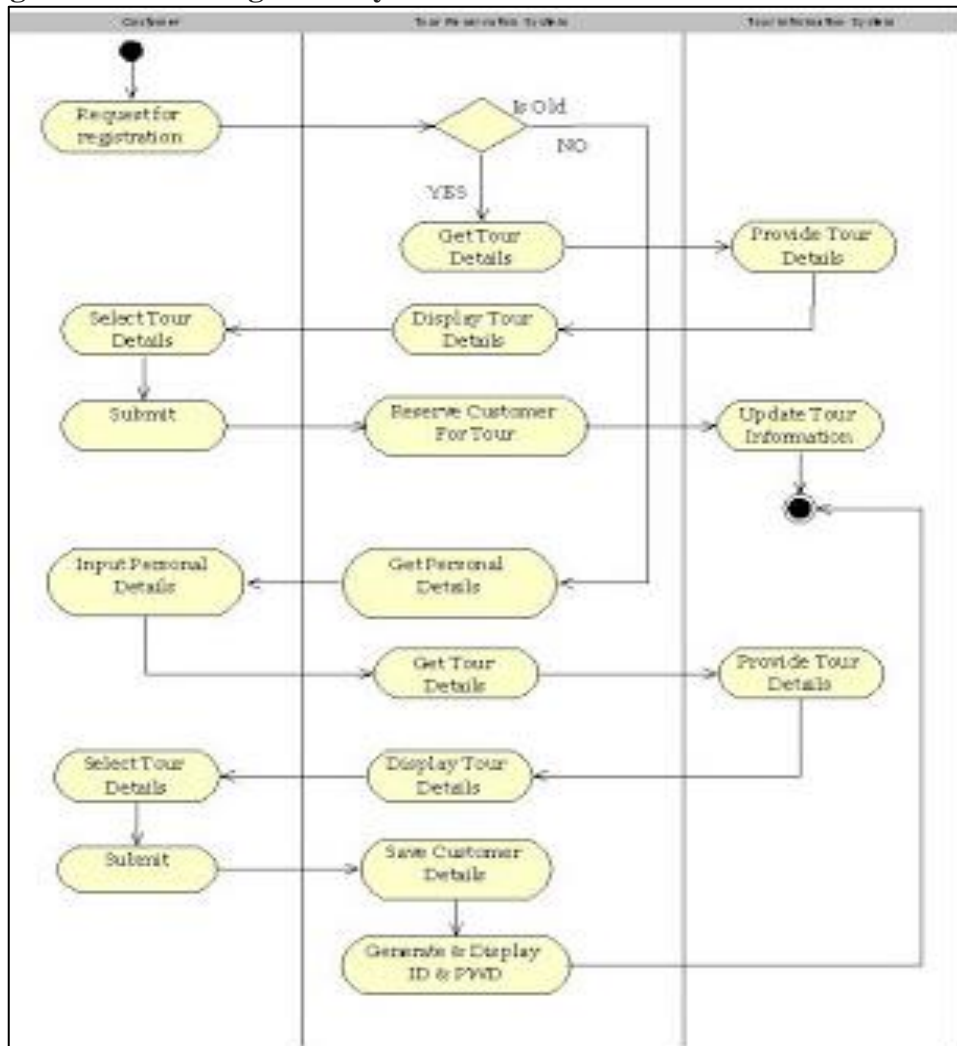
**PROGRAM 6: Hospitality(Tourism Management:Telangana,Tourism/IncredibleIndia,Event Management:MeraEvents/BookMyShow/Explara/EventBrite)**

**AIM : Hospitality(Tourism Management:Telangana,Tourism/IncredibleIndia,Event Management:MeraEvents/BookMyShow/Explara/EventBrite)**

Use Case Diagram Tour Management System :-

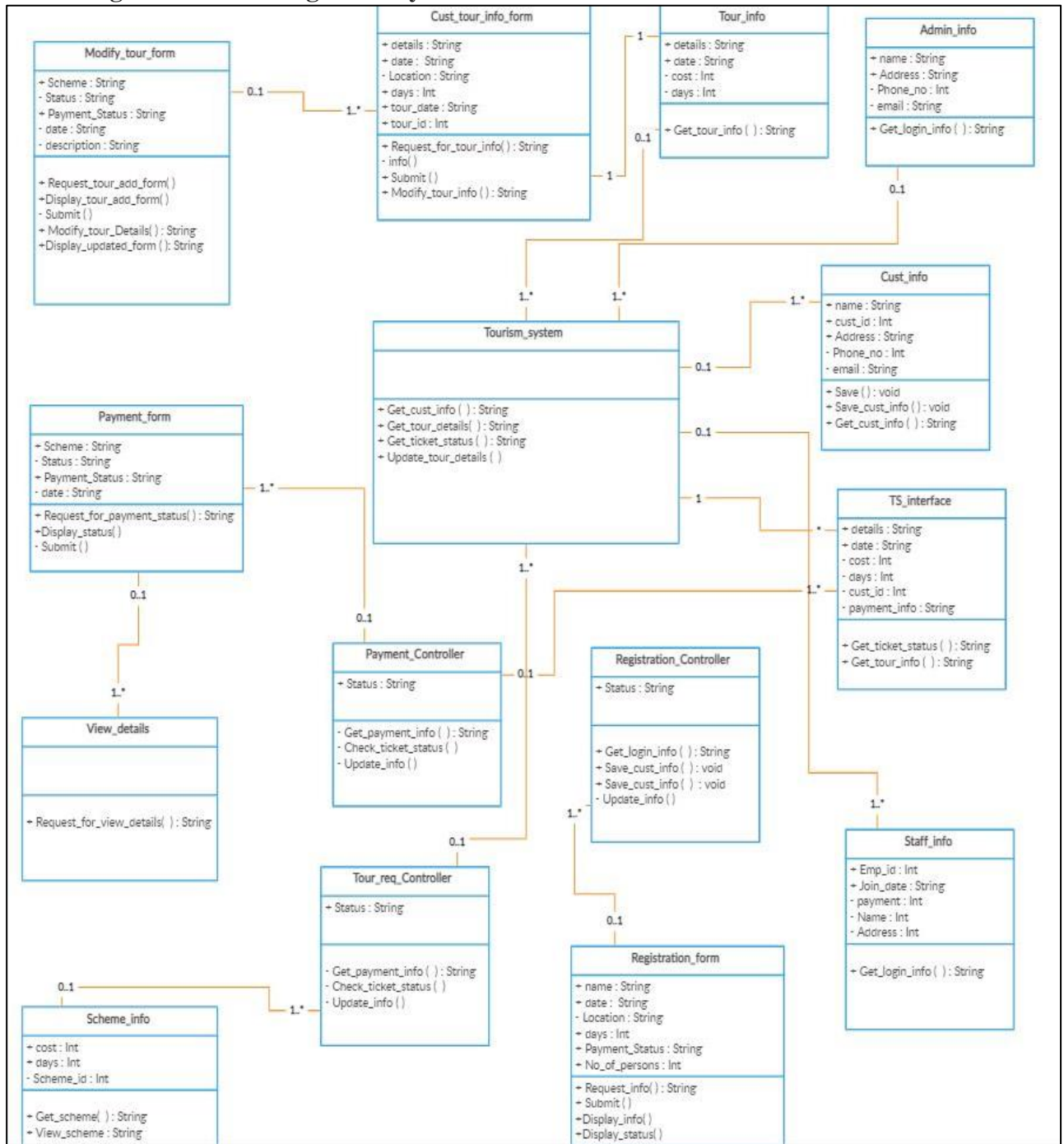


Activity Diagram Tour Management System :-

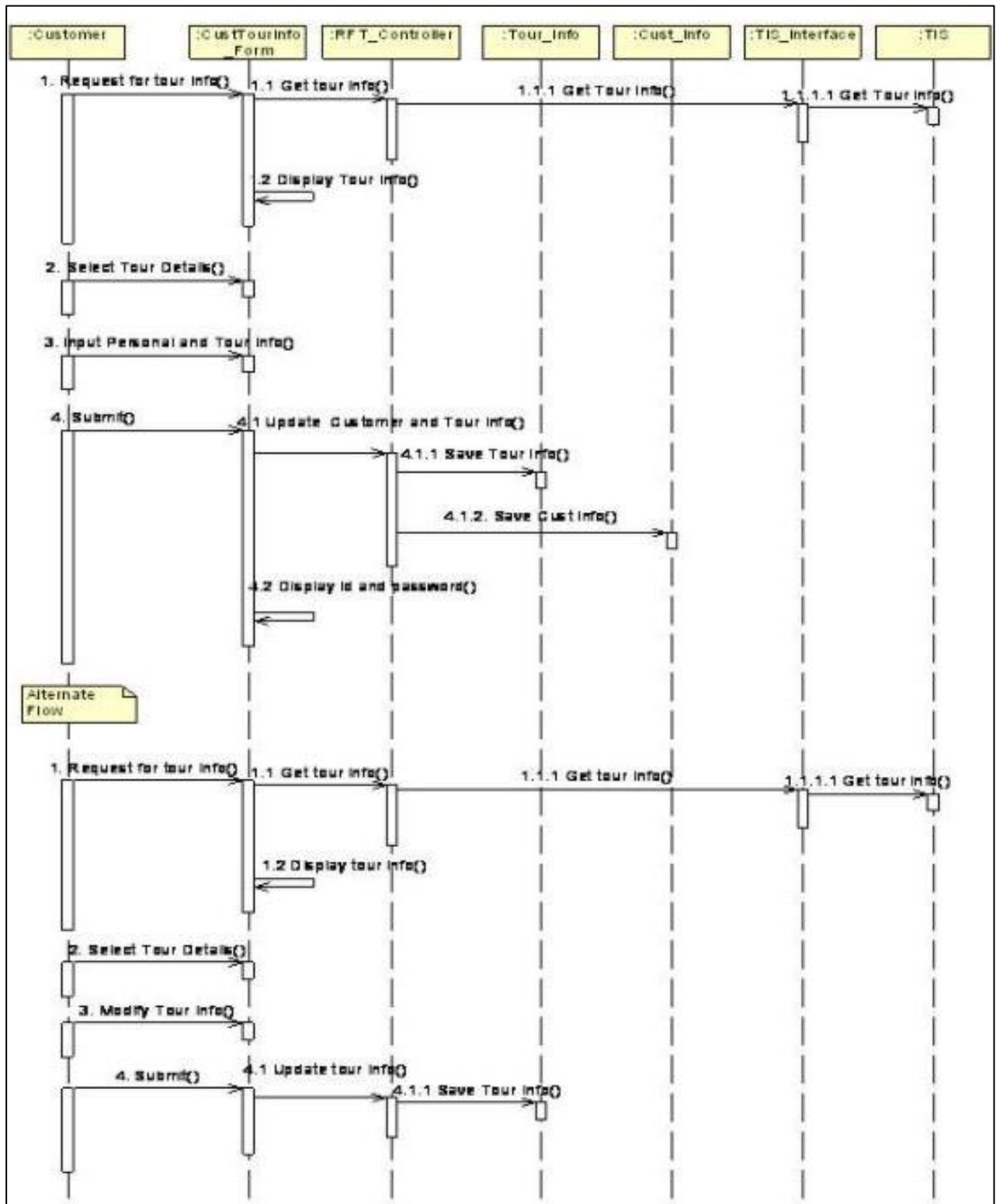




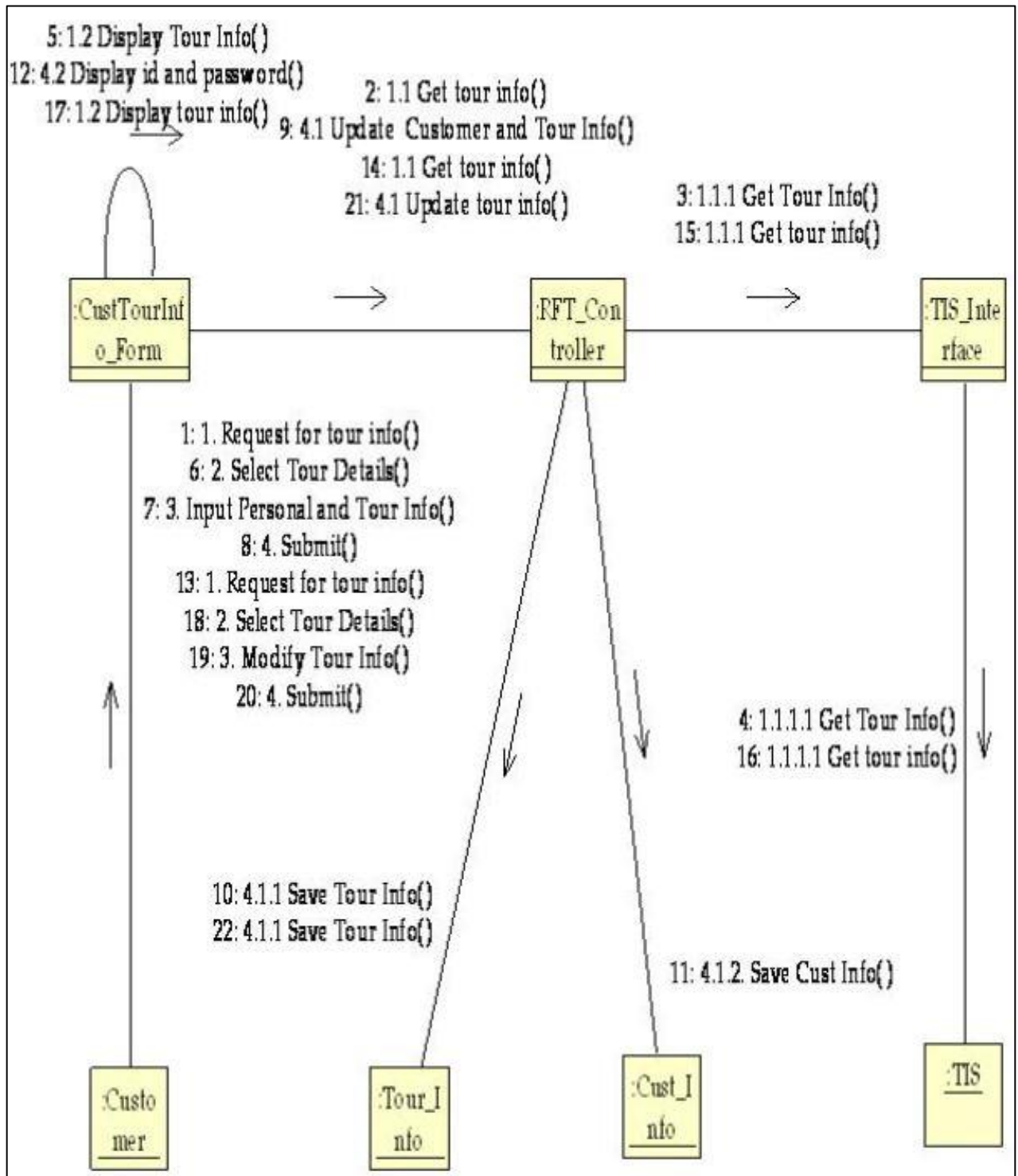
**Class Diagram Tour Management System:-**



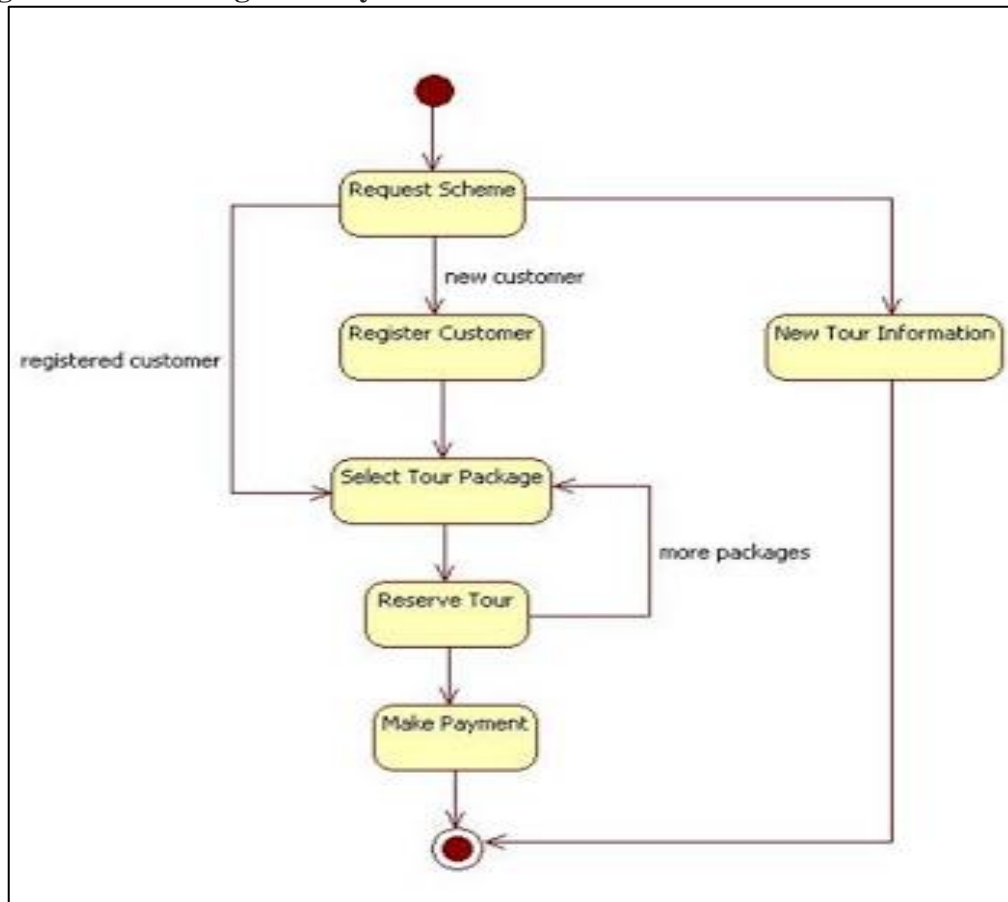
Sequence Diagram Tour Management System :



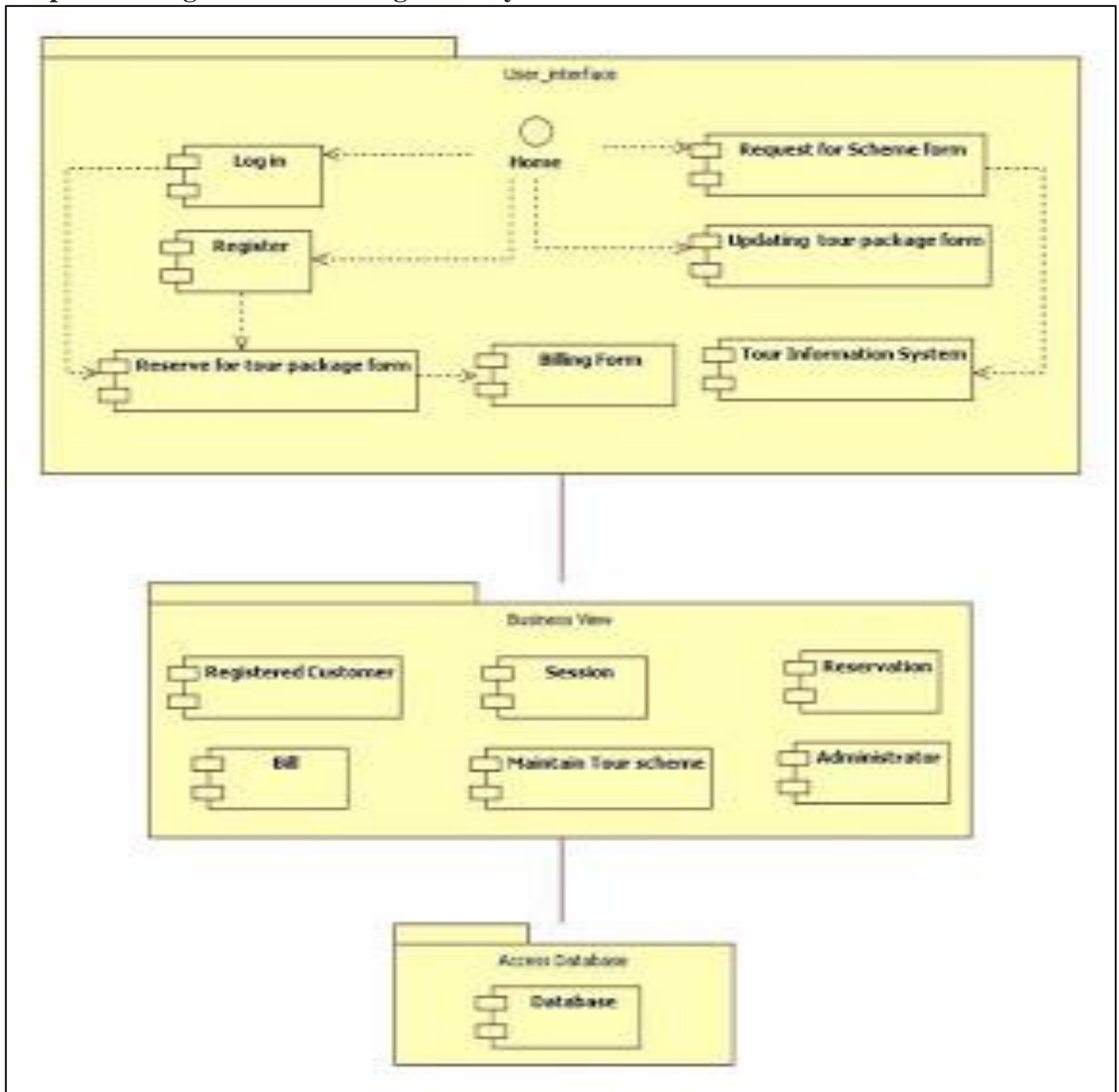
Collaboration Diagram Tour Management System :-



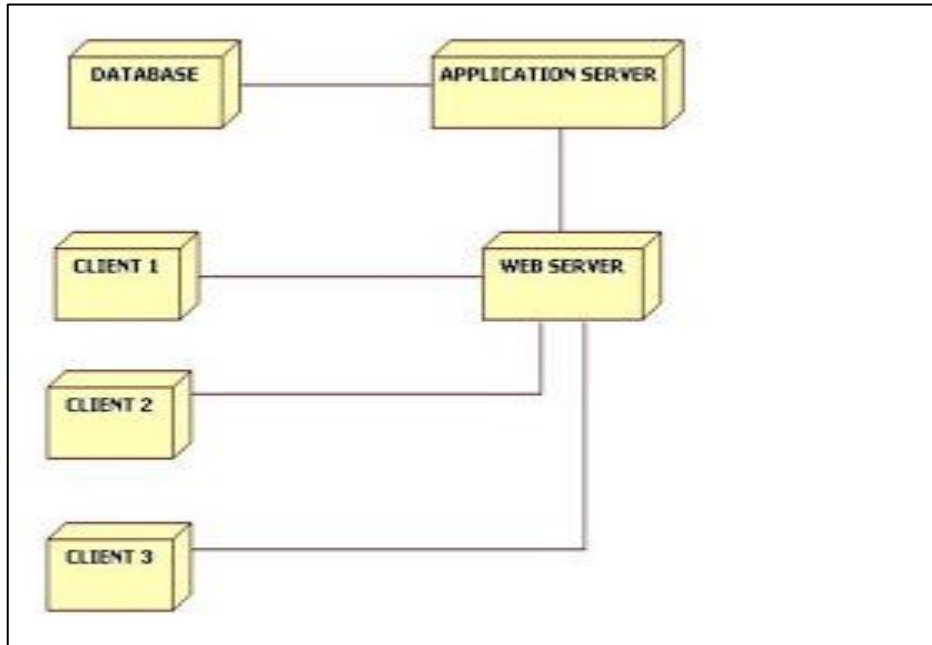
## State Diagram Tour Management System :-

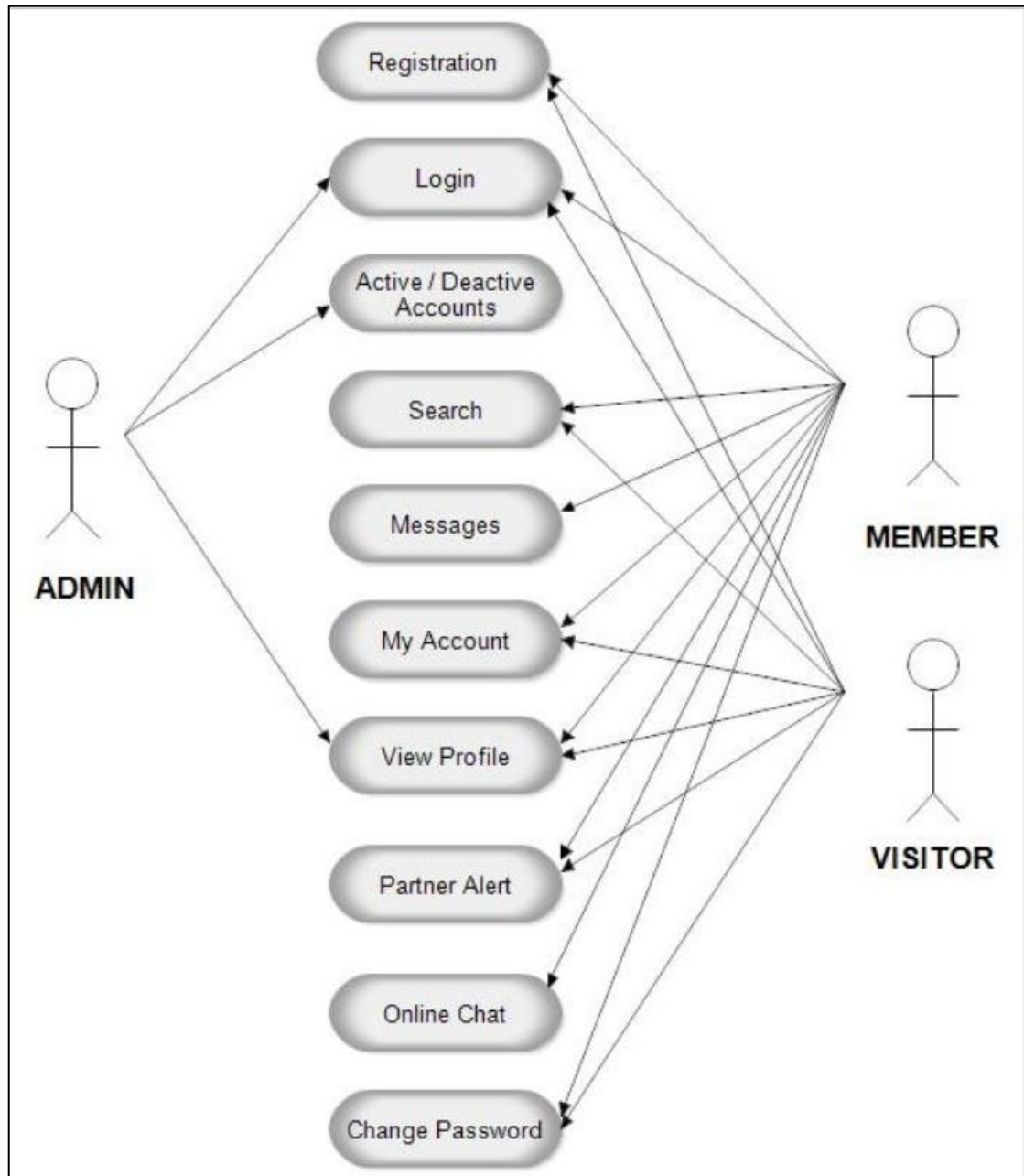


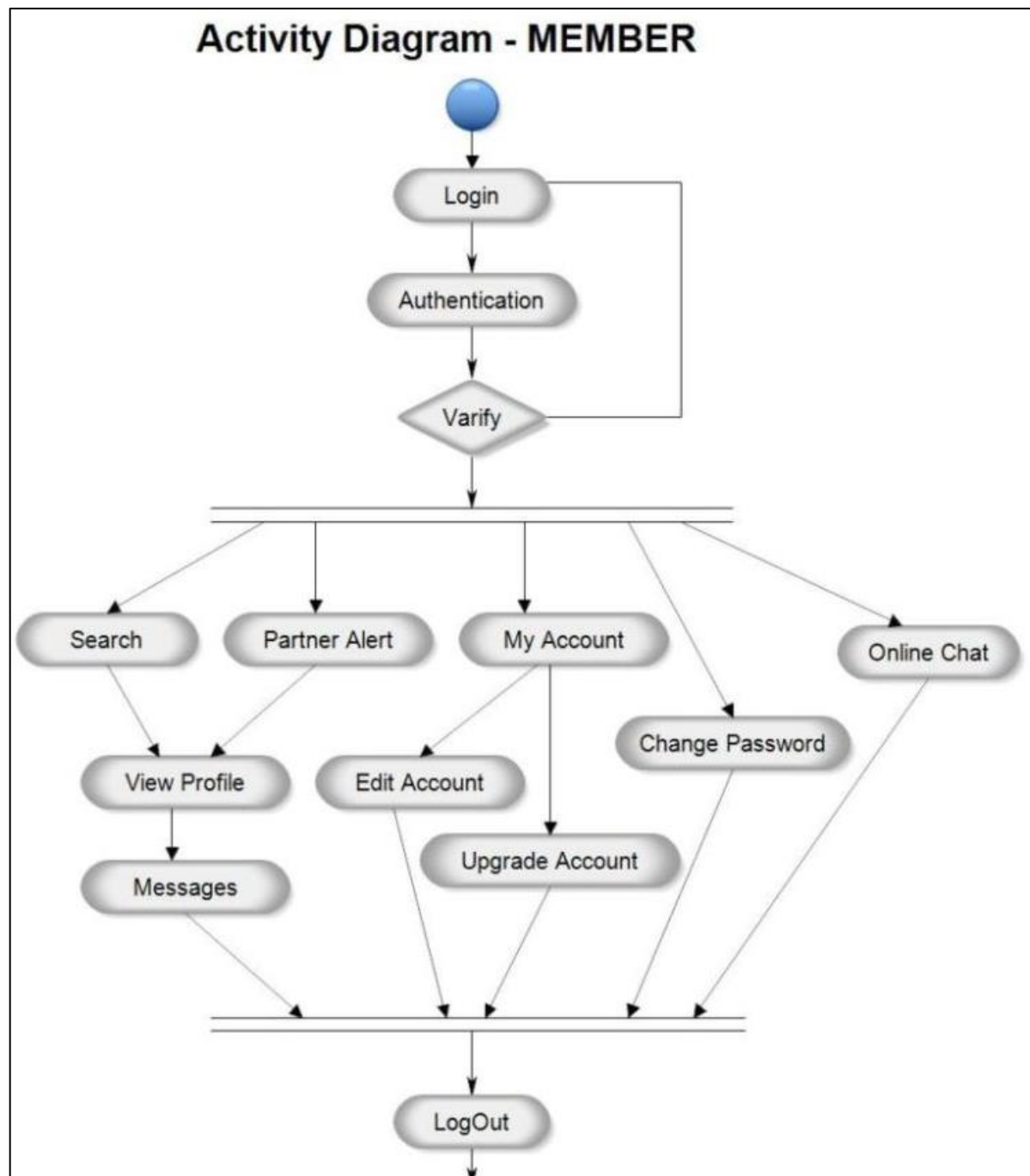
Component Diagram Tour Management System :-



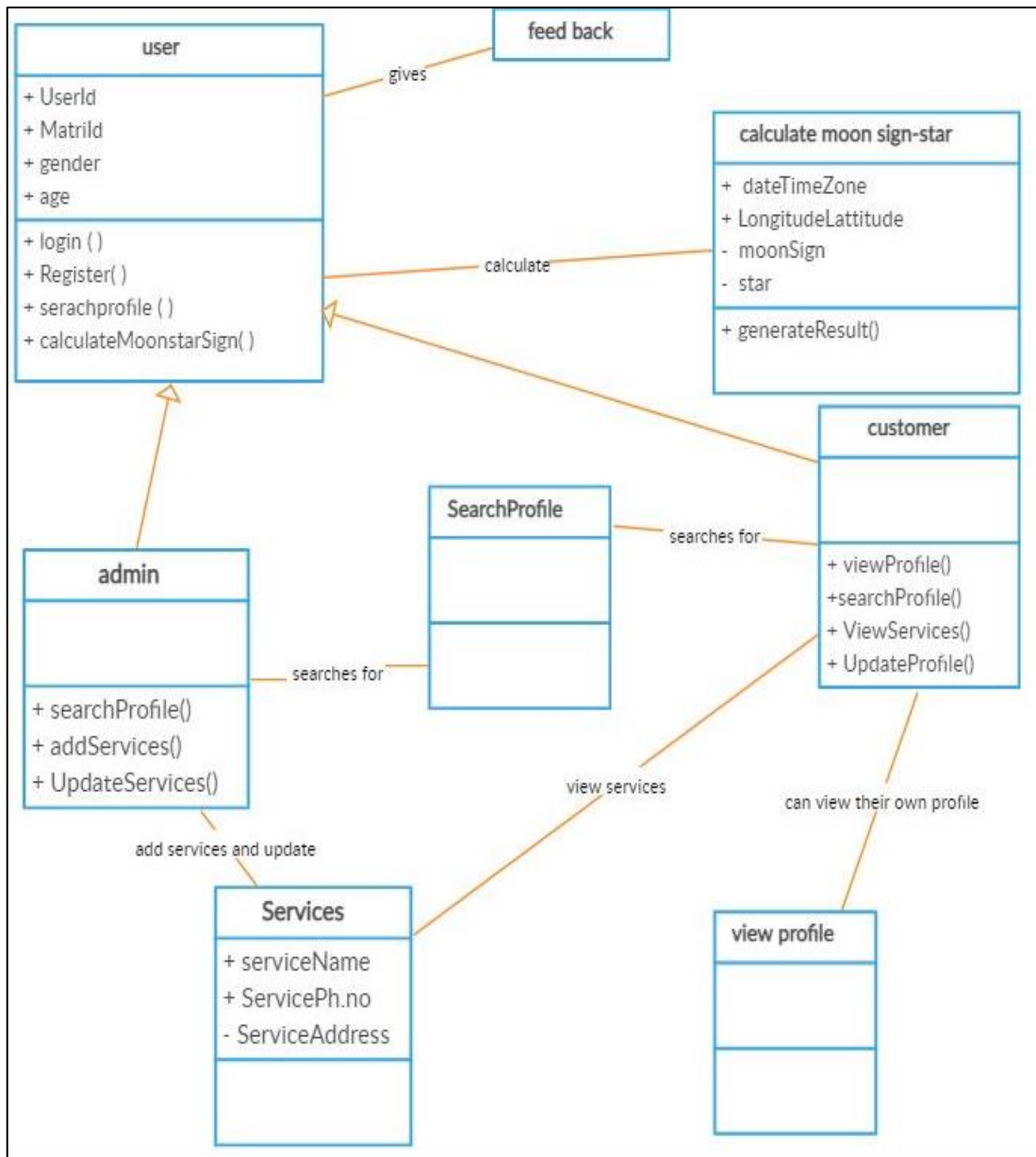
Deployment Diagram Tour Management System:-

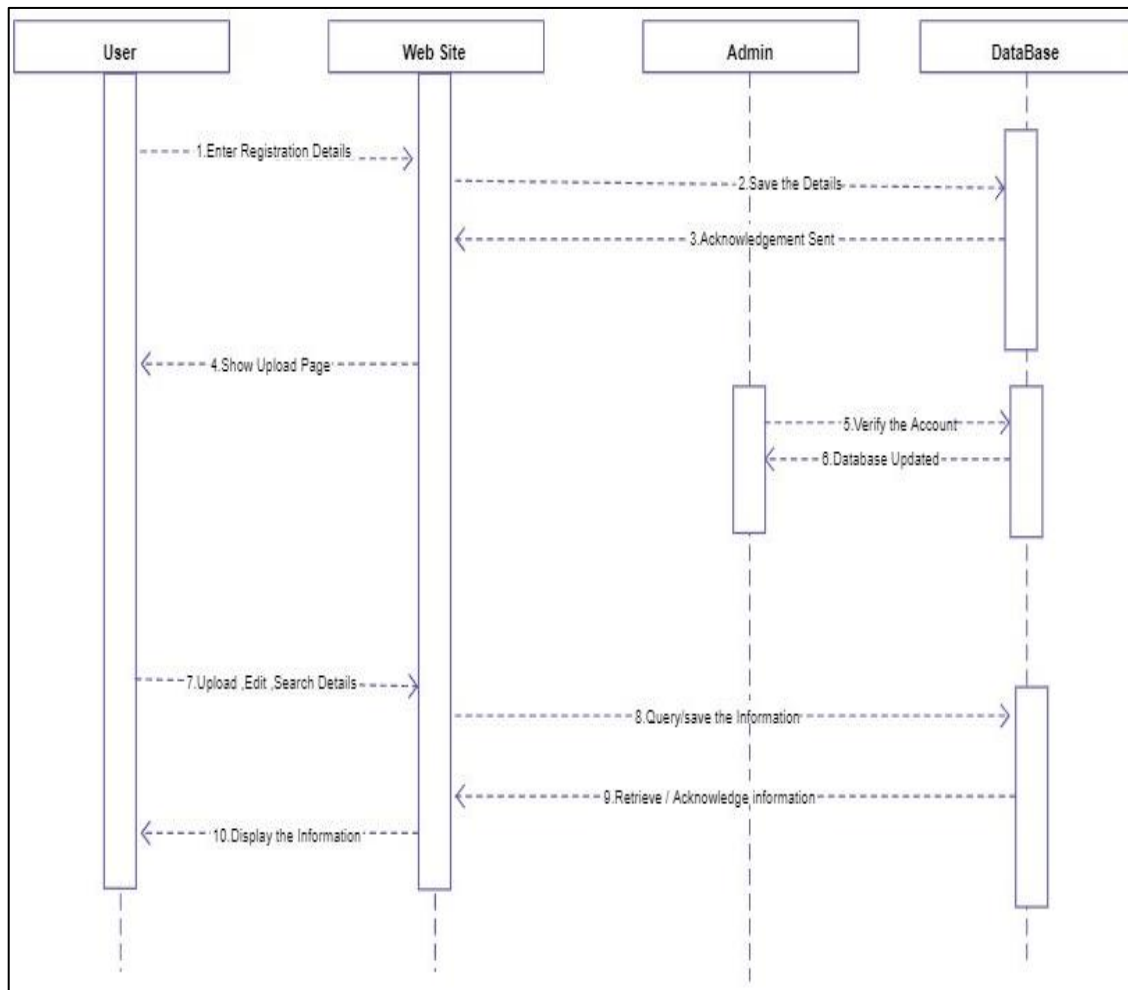


**PROGRAM 7: Social Networking ( LinkedIn, FaceBook, Shaadi.com, Bharat Matrimony, Tinder)****AIM: Social Networking ( LinkedIn, FaceBook, Shaadi.com, BharatMatrimony,Tinder)**



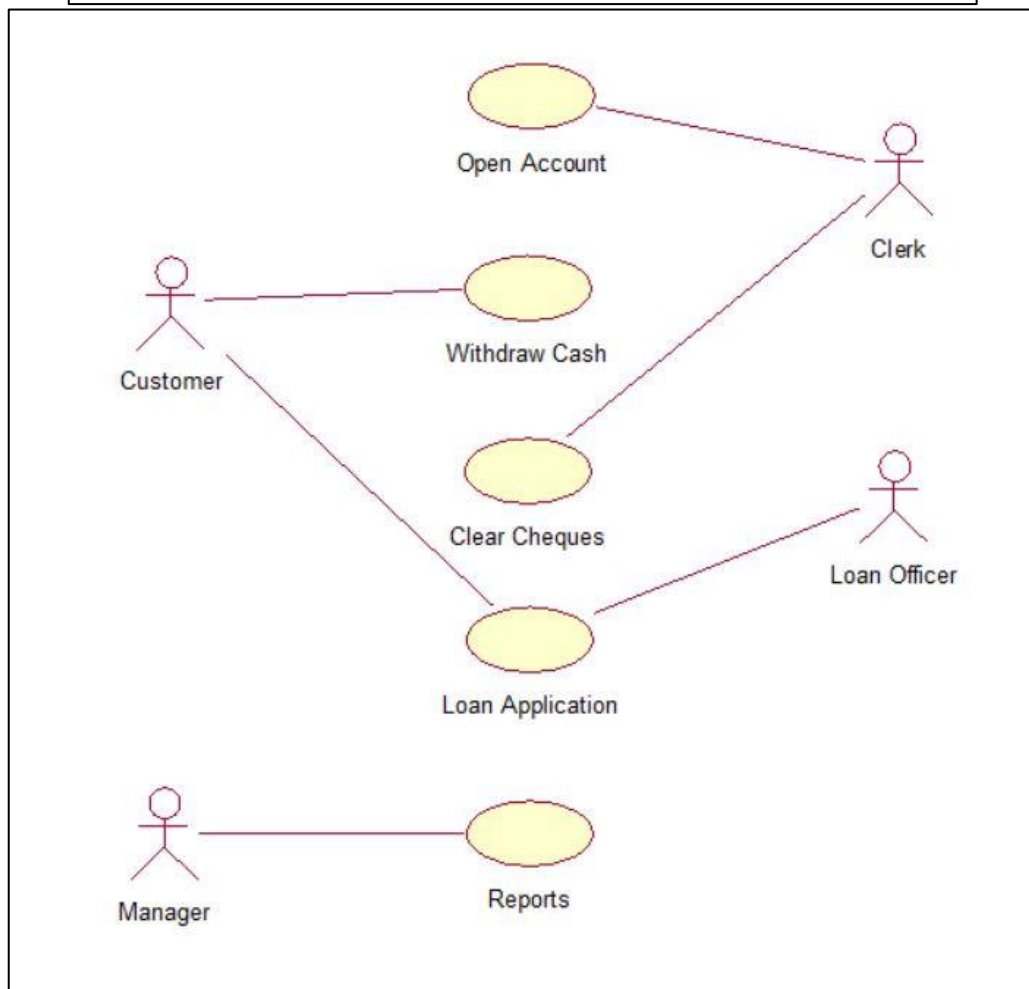
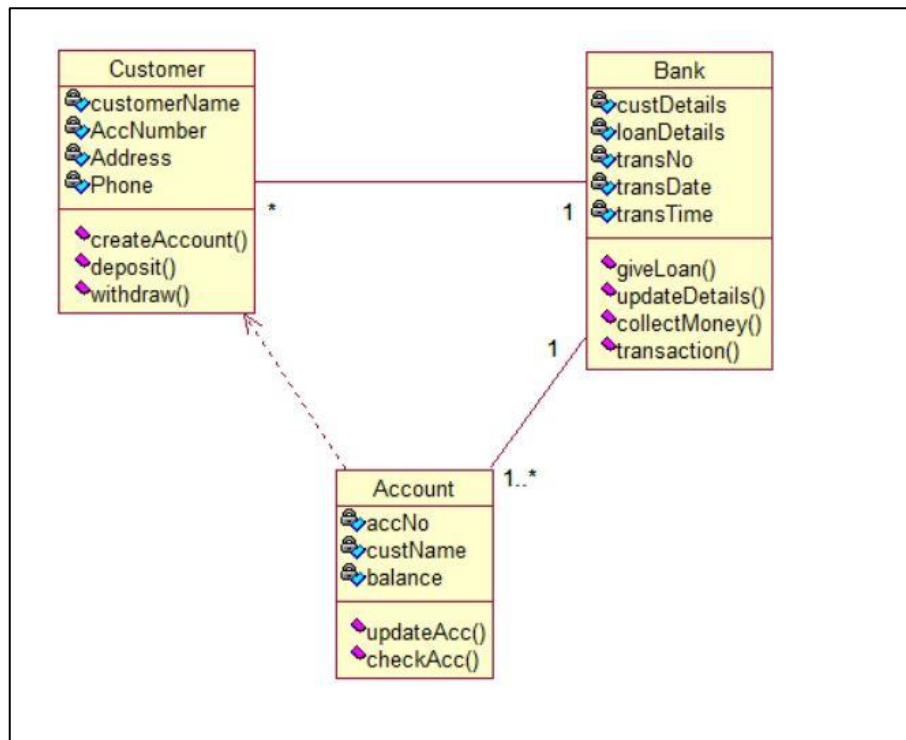


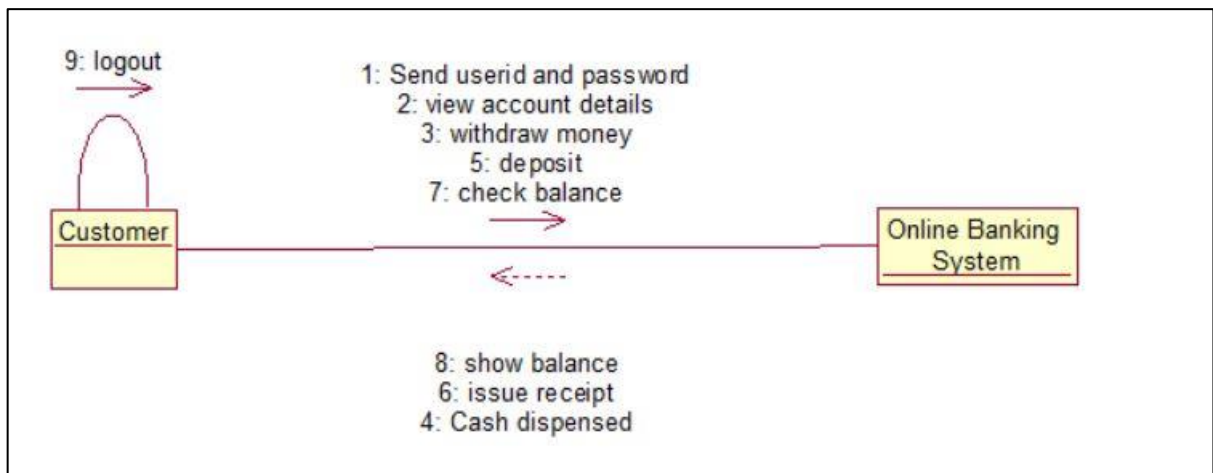
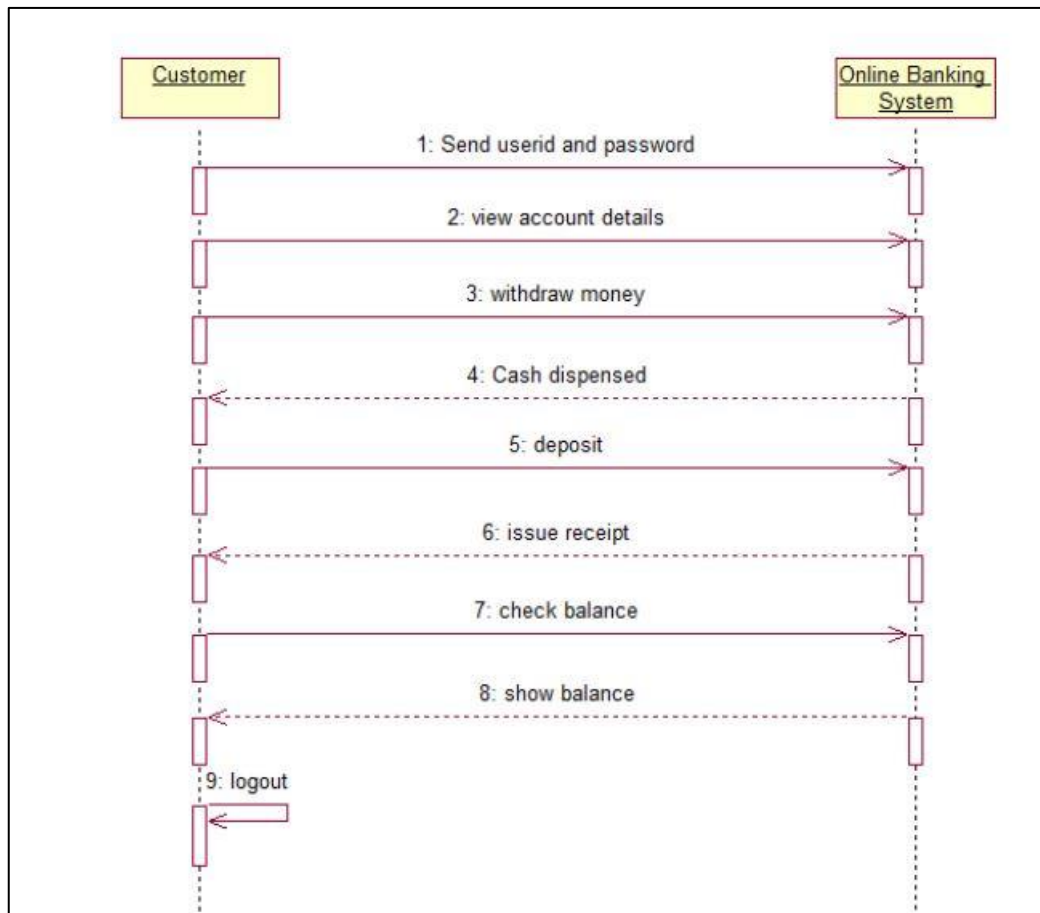


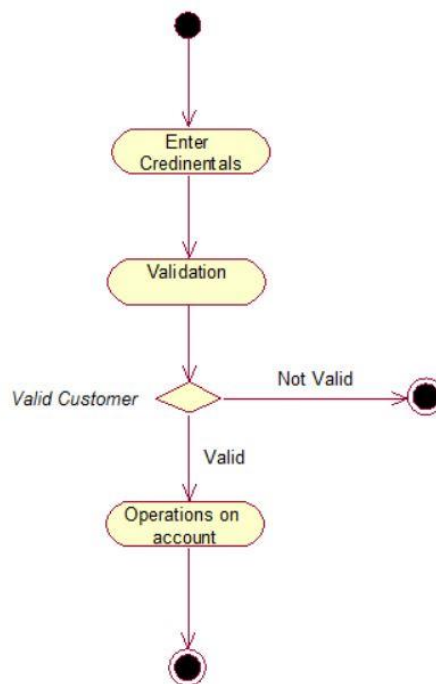
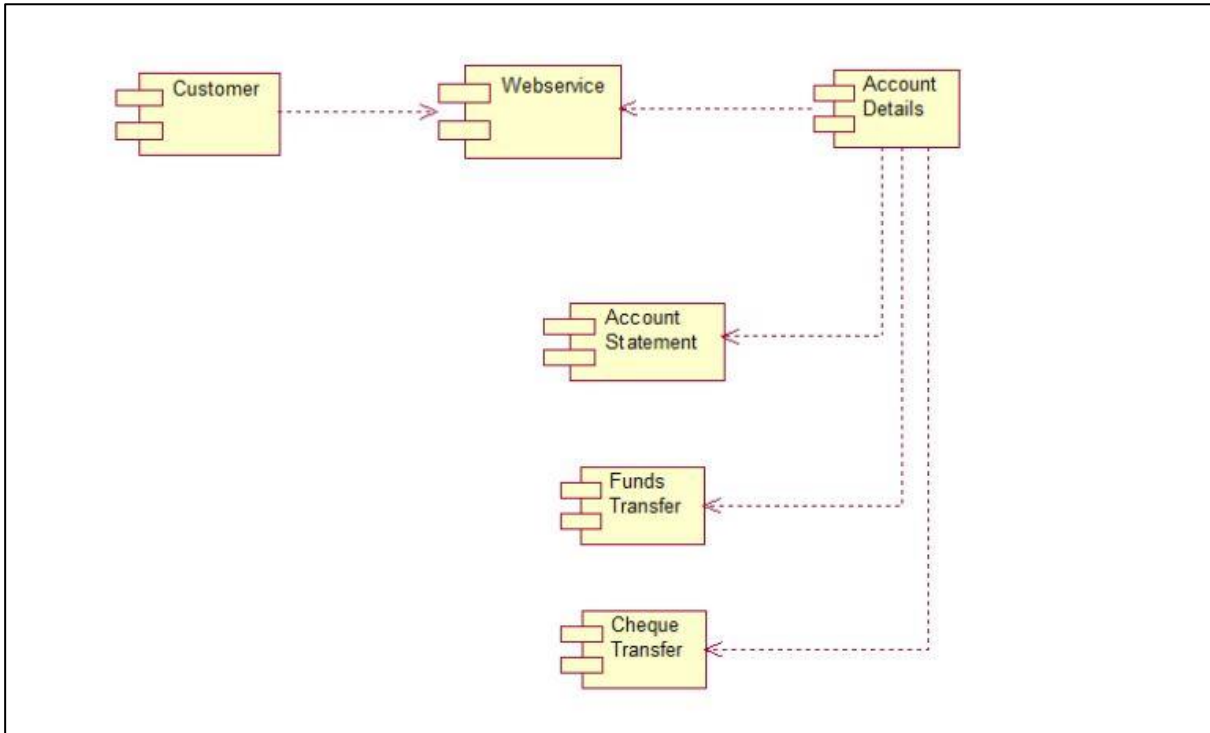
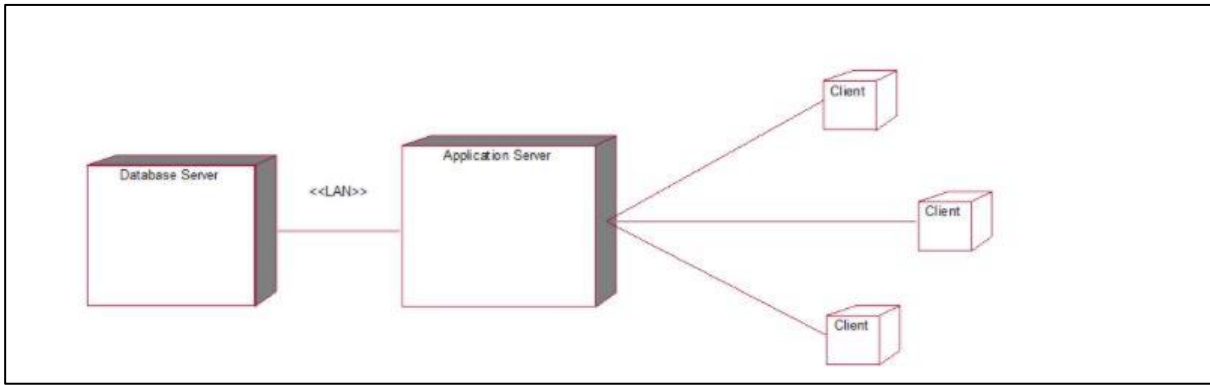


**PROGRAM 8 : Customer Support (Banking Ombudsman,Indian Consumer Complaints Forum)**

**AIM : Customer Support (Banking Ombudsman,Indian Consumer Complaints Forum)**







**PROGRAM 9 : Booking/Ticketing(Food:Zomato/Swiggy/BigBasket/Grofers/JioMart, Hotel:OYO/Trivago or Travel:{Cars:Uber/OLA/Zoom, Railways:IRCTC, Buses:OnlineTSRTC/RedBus/AbhiBus,Flights:MakeMyTrip/Goibibo, Ships:Lakport})**

**AIM : Booking/Ticketing(Food:Zomato/Swiggy/BigBasket/Grofers/JioMart, Hotel:OYO/Trivago or Travel:{Cars:Uber/OLA/Zoom, Railways:IRCTC, Buses:OnlineTSRTC/RedBus/AbhiBus,Flights:MakeMyTrip/Goibibo, Ships:Lakport})**

### **1. PROBLEM STATEMENT:**

This project is about online ticket reservation and consists of two modules. The reservation and the cancellation module. The reservation module allows the user to reserve tickets for a particular train on a particular date. If there is a ticket available, the users can know the vacancy details through the enquiry module. The cancellation module allows user to cancel the tickets for a particular date through reservation officer (system). This module performs status reveal before tickets are being reserved and after they get booked. All these modules together prove to be a flexible online reservation system and it provides complete flexibility to end users and it assumes the desired performance.

### **2. OVERALL DESCRIPTION:**

#### **2.1 MODULES:**

Login

Display train list

Search for train

Reservation

Cancellation

Train Status

#### **2.2 MODULE DELIVERABLES:**

##### **1. LOGIN**

Basic Flow

This use case starts when the passenger wishes to Login to the Online Ticket Reservation system

1. The System requests that the passenger enter his/her name and password
2. The passenger enters his/her name and password
3. The System validates the entered name and password and logs the passenger into the System

Alternative Flows: Invalid Name/Password

If, in the Basic flow, the passenger enters an invalid name and/or password, the system displays an error message. The passenger chooses to either return to the beginning of the Basic flow or cancel the login, at which point the use case ends.

Pre-Conditions: None

Post-Conditions: If the use case was successful, the passenger is now logged into the system. If not, the system State is unchanged.

## **2. Display Train List**

Basic Flow: This use case gives passenger information about each train namely train no, train name, Stations passes, Arrival Time, Departure Time etc

Alternative Flows: None

Pre-Conditions: None

Post-Conditions: If the use case was successful, the passenger information about each train namely train no, train name, Stations passes, Arrival Time, Departure Time etc

## **3. Search for Train**

Basic flow

The passenger can obtain train information either by entering train no or Source and Destination Station

1. If the passenger train no gives the information about train
2. If the passenger enter Source and Destination Station from list gives information about list of trains passing through station. From the list link will be provided to each train, which contains the information

Alternate flow: If the passenger enters an invalid train no then it gives error message invalid train no and asks the passenger to enter a valid train no.

Pre-Conditions: None

Post-Conditions: If the use case was successful, the passenger can able to view the list of trains.

## **4.Reservation**

Basic flow

1. The user reserves the ticket by giving following
  - a) Passenger name, Sex, Age, Address
  - b) Credit Card No, Bank Name
  - c) Class through passenger is going to travel i.e First class or Second class or AC
  - d) Train no and Train name, Date of Journey and number of tickets to be booked.
2. If the ticket is available in a train then the ticket will be issued with PNR No.else the ticket will be issued with a waiting list number.

Alternative flow: If the passenger gives an invalid credit card no or specified a bank where does have any account. Error message will be displayed.

Pre-Conditions: The passenger has to decide about the train he is going to travel.

Post-Conditions: If the use case was successful, the passenger will get the ticket.

## **5. Cancellation**

Basic flow

This use case used by passenger to cancel the ticket, which he/she booked earlier by Entering PNR No. The cancellation has been done reallocating the tickets allotted to the Passenger.

Alternate flow: If the Passenger had entered invalid PNR No then has been asked to enter valid PNR No.

Pre-Conditions: The Passenger had reserved tickets in a train.

Post-Conditions: If the use case was successful, the passenger can cancel the ticket.

**6. Ticket Status**

Basic flow

1. The passenger should give PNR No to know the status of ticket, which he/she booked earlier.
2. If the PNR No is valid, the status of the ticket will be displayed.

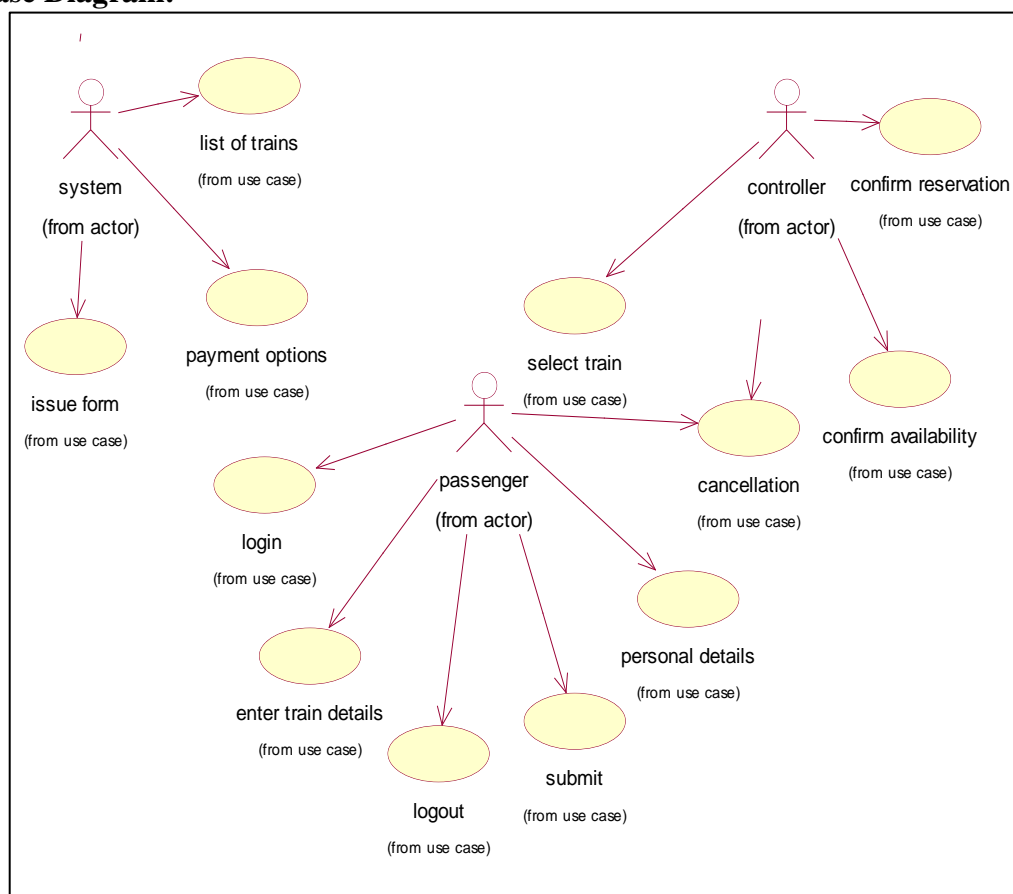
Alternate flow: If passenger had entered an invalid no or PNR NO, which does not exists then error Message will be displayed.

Pre-Conditions: The Passenger had reserved tickets in a train.

Post-Conditions: If the use case was successful, the passenger can view status of the ticket.

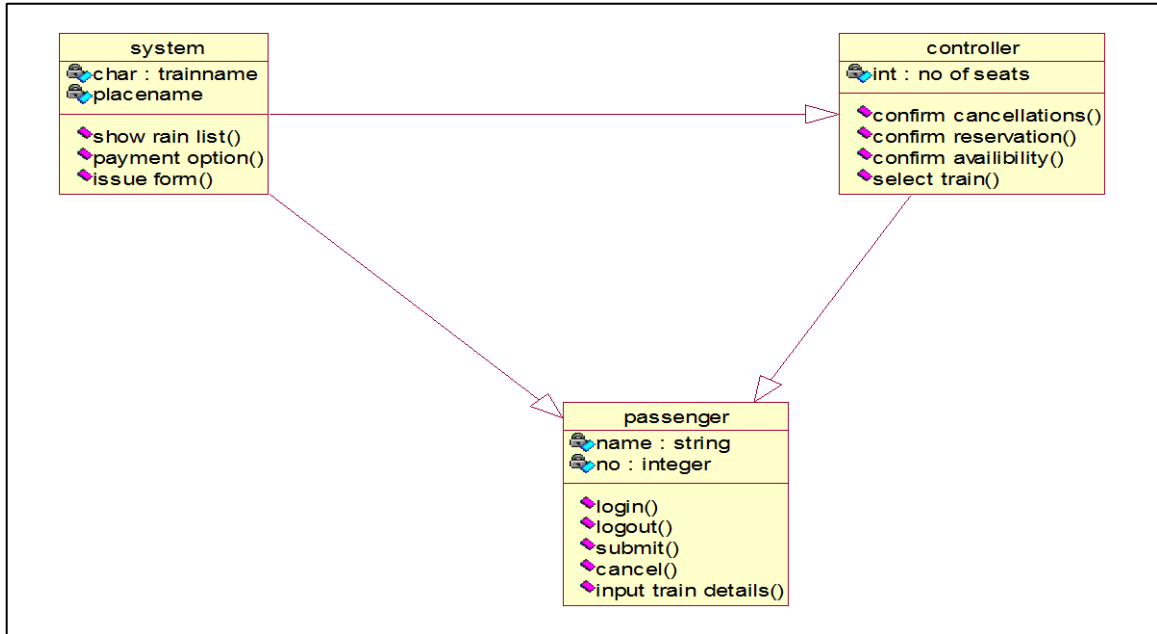
**3. UML DIAGRAMS:**

**3.1. Usecase Diagram:**

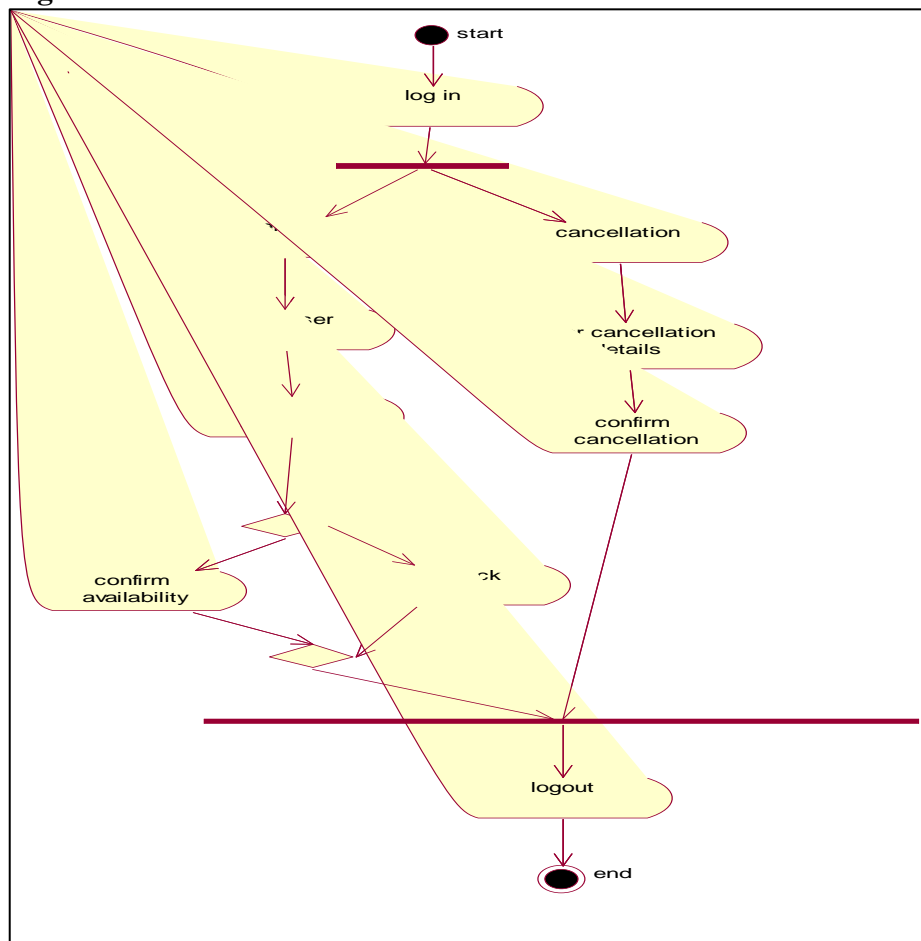




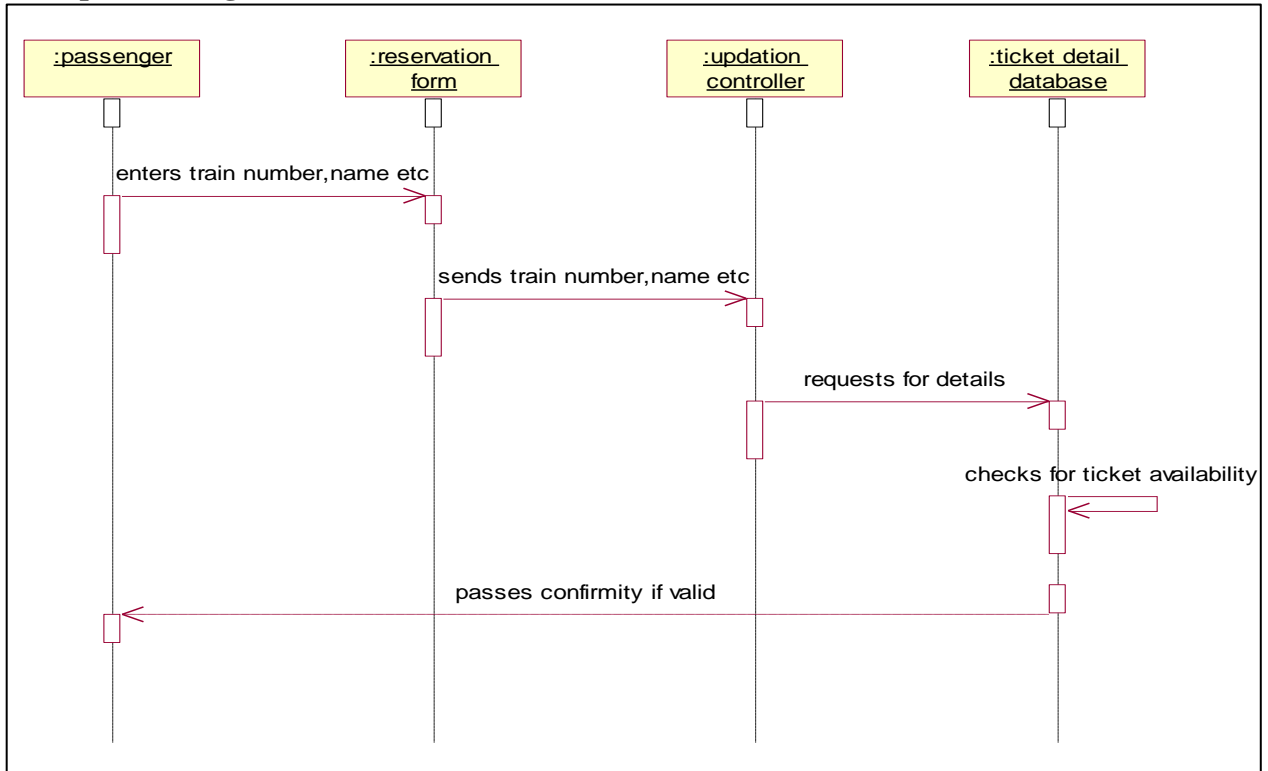
3.2 Class Diagram:



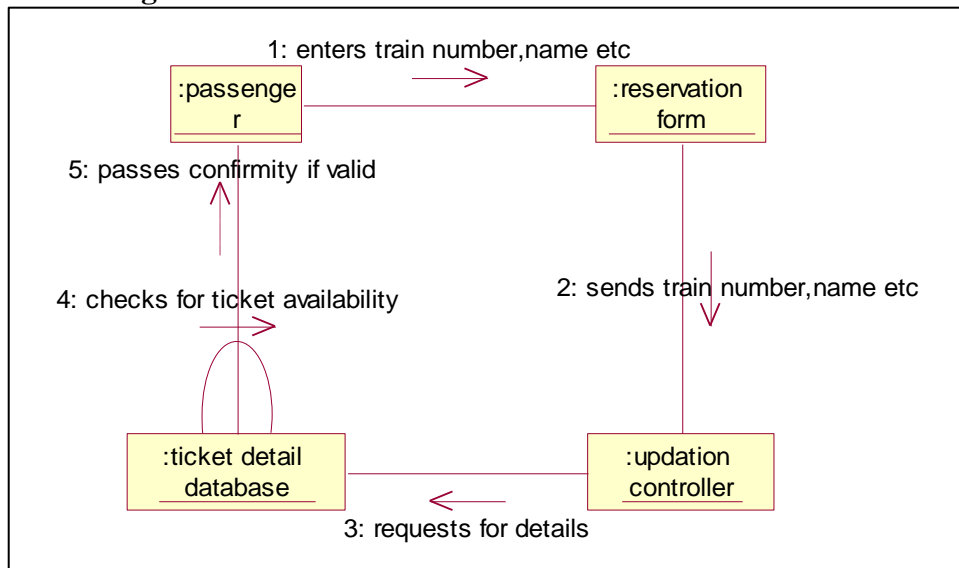
3.3 Activity Diagram:



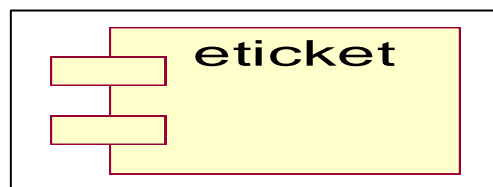
3.4 Sequence Diagram:

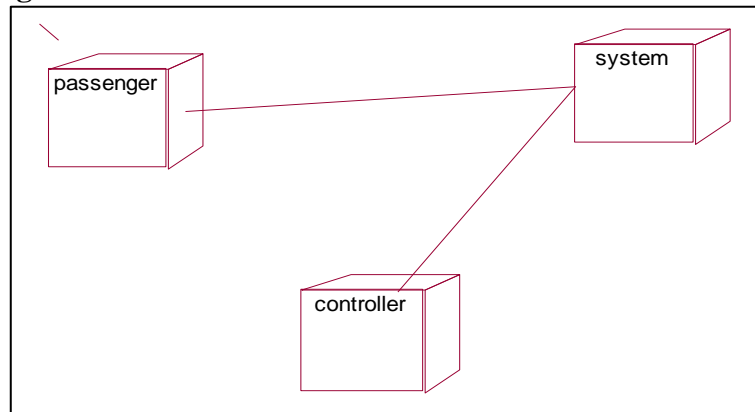


3.5 Collaboration Diagram



3.6 Component Diagram:



**3.7 Deployment Diagram:****4. DATABASE DESIGN**

Database name: Rail

Table name :student name

Field name	Datatype
Name	Text(50)
Place	Integer
Mobile number	integer

Table name: reservation

Field name	Datatype
Name	Text
Age	Integer
Place	Text
Train and number	Text
Seat Selection	Text
Sex	Text
Date	Text
Time	Text
From	Text
To	Text
Food	Text
Class	Text

Table name : reservation 2

Field name	Data type
Adhar no	Text
Pan card number	Text
e-mail id	Text
Mobile number	Text

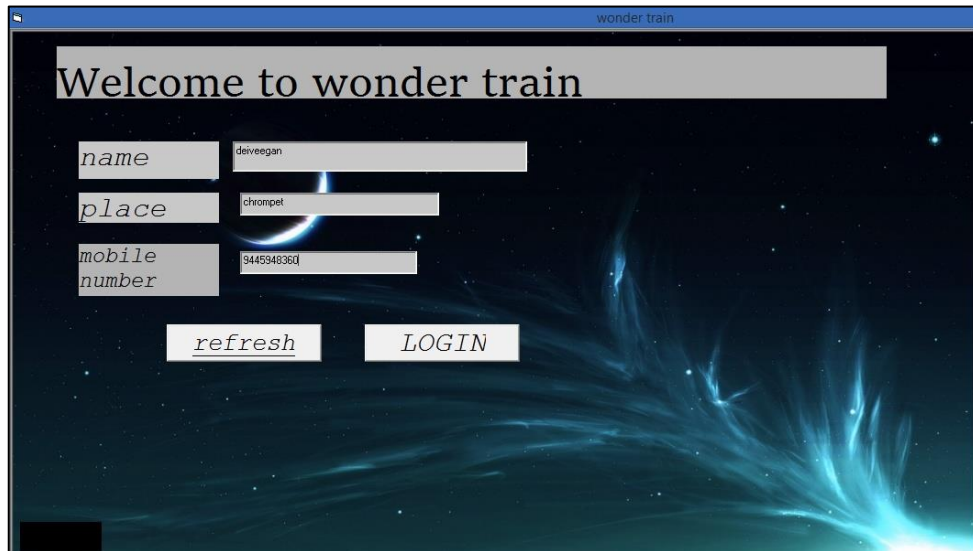
Table name: cancellation

Field name	Data type
Train name & number	Text
Name	Text
Pnr number	Text
e-mail ID	Text
Phone number	Text

Table name: ticket status

Field name	Data type
Train name & number	Text
Name	Text
Pnr number	Text
e-mail ID	Text
Phone number	Text

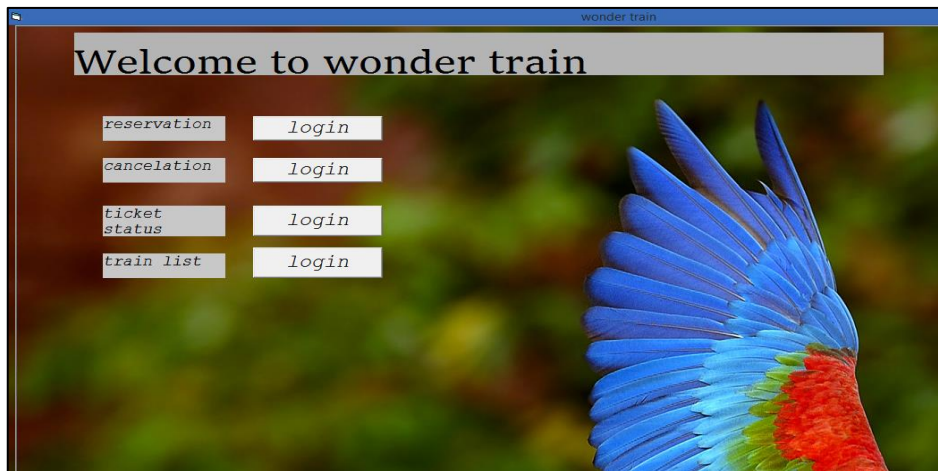
## 5. IMPLEMENTATION



### Coding:

```
Private Sub Command1_Click()
Form1.Show
Data1.Recordset.AddNew
Data1.Recordset.Fields("name") = Text1.Text
Data1.Recordset.Fields("place") = Text2.Text
Data1.Recordset.Fields("mobile number") = Text3.Text
Data1.Recordset.Update
End Sub
```

```
Private Sub Command2_Click()
Dim ctr As Control
For Each ctr In Me.Controls
If TypeOf ctr Is TextBox Then
ctr.Text = Empty
End If
Next ctr
End Sub
```

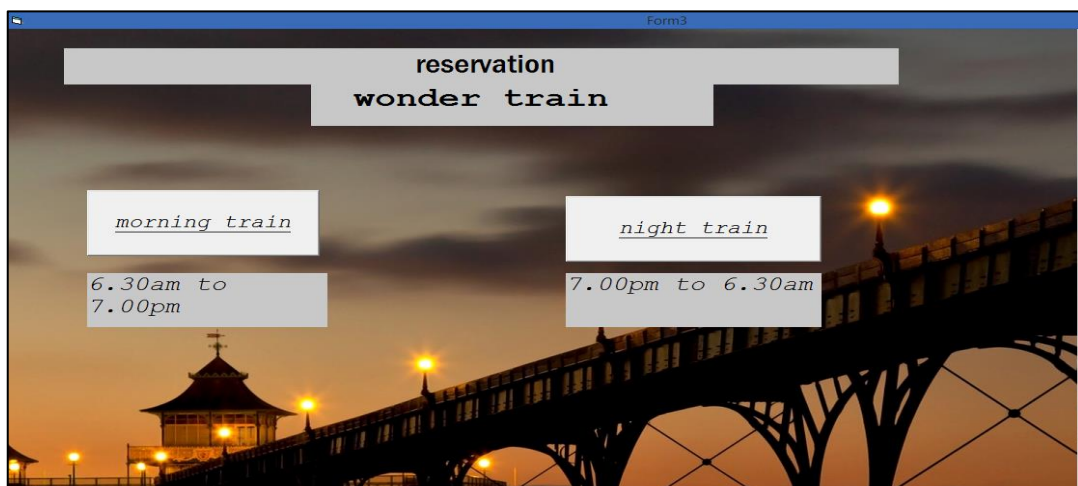
**coding**

```
Private Sub Command1_Click()
Form3.Show
End Sub
```

```
Private Sub Command2_Click()
Form7.Show
End Sub
```

```
Private Sub Command3_Click()
Form9.Show
End Sub
```

```
Private Sub Command4_Click()
Form12.Show
End Sub
```

**coding**

```
Private Sub Command1_Click()
Form2.Show
End Sub
```

```
Private Sub Command2_Click()
Form6.Show
End Sub
```

The screenshot shows a web form titled "wonder train reservation (morning)". The form is set against a dark, starry background. It contains several input fields and radio buttons. The fields are labeled as follows: "name" (text input with "k.deveegan"), "age" (text input with "18"), "place" (text input with "chromapel"), "train name and number" (text input with "123456789"), "seat selection" (radio buttons for "upper" and "lower", with "upper" selected), "sex" (radio buttons for "male" and "female", with "male" selected), "date" (text input with "24/5/2016"), "time" (text input with "12:00"), "from" (text input with "chennai"), "to" (text input with "delhi"), "food" (radio buttons for "veg" and "non-veg", with "veg" selected), and "class" (radio buttons for "1 class" and "2 class", with "1 class" selected). At the bottom right, there are two buttons: "refresh" and "next".

**coding**

```

Private Sub Command1_Click()
Form4.Show
Data1.Recordset.AddNew
Data1.Recordset.Fields("name") = Text1.Text
Data1.Recordset.Fields("age") = Text2.Text
Data1.Recordset.Fields("place") = Text3.Text
Data1.Recordset.Fields("train name and number") = Text8.Text
If Option1.Value = True Then
Data1.Recordset.Fields("seat selection") = "upper"
Else
Data1.Recordset.Fields("seat selection") = "lower"
End If
If Option6.Value = True Then
Data1.Recordset.Fields("sex") = "male"
Else
Data1.Recordset.Fields("sex") = "femal"
End If
If Option10.Value = True Then
Data1.Recordset.Fields("food") = "veg"
Else
Data1.Recordset.Fields("food") = "non-veg"
End If
If Option3.Value = True Then
Data1.Recordset.Fields("class") = "1 class"
Else
Data1.Recordset.Fields("seat selection") = "2 class"
End If
Data1.Recordset.Fields("date") = Text4.Text
Data1.Recordset.Fields("time") = Text5.Text
Data1.Recordset.Fields("from") = Text6.Text
Data1.Recordset.Fields("to") = Text7.Text
Data1.Recordset.Update
End Sub

```

```

Private Sub Command4_Click()
Dim ctr As Control
For Each ctr In Me.Controls
If TypeOf ctr Is TextBox Then

```

```

    ctr.Text = Empty
End If
Next
End Sub

```

### coding

```

Private Sub Command2_Click()
Form5.Show
Data1.Recordset.AddNew
Data1.Recordset.Fields("adhar no") = Text1.Text
Data1.Recordset.Fields("pan card number") = Text2.Text
Data1.Recordset.Fields("e-mail ID") = Text3.Text
Data1.Recordset.Fields("mobile number") = Text4.Text
Data1.Recordset.Update
End Sub

```

```

Private Sub Command4_Click()
Dim ctr As Control
For Each ctr In Me.Controls
    If TypeOf ctr Is TextBox Then
        ctr.Text = Empty
    End If
Next
End Sub

```



**coding**

```
Private Sub Command1_Click()
train.Show
End Sub
```

**coding**

```
Private Sub Command1_Click()
Form4.Show
Data1.Recordset.AddNew
Data1.Recordset.Fields("name") = Text1.Text
Data1.Recordset.Fields("age") = Text2.Text
Data1.Recordset.Fields("place") = Text3.Text
Data1.Recordset.Fields("train name and number") = Text8.Text
If Option1.Value = True Then
Data1.Recordset.Fields("seat selection") = "upper"
Else
Data1.Recordset.Fields("seat selection") = "lower"
End If
If Option6.Value = True Then
Data1.Recordset.Fields("sex") = "male"
Else
Data1.Recordset.Fields("sex") = "femal"
End If
If Option10.Value = True Then
Data1.Recordset.Fields("food") = "veg"
Else
Data1.Recordset.Fields("food") = "non-veg"
End If
If Option3.Value = True Then
Data1.Recordset.Fields("class") = "1 class"
Else
Data1.Recordset.Fields("seat selection") = "2 class"
End If
Data1.Recordset.Fields("date") = Text4.Text
Data1.Recordset.Fields("time") = Text5.Text
Data1.Recordset.Fields("from") = Text6.Text
Data1.Recordset.Fields("to") = Text7.Text
Data1.Recordset.Update
```



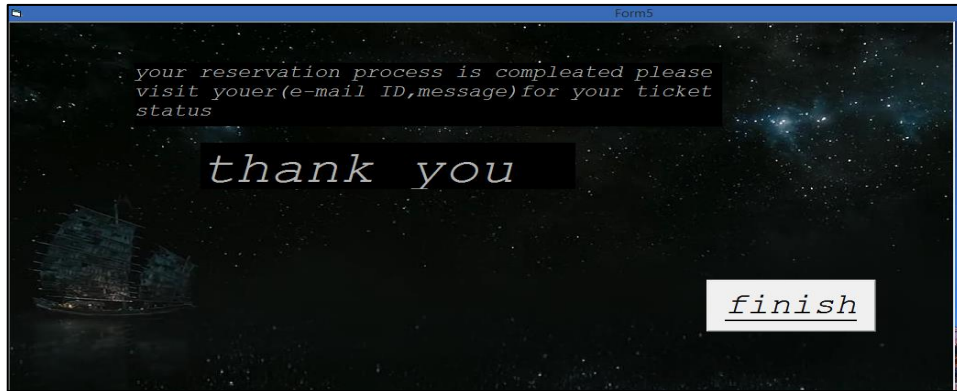
End Sub

```
Private Sub Command4_Click()
Dim ctr As Control
For Each ctr In Me.Controls
    If TypeOf ctr Is TextBox Then
        ctr.Text = Empty
    End If
Next
End Sub
```

### coding

```
Private Sub Command2_Click()
Form5.Show
Data1.Recordset.AddNew
Data1.Recordset.Fields("adhar no") = Text1.Text
Data1.Recordset.Fields("pan card number") = Text2.Text
Data1.Recordset.Fields("e-mail ID") = Text3.Text
Data1.Recordset.Fields("mobile number") = Text4.Text
Data1.Recordset.Update
End Sub
```

```
Private Sub Command4_Click()
Dim ctr As Control
For Each ctr In Me.Controls
    If TypeOf ctr Is TextBox Then
        ctr.Text = Empty
    End If
Next
End Sub
```



**coding**

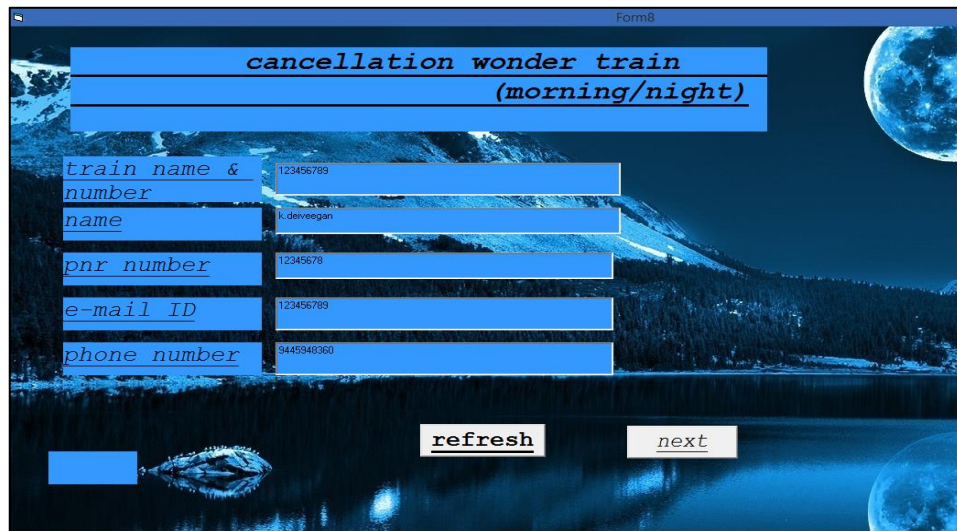
```
Private Sub Command1_Click()
train.Show
End Sub
```



**coding**

```
Private Sub Command1_Click()
Form8.Show
End Sub
```

```
Private Sub Command2_Click()
Form8.Show
End Sub
```



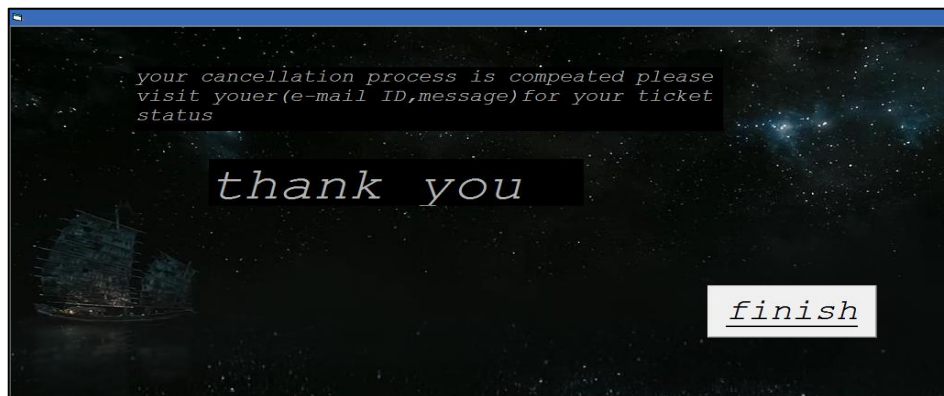
**coding**

```
Private Sub Command2_Click()
```

```

Form10.Show
Data1.Recordset.AddNew
Data1.Recordset.Fields("train name & number") = Text1.Text
Data1.Recordset.Fields("name") = Text2.Text
Data1.Recordset.Fields("pnr number") = Text3.Text
Data1.Recordset.Fields("e-mail ID") = Text4.Text
Data1.Recordset.Fields("phone number") = Text5.Text
Data1.Recordset.Update
End Sub
Private Sub Command4_Click()
Dim ctr As Control
For Each ctr In Me.Controls
    If TypeOf ctr Is TextBox Then
        ctr.Text = Empty
    End If
Next
End Sub

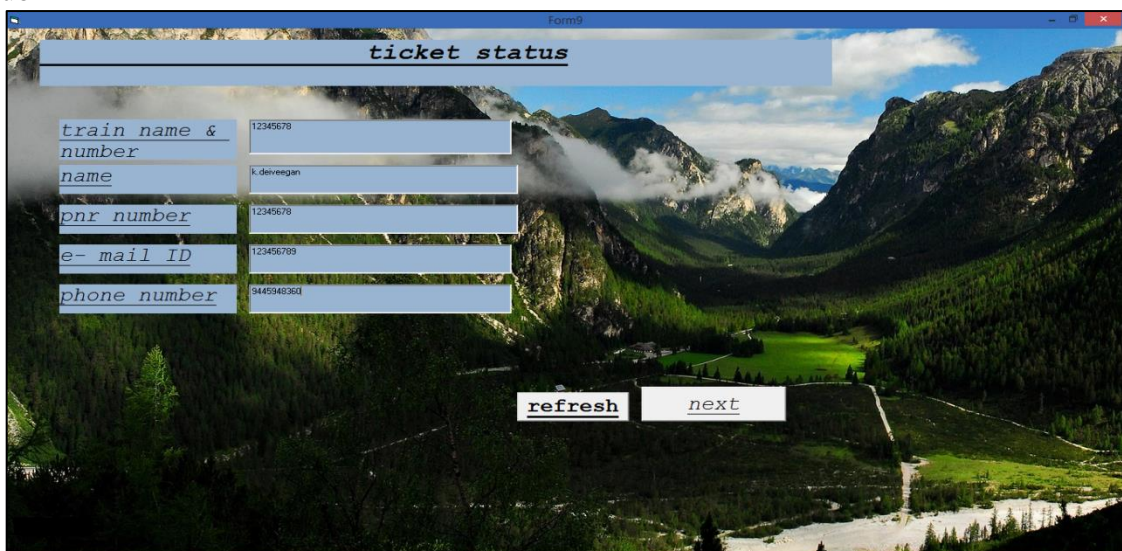
```

**coding**

```

Private Sub Command1_Click()
train.Show
End Sub

```

**coding**

```

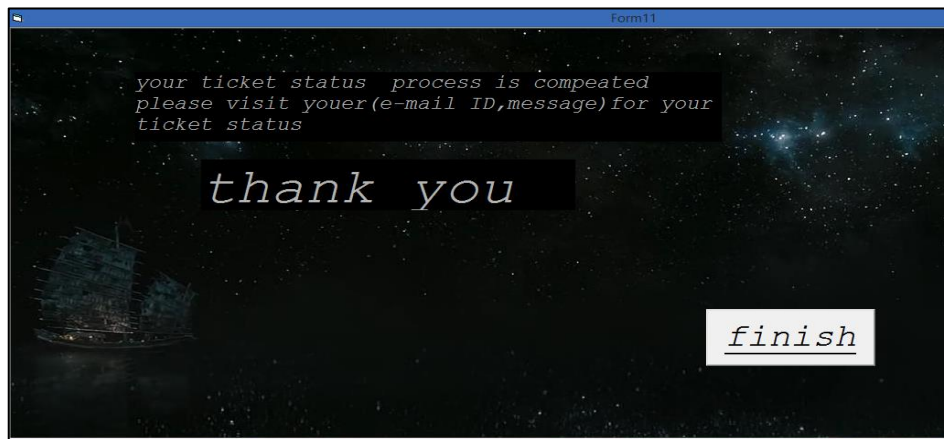
Private Sub Command2_Click()
Form11.Show
Data1.Recordset.AddNew
Data1.Recordset.Fields("train name & number") = Text1.Text
Data1.Recordset.Fields("name") = Text2.Text
Data1.Recordset.Fields("pnr number") = Text3.Text

```



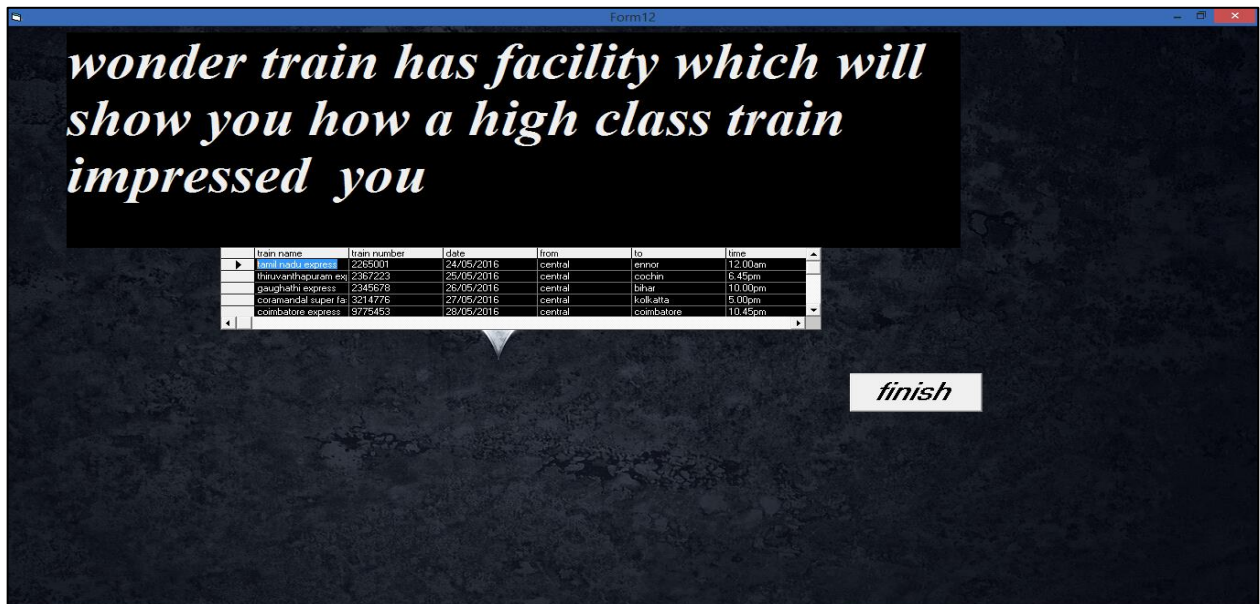
```
Data1.Recordset.Fields("e-mail ID") = Text4.Text
Data1.Recordset.Fields("phone number") = Text5.Text
Data1.Recordset.Update
End Sub
```

```
Private Sub Command4_Click()
Dim ctr As Control
For Each ctr In Me.Controls
    If TypeOf ctr Is TextBox Then
        ctr.Text = Empty
    End If
Next ctr
End Sub
```



### Coding v bn...

```
Private Sub Command1_Click()
train.Show
End Sub
```



### coding

```
Private Sub Command1_Click()
train.Show
End Sub
```

**6.TESTING:**

Test case ID: Test_01					
Test priority (Low/Medium/High):Medium					
Module name: login					
Test title :verify login with valid username and password					
Precondition: user has invalid username and password					
S.NO	TEST STEPS	EXPECTED RESULTS	ACTUAL RESULTS	STATUS	NOTES
1	Provide valid User name	User should Be able to login	The user is able to move to next Entry	Success	-
2	Provide valid password	User should be Able to Login	The user is able To login Successfully	Success	Incase of wrong Password was given an error Message box was displayed
3	Click login	User should be able to navigate to next page after validation	User name and password is validated and next page is displayed	Success	Incase user gives wrong entry the sign in page remains active
4	Click signup	User should be able to navigate to next page where user enters his credentials	User navigates to the signup page where his user name and password is validated	success	-

**RESULT:**The online ticket reservation system was designed and implemented successfully.

**PROGRAM II : REVERSE ENGINEERING**

Software Reverse Engineering is a process of recovering the design, requirement specifications and functions of a product from an analysis of its code. It builds a program database and generates information from this. The purpose of reverse engineering is to facilitate the maintenance work by improving the understandability of a system and to produce the necessary documents for a legacy system.

Reverse Engineering Goals:

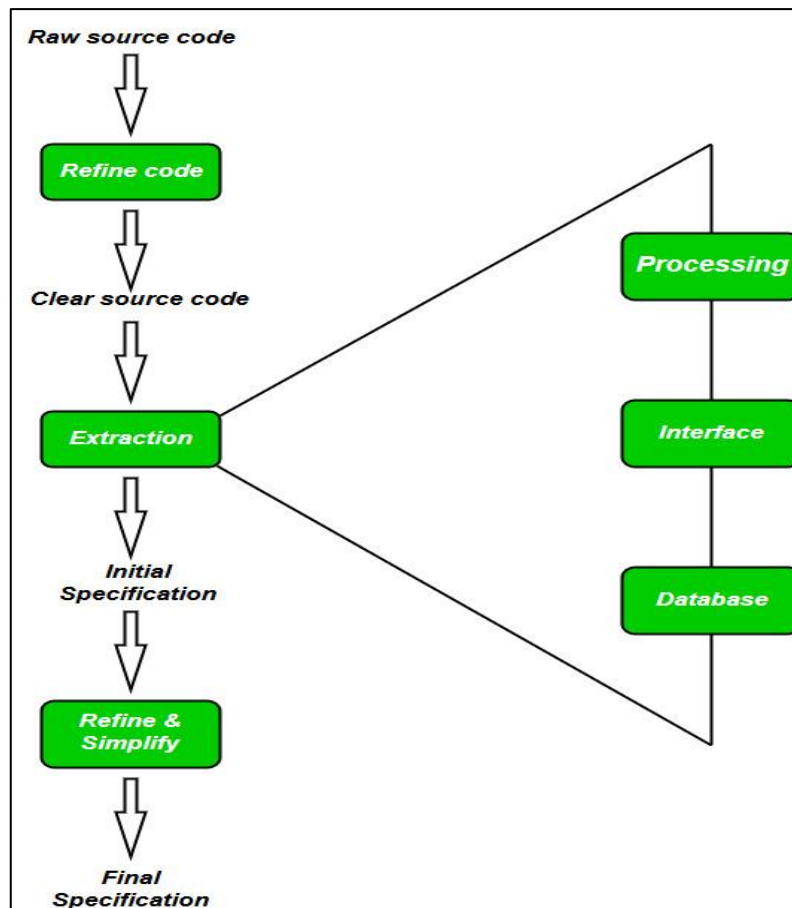
Cope with Complexity.

Recover lost information.

Detect side effects.

Synthesise higher abstraction.

Facilitate Reuse.

**Steps of Software Reverse Engineering:**

- Collection Information:  
This step focuses on collecting all possible information (i.e., source design documents etc.) about the software.
- Examining the information:  
The information collected in step-1 is studied so as to get familiar with the system.

- Extracting the structure:  
This step concerns with identification of program structure in the form of structure chart where each node corresponds to some routine.
- Recording the functionality:  
During this step processing details of each module of the structure, charts are recorded using structured language like decision table, etc.
- Recording data flow:  
From the information extracted in step-3 and step-4, set of data flow diagrams are derived to show the flow of data among the processes.
- Recording control flow:  
High level control structure of the software is recorded.
- Review extracted design:  
Design document extracted is reviewed several times to ensure consistency and correctness. It also ensures that the design represents the program.
- Generate documentation:  
Finally, in this step, the complete documentation including SRS, design document, history, overview, etc. are recorded for future use.

Reverse engineering is the process of transforming code into a model through a mapping from a specific implementation language. Reverse engineering provides good information but is incomplete.

To Reverse engineer a class diagram:

- Identify the rules from mapping from implementation language.
- Using a tool, point to the code we could like to reverse engineer.
- Using the tool, create a class diagram by querying the model

Reverse engineering object diagram: It is the creation of a model from code an object diagram is a very different thing.

To reverse engineer an object diagram

- Choose the target you want to reverse engineer.
- Using a tool or simply walking through a scenario, stop execution at a certain moment in time.
- Identify the set of interesting objects that collaborate in that context and render them in an object diagram.
- As necessary to understand their semantics, expose these object's states
- As necessary to understand their semantics, identify the links that exist among these objects

Reverse Engineering use case:

- Identify each actor that interacts with the system
- For each actor, consider the manner in which that actor interacts with the system, changes the state of the system or its environment , or responds to some event.
- Trace the flow of events in the executable system relative to each actor. Start with primary flows and only later consider alternative paths.
- Represent these actors and use cases in a use case diagram, and establish their relationships.

Reverse engineering in state chart is theoretically possible, but practically not very useful

Reverse engineering in deployment diagram is a process of producing a logical or physical model from the executable code.

- Identify the target element to reverse engineer.
- Walk across the system to discover the h/w structure.
- Walk across the system to discover the components.
- With the help of modeling techniques, design, a create a deployment diagram.



**PROGRAM III :TESTING**

Software Testing is evaluation of the software against requirements gathered from users and system specifications. Testing is conducted at the phase level in software development life cycle or at module level in program code. Software testing comprises of Validation and Verification.

**Software Validation**

Validation is process of examining whether or not the software satisfies the user requirements. It is carried out at the end of the SDLC. If the software matches requirements for which it was made, it is validated.

Validation ensures the product under development is as per the user requirements.

Validation answers the question – "Are we developing the product which attempts all that user needs from this software ?".

Validation emphasizes on user requirements.

**Software Verification**

Verification is the process of confirming if the software is meeting the business requirements, and is developed adhering to the proper specifications and methodologies.

Verification ensures the product being developed is according to design specifications.

Verification answers the question– "Are we developing this product by firmly following all design specifications ?"

Verifications concentrates on the design and system specifications.

Target of the test are -

- Errors - These are actual coding mistakes made by developers. In addition, there is a difference in output of software and desired output, is considered as an error.
- Fault - When error exists fault occurs. A fault, also known as a bug, is a result of an error which can cause system to fail.
- Failure - failure is said to be the inability of the system to perform the desired task. Failure occurs when fault exists in the system.

**Manual Vs Automated Testing**

Testing can either be done manually or using an automated testing tool:

Manual - This testing is performed without taking help of automated testing tools. The software tester prepares test cases for different sections and levels of the code, executes the tests and reports the result to the manager.

Manual testing is time and resource consuming. The tester needs to confirm whether or not right test cases are used. Major portion of testing involves manual testing.

Automated This testing is a testing procedure done with aid of automated testing tools. The limitations with manual testing can be overcome using automated test tools.

A test needs to check if a webpage can be opened in Internet Explorer. This can be easily done with

manual testing. But to check if the web-server can take the load of 1 million users, it is quite impossible to test manually.

There are software and hardware tools which helps tester in conducting load testing, stress testing, regression testing.

#### Testing Approaches

Tests can be conducted based on two approaches –

Functionality testing

Implementation testing

When functionality is being tested without taking the actual implementation in concern it is known as black-box testing. The other side is known as white-box testing where not only functionality is tested but the way it is implemented is also analyzed.

Exhaustive tests are the best-desired method for a perfect testing. Every single possible value in the range of the input and output values is tested. It is not possible to test each and every value in real world scenario if the range of values is large.

Black-box testing

It is carried out to test functionality of the program. It is also called ‘Behavioral’ testing. The tester in this case, has a set of input values and respective desired results. On providing input, if the output matches with the desired results, the program is tested ‘ok’, and problematic otherwise.

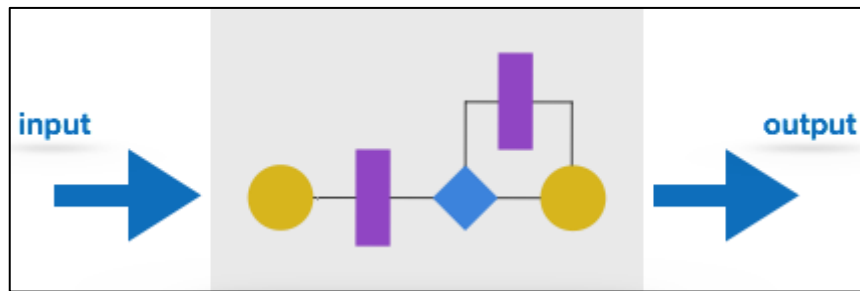
In this testing method, the design and structure of the code are not known to the tester, and testing engineers and end users conduct this test on the software.

Black-box testing techniques:

- Equivalence class - The input is divided into similar classes. If one element of a class passes the test, it is assumed that all the class is passed.
- Boundary values - The input is divided into higher and lower end values. If these values pass the test, it is assumed that all values in between may pass too.
- Cause-effect graphing - In both previous methods, only one input value at a time is tested. Cause (input) – Effect (output) is a testing technique where combinations of input values are tested in a systematic way.
- Pair-wise Testing - The behavior of software depends on multiple parameters. In pairwise testing, the multiple parameters are tested pair-wise for their different values.
- State-based testing - The system changes state on provision of input. These systems are tested based on their states and input.

White-box testing

It is conducted to test program and its implementation, in order to improve code efficiency or structure. It is also known as ‘Structural’ testing.



In this testing method, the design and structure of the code are known to the tester. Programmers of the code conduct this test on the code.

The below are some White-box testing techniques:

- Control-flow testing - The purpose of the control-flow testing to set up test cases which covers all statements and branch conditions. The branch conditions are tested for both being true and false, so that all statements can be covered.
- Data-flow testing - This testing technique emphasis to cover all the data variables included in the program. It tests where the variables were declared and defined and where they were used or changed.

### Testing Levels

Testing itself may be defined at various levels of SDLC. The testing process runs parallel to software development. Before jumping on the next stage, a stage is tested, validated and verified. Testing separately is done just to make sure that there are no hidden bugs or issues left in the software. Software is tested on various levels -

- Unit Testing  
While coding, the programmer performs some tests on that unit of program to know if it is error free. Testing is performed under white-box testing approach. Unit testing helps developers decide that individual units of the program are working as per requirement and are error free.
- Integration Testing  
Even if the units of software are working fine individually, there is a need to find out if the units if integrated together would also work without errors. For example, argument passing and data updation etc.
- System Testing  
The software is compiled as product and then it is tested as a whole. This can be accomplished using one or more of the following tests:
  - Functionality testing - Tests all functionalities of the software against the requirement.
  - Performance testing - This test proves how efficient the software is. It tests the effectiveness and average time taken by the software to do desired task. Performance testing is done by means of load testing and stress testing where the software is put under high user and data

load under various environment conditions.

- Security & Portability - These tests are done when the software is meant to work on various platforms and accessed by number of persons.
- Acceptance Testing

When the software is ready to hand over to the customer it has to go through last phase of testing where it is tested for user-interaction and response. This is important because even if the software matches all user requirements and if user does not like the way it appears or works, it may be rejected.

- Alpha testing - The team of developer themselves perform alpha testing by using the system as if it is being used in work environment. They try to find out how user would react to some action in software and how the system should respond to inputs.
- Beta testing - After the software is tested internally, it is handed over to the users to use it under their production environment only for testing purpose. This is not as yet the delivered product. Developers expect that users at this stage will bring minute problems, which were skipped to attend.
- Regression Testing  
Whenever a software product is updated with new code, feature or functionality, it is tested thoroughly to detect if there is any negative impact of the added code. This is known as regression testing.

#### Testing Documentation

Testing documents are prepared at different stages -

##### Before Testing

Testing starts with test cases generation. Following documents are needed for reference –

- SRS document - Functional Requirements document
- Test Policy document - This describes how far testing should take place before releasing the product.
- Test Strategy document - This mentions detail aspects of test team, responsibility matrix and rights/responsibility of test manager and test engineer.
- Traceability Matrix document - This is SDLC document, which is related to requirement gathering process. As new requirements come, they are added to this matrix. These matrices help testers know the source of requirement. They can be traced forward and backward.

##### While Being Tested

The following documents may be required while testing is started and is being done:

- Test Case document - This document contains list of tests required to be conducted. It includes Unit test plan, Integration test plan, System test plan and Acceptance test plan.

- Test description - This document is a detailed description of all test cases and procedures to execute them.
- Test case report - This document contains test case report as a result of the test.
- Test logs - This document contains test logs for every test case report.

#### After Testing

The following documents may be generated after testing :

- Test summary - This test summary is collective analysis of all test reports and logs. It summarizes and concludes if the software is ready to be launched. The software is released under version control system if it is ready to launch.

#### **Testing vs. Quality Control, Quality Assurance and Audit**

We need to understand that software testing is different from software quality assurance, software quality control and software auditing.

- Software quality assurance - These are software development process monitoring means, by which it is assured that all the measures are taken as per the standards of organization. This monitoring is done to make sure that proper software development methods were followed.
- Software quality control - This is a system to maintain the quality of software product. It may include functional and non-functional aspects of software product, which enhance the goodwill of the organization. This system makes sure that the customer is receiving quality product for their requirement and the product certified as 'fit for use'.
- Software audit - This is a review of procedure used by the organization to develop the software. A team of auditors, independent of development team examines the software process, procedure, requirements and other aspects of SDLC. The purpose of software audit is to check that software and its development process, both conform standards, rules and regulations.

#### **What is a Test Plan?**

Test planning, the most important activity to ensure that there is initially a list of tasks and milestones in a baseline plan to track the progress of the project. It also defines the size of the test effort.

It is the main document often called as master test plan or a project test plan and usually developed during the early phase of the project.

## Test Plan Identifiers:

S.No.	Parameter	Description
1.	Test plan identifier	Unique identifying reference.
2.	Introduction	A brief introduction about the project and to the document.
3.	Test items	A test item is a software item that is the application under test.
4.	Features to be tested	A feature that needs to be tested on the testware.
5.	Features not to be tested	Identify the features and the reasons for not including as part of testing.
6.	Approach	Details about the overall approach to testing.
7.	Item pass/fail criteria	Documented whether a software item has passed or failed its test.
8.	Test deliverables	The deliverables that are delivered as part of the testing process, such as test plans, test specifications and test summary reports.
9.	Testing tasks	All tasks for planning and executing the testing.
10.	Environmental needs	Defining the environmental requirements such as hardware, software, OS, network configurations, tools required.
11.	Responsibilities	Lists the roles and responsibilities of the team members.
12.	Staffing and training needs	Captures the actual staffing requirements and any specific skills and training requirements.
13.	Schedule	States the important project delivery dates and key milestones.
14.	Risks and Mitigation	High-level project risks and assumptions and a mitigating plan for each identified risk.
15.	Approvals	Captures all approvers of the document, their titles and the sign off date.

**Test Planning Activities:**

To determine the scope and the risks that need to be tested and that are NOT to be tested.

Documenting Test Strategy.

Making sure that the testing activities have been included.

Deciding Entry and Exit criteria.

Evaluating the test estimate.

Planning when and how to test and deciding how the test results will be evaluated, and defining test exit criterion.

The Test artefacts delivered as part of test execution.

Defining the management information, including the metrics required and defect resolution and risk issues.

Ensuring that the test documentation generates repeatable test assets.

What is Test case?

A test case is a document, which has a set of test data, preconditions, expected results and postconditions, developed for a particular test scenario in order to verify compliance against a specific requirement.

Test Case acts as the starting point for the test execution, and after applying a set of input values, the application has a definitive outcome and leaves the system at some end point or also known as execution postcondition.

Typical Test Case Parameters:

Test Case ID

Test Scenario

Test Case Description

Test Steps

Prerequisite

Test Data

Expected Result

Test Parameters

Actual Result

Environment Information

Comments

Example:

Let us say that we need to check an input field that can accept maximum of 10 characters.

While developing the test cases for the above scenario, the test cases are documented the following way. In the below example, the first case is a pass scenario while the second case is a FAIL.

Scenario	Test Step	Expected Result	Actual Outcome
Verify that the input field that can accept maximum of 10 characters	Login to application and key in 10 characters	Application should be able to accept all 10 characters.	Application accepts all 10 characters.
Verify that the input field that can accept maximum of 11 characters	Login to application and key in 11 characters	Application should NOT accept all 11 characters.	Application accepts all 10 characters.

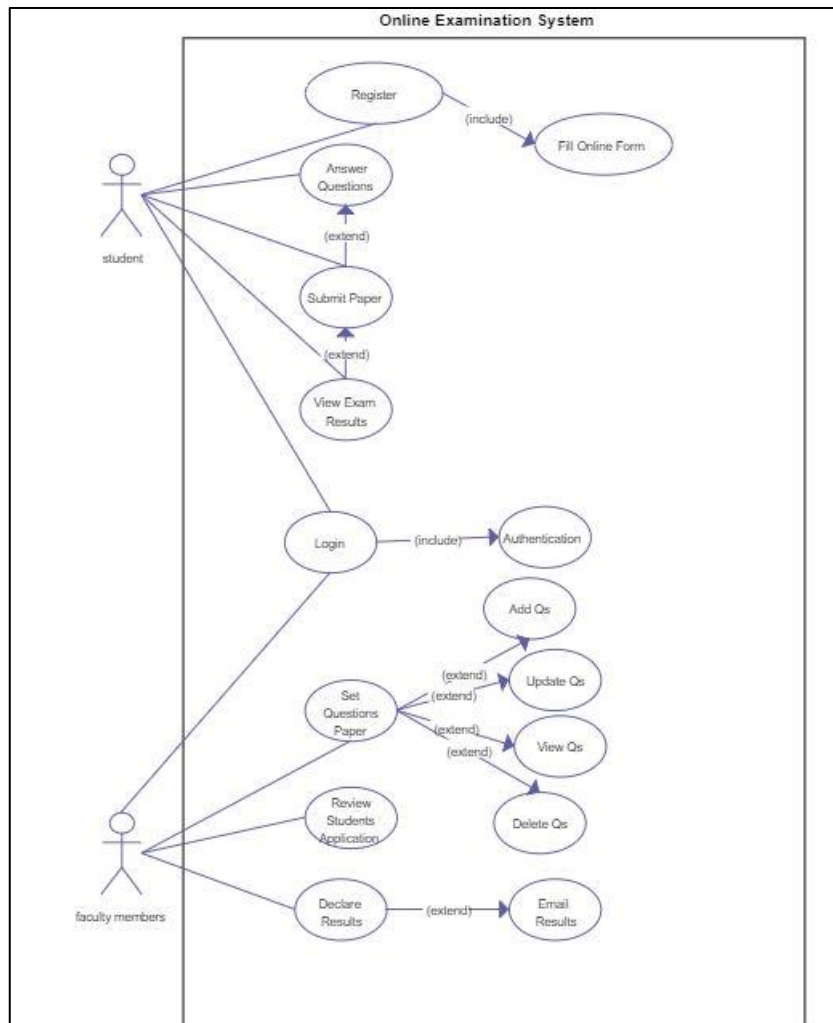
If the expected result doesn't match with the actual result, then we log a defect. The defect goes through the defect life cycle and the testers address the same after fix

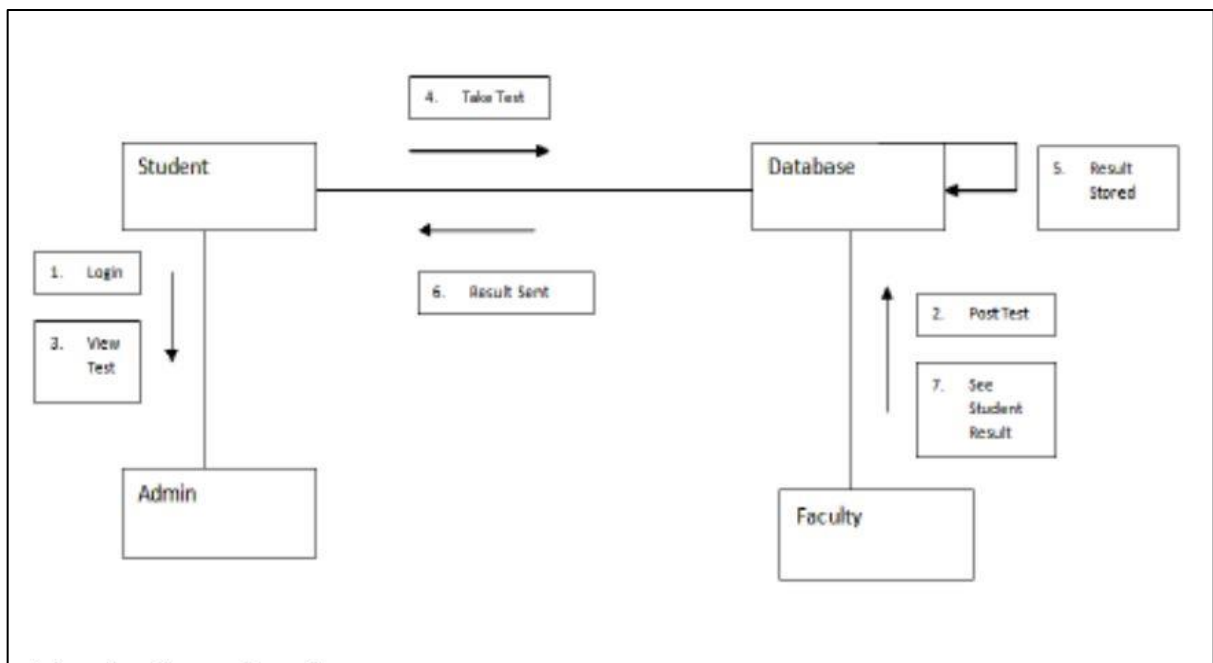
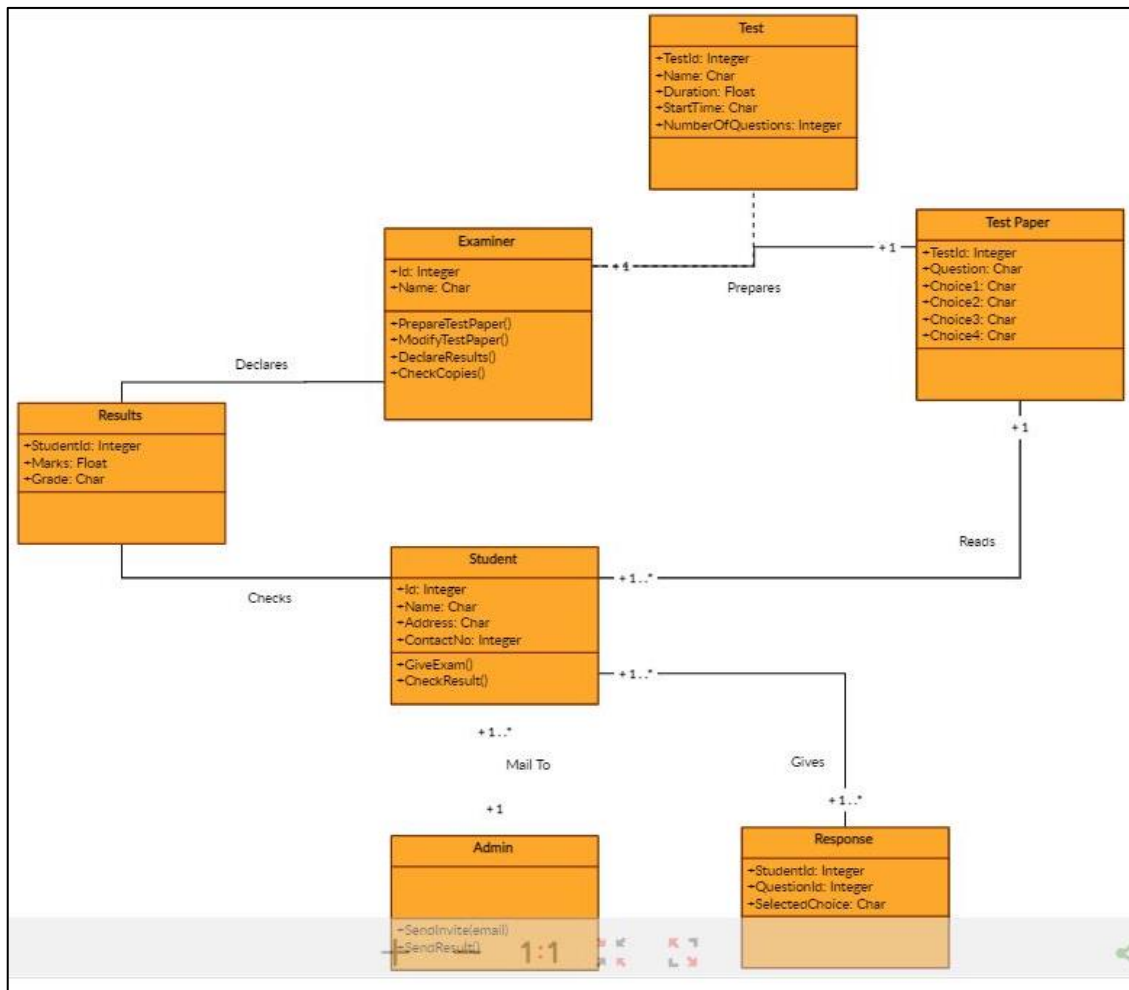


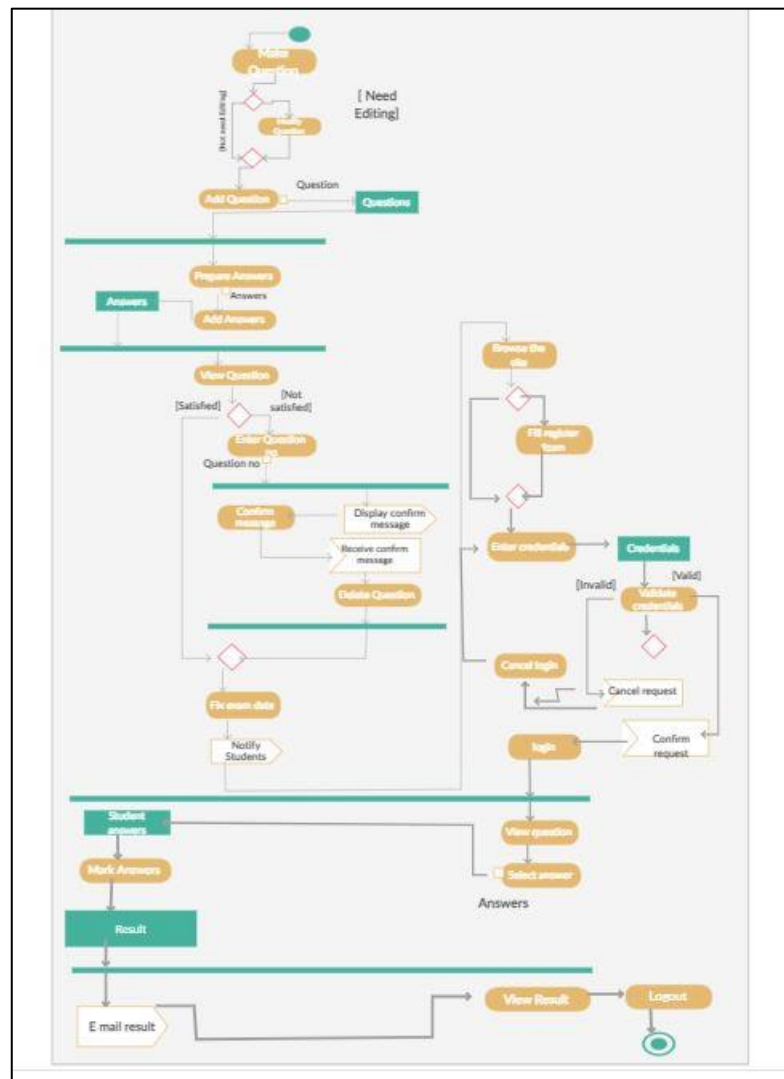
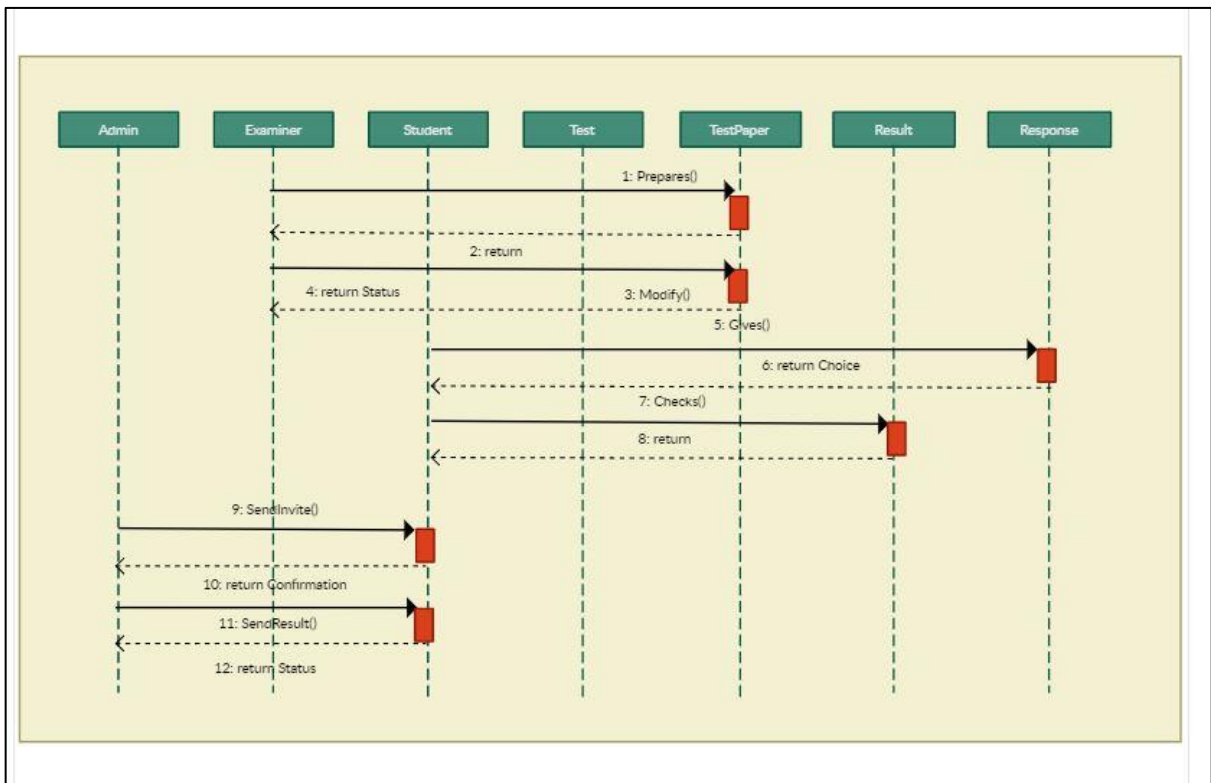
ADDITIONAL EXPERIMENTS

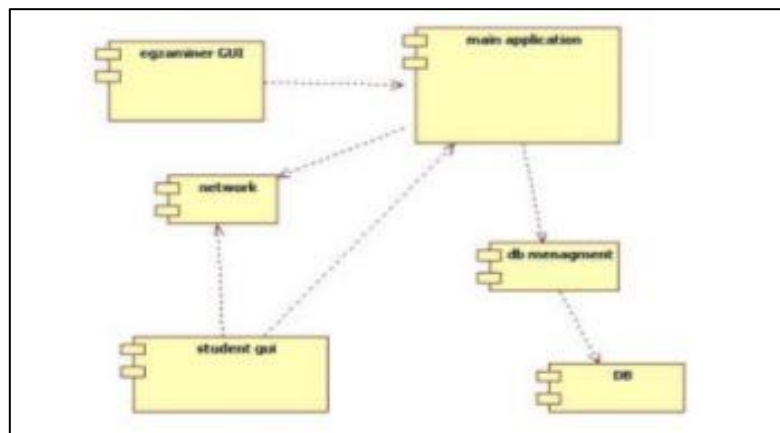
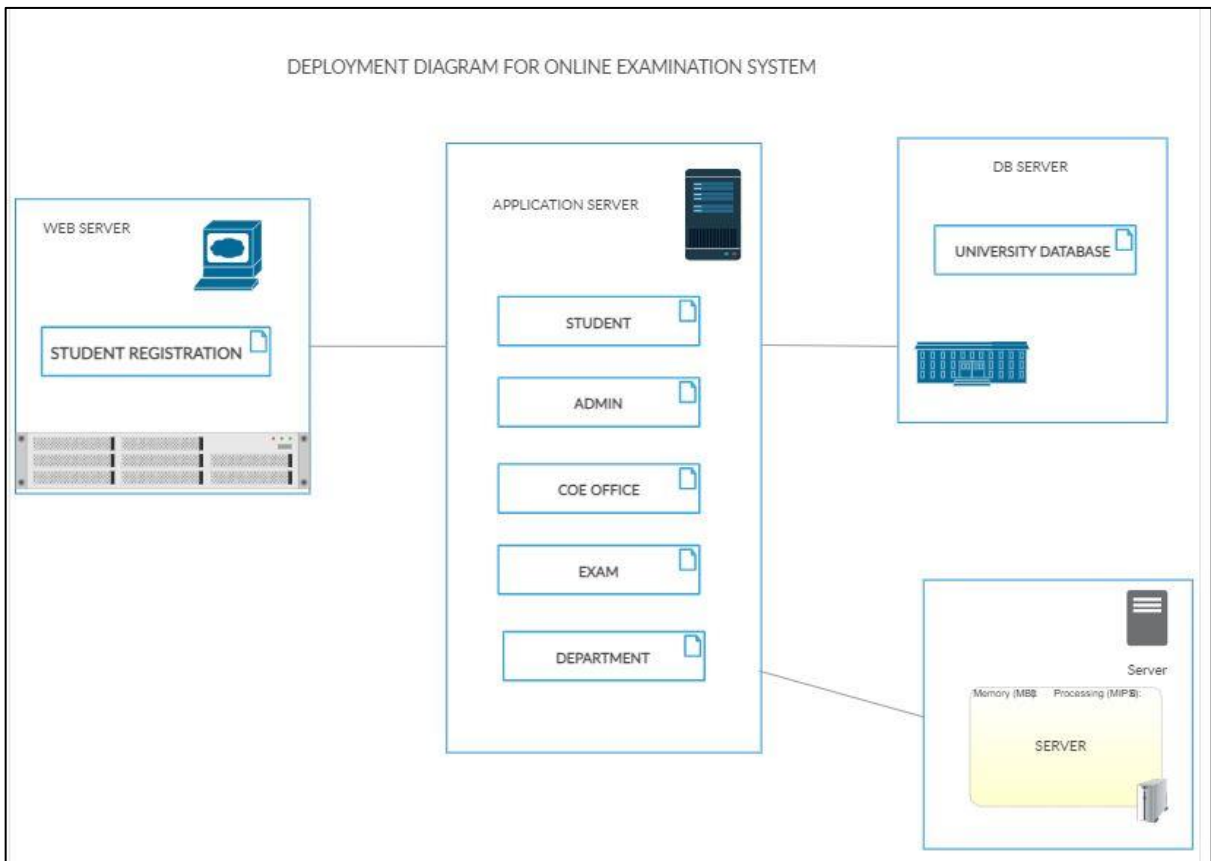
PROGRAM 12:

AIM: Education system-online eamcet examination case study



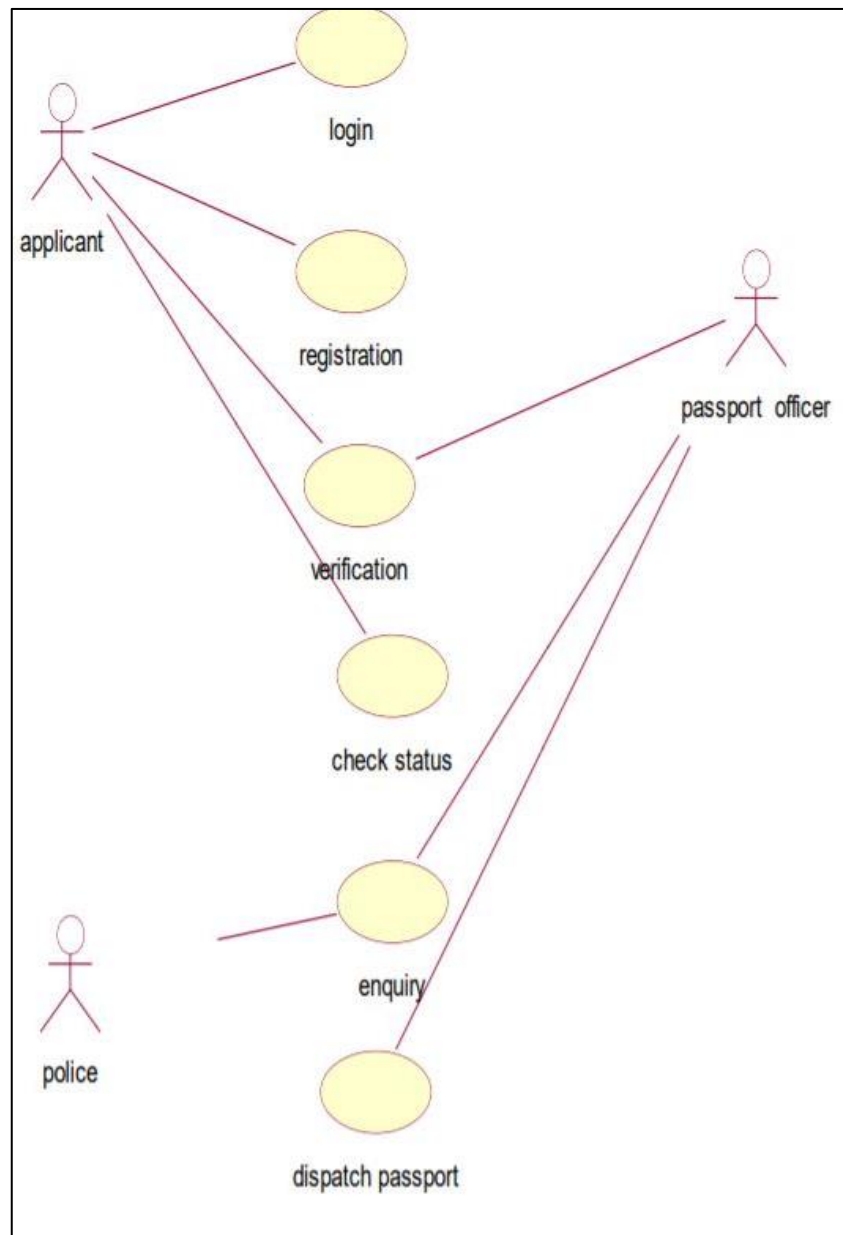


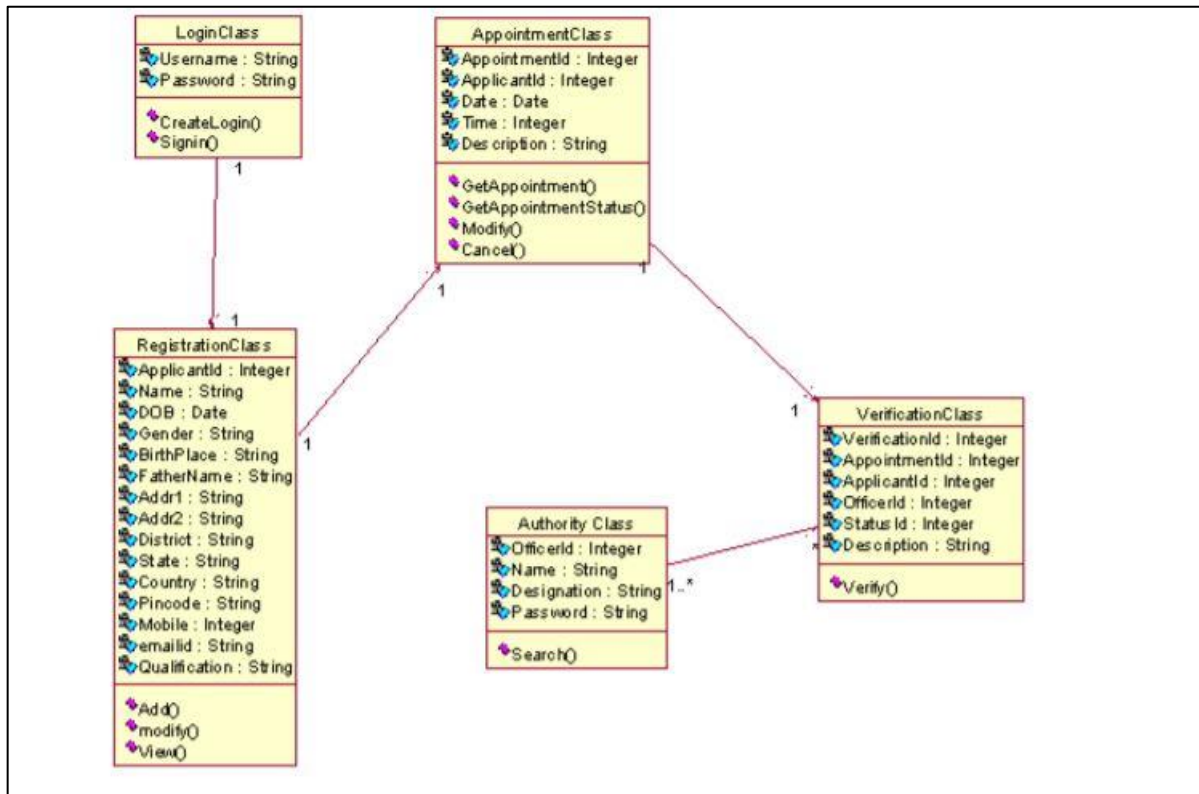


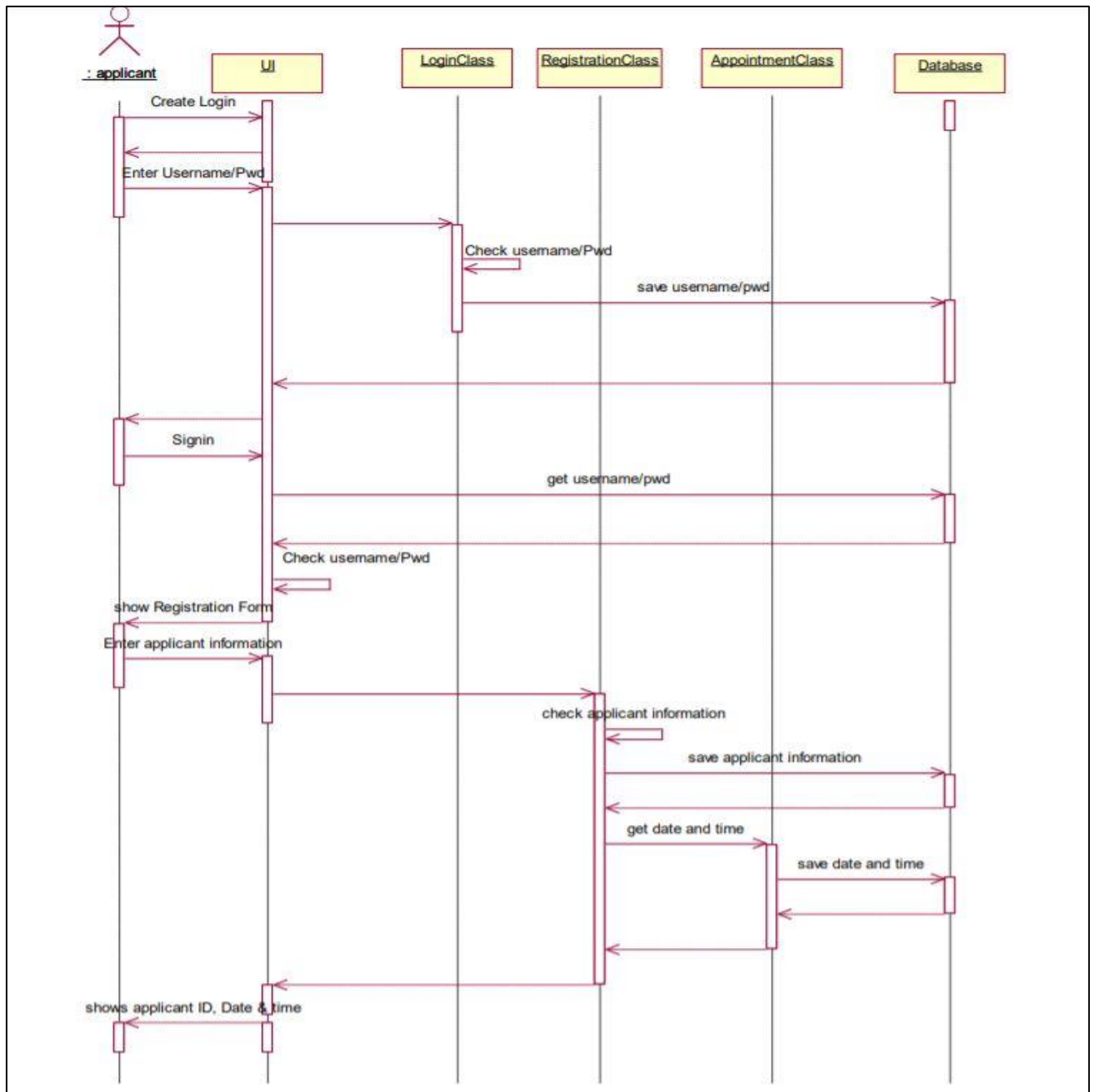


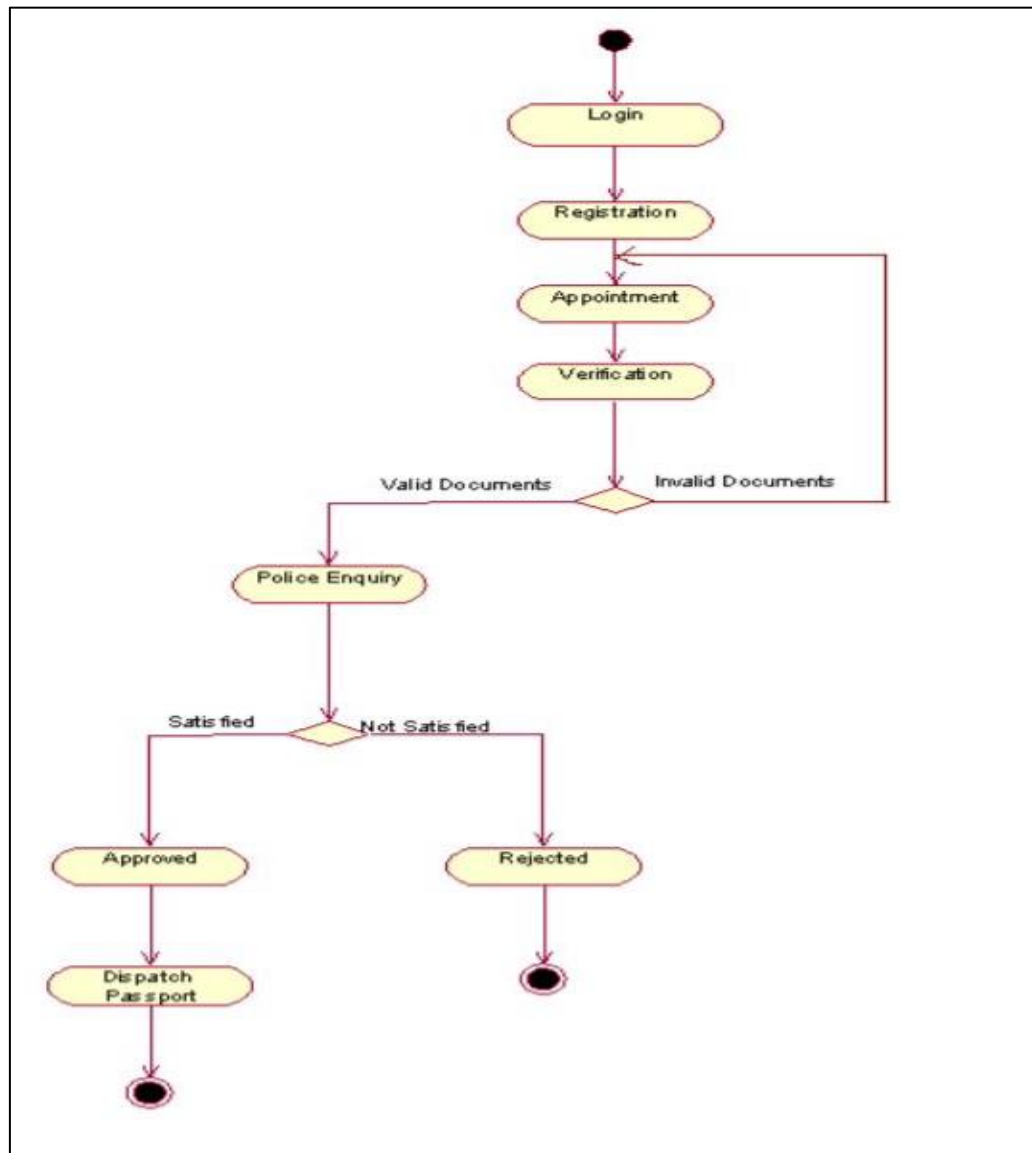
## PROGRAM 13

AIM : Applying for online passport system back end verification process

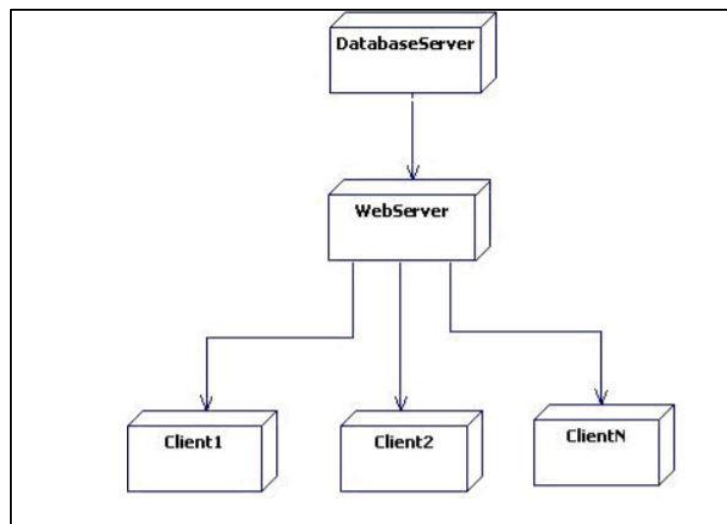
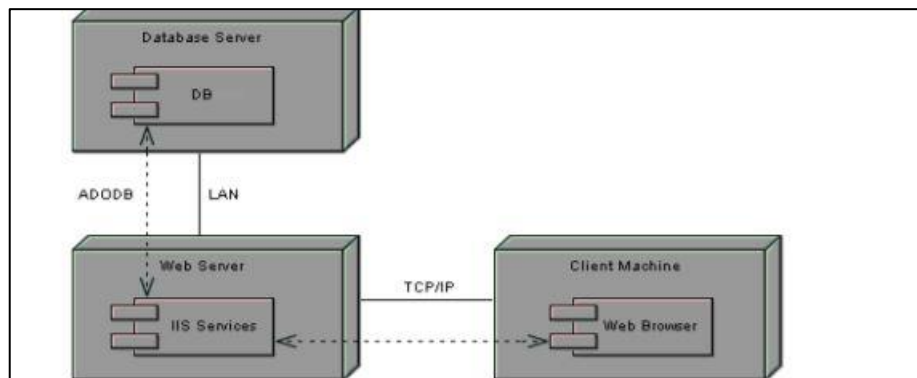
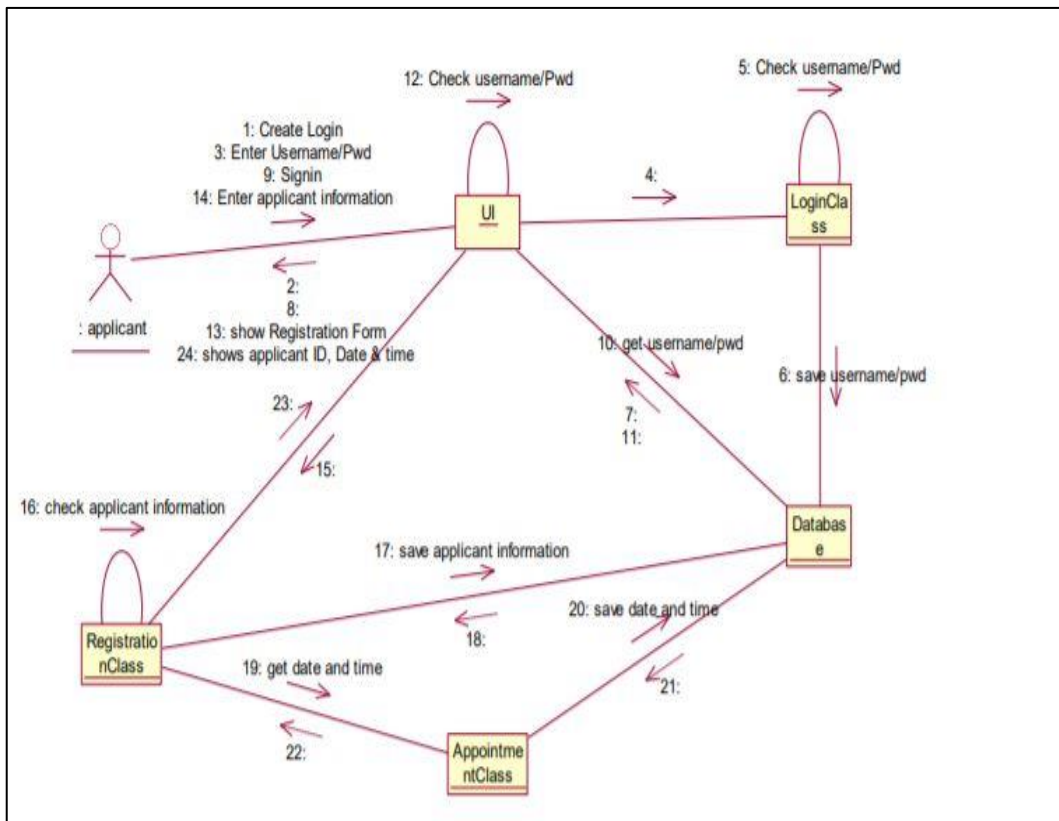








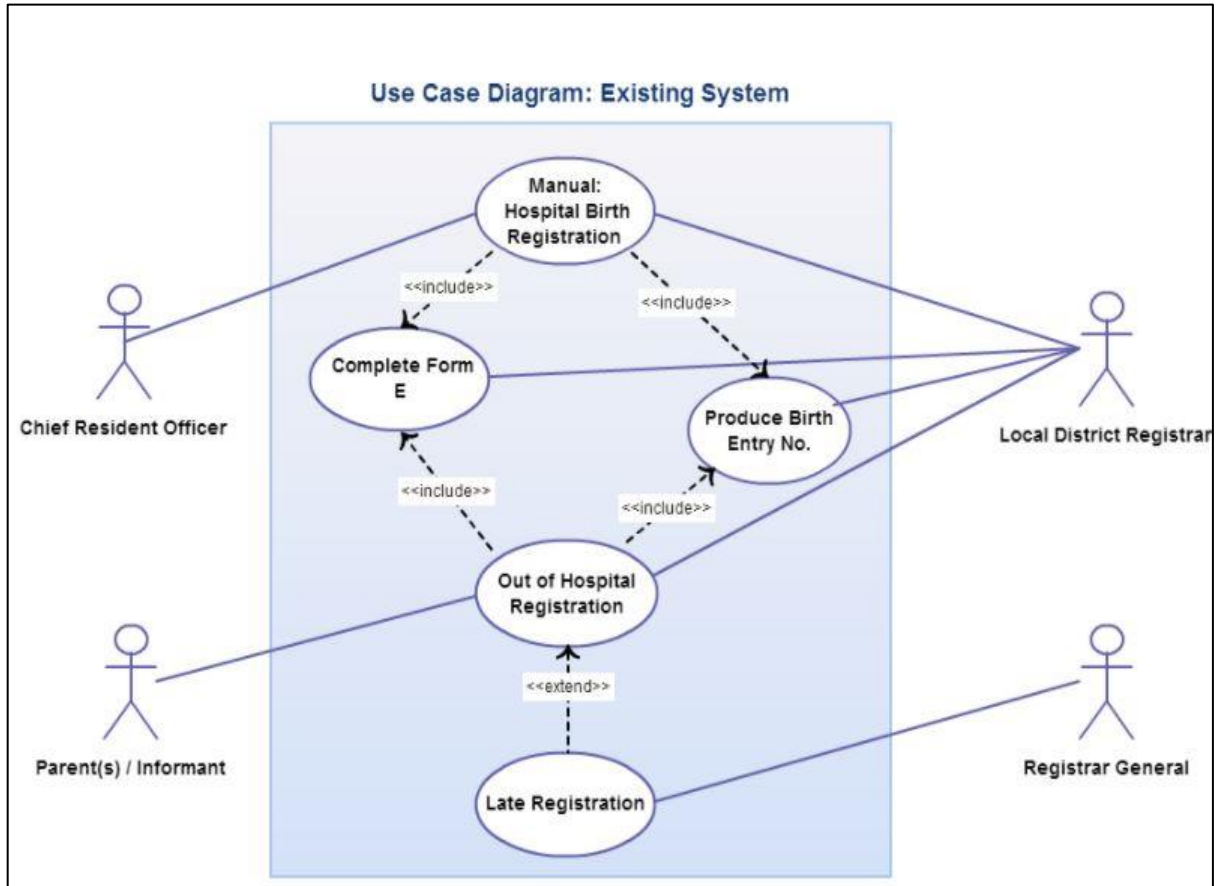


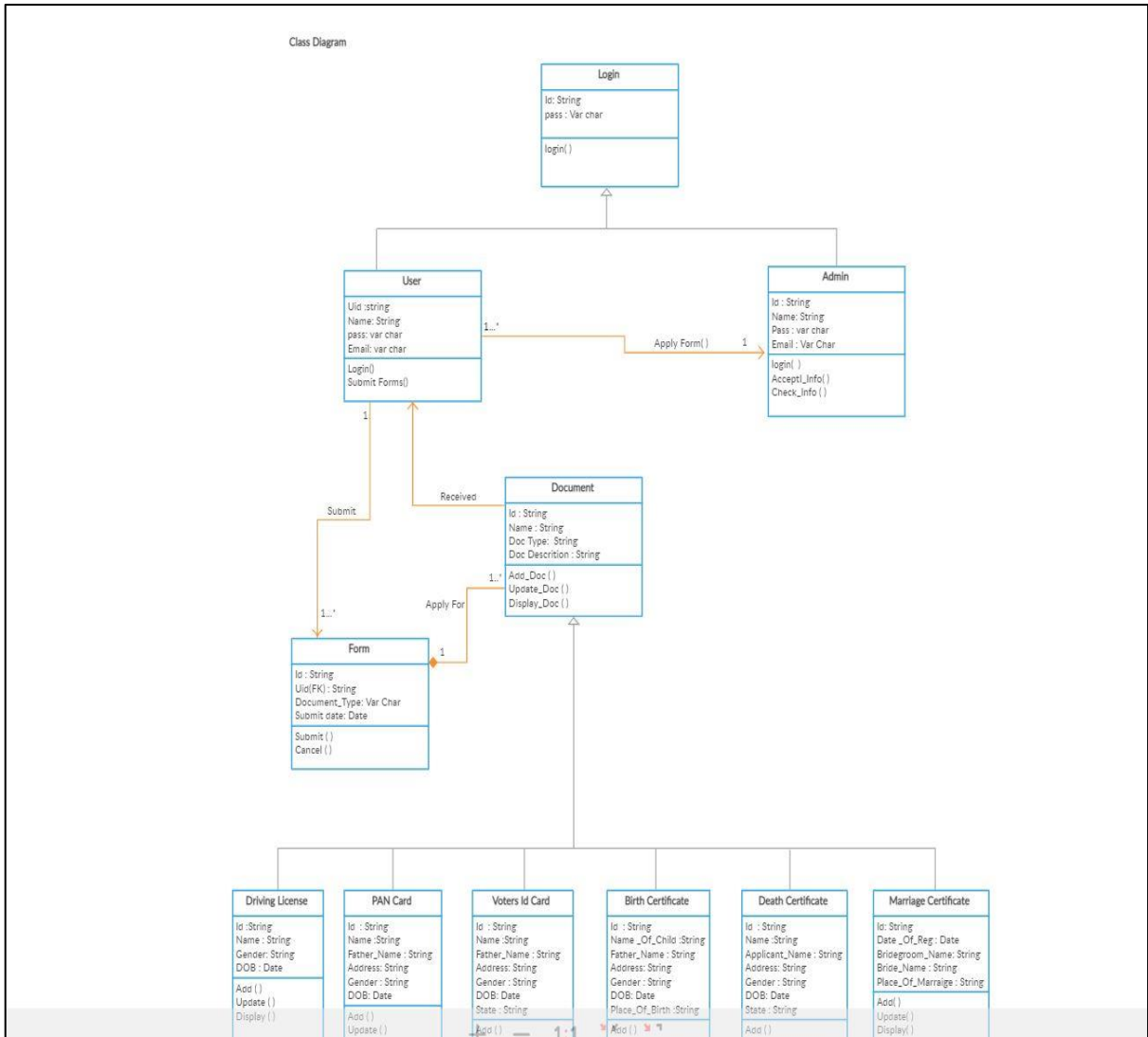


**PROGRAM 14:**

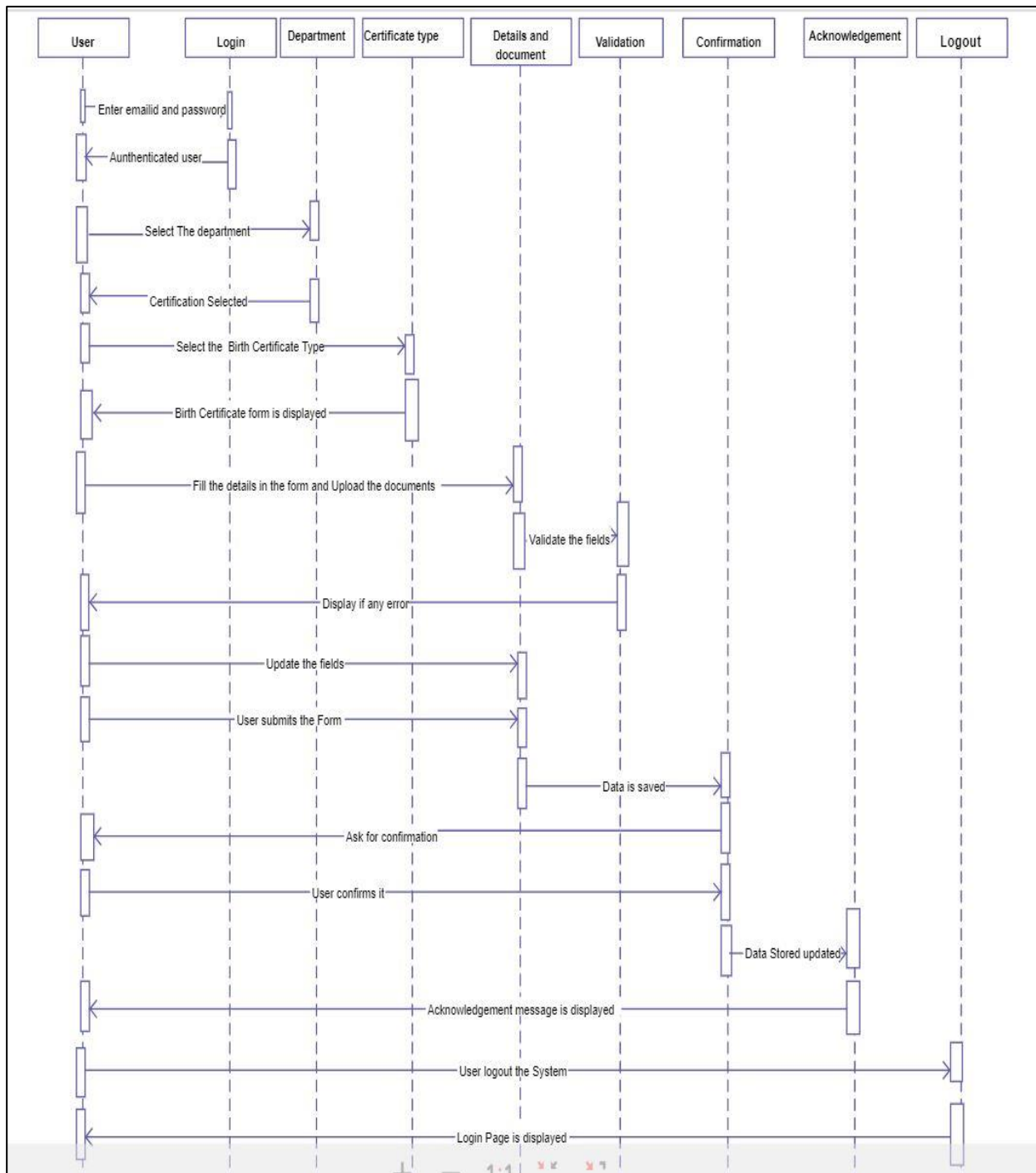
**AIM : Govt. Services: applying certificates**

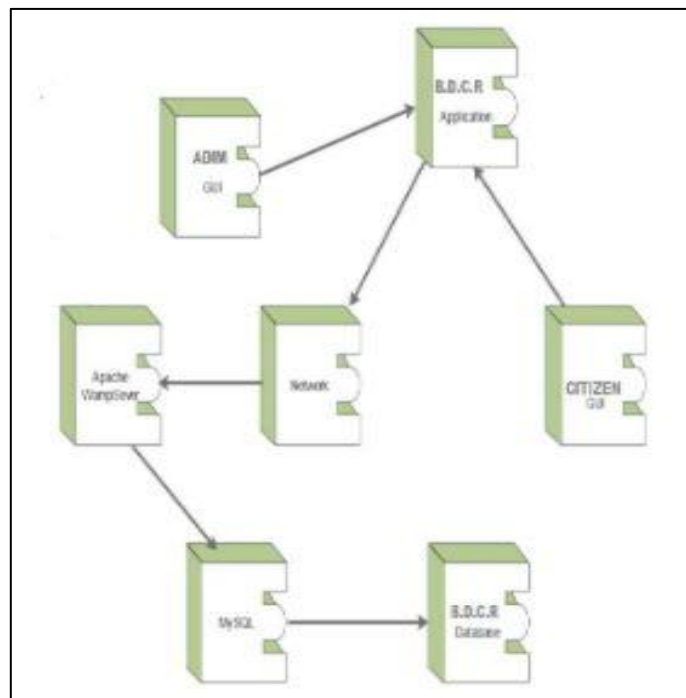
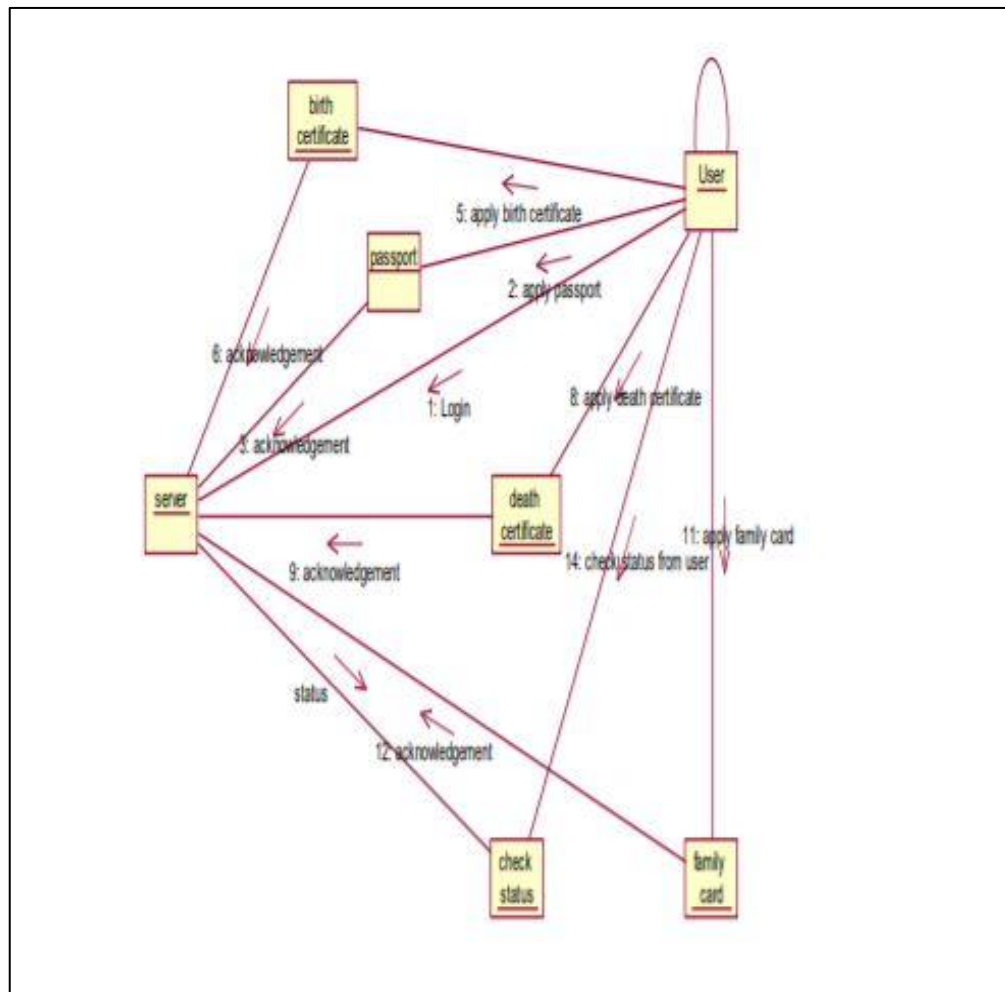
1. Birth certificate
2. caste certificate
3. migration certificate
4. Death certificate

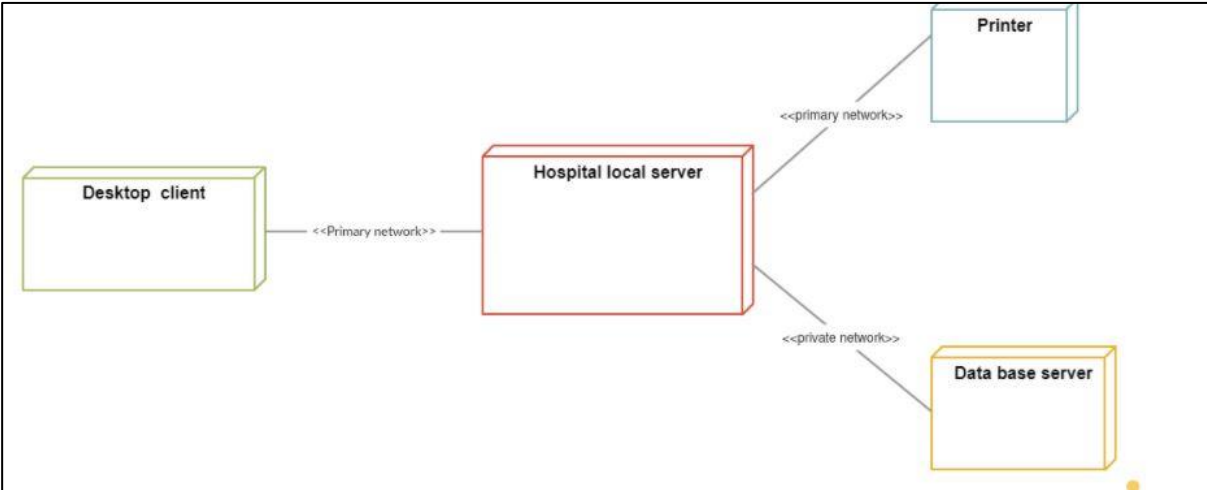












**Viva Questions**

- 1 What is Software Engineering?
- 2 What is the difference between program and software?
- 3 Write out the reasons for the Failure of Water Fall Model.
- 4 What are the characteristics of the software?
- 5 Define the terms :
  - a) Agility b) Agile Team
- 6 What are the various categories of software?
- 7 What are the challenges in software?
- 8 Define software process
- 9 What are the fundamental activities of a software process?
- 10 What are the umbrella activities of a software process?
- 11 What are the merits of incremental model?
- 12 List the task regions in the Spiral model.
- 13 What are the drawbacks of spiral model?
- 14 What is System Engineering?
- 15 List the process maturity levels in SEIs CMM.
- 16 What is an effectors process?
- 17 Define the computer based system.
- 18 What does Verification represent?
- 19 What does Validation represent?
- 20 What is the difference between the “Known Risks” and Predictable Risks”?
- 21 What are the steps followed in testing?
- 22 Explain about the incremental model.
- 23 Explain in detail about the software process.
- 24 Explain in detail about the life cycle process.
- 25 Explain Spiral model and win-win spiral model in detail?
- 26 Name the Evolutionary process Models.
- 27 What are the Objectives of Requirement Analysis?
- 28 What is requirement engineering?
- 29 What are the various types of traceability in software engineering?
- 30 Define software prototyping.
- 31 What are the Requirements Engineering Process Functions?
- 32 What are the benefits of prototyping?
- 33 What are the prototyping approaches in software process?
- 34 What are the Difficulties in Elicitation?



- 35 What are the advantages of evolutionary prototyping?
- 36 What are the various Rapid prototyping techniques?
- 37 What is the use of User Interface prototyping?
- 38 What is System Modeling?
- 39 What are the characteristics of SRS?
- 40 What are the objectives of Analysis modeling?
- 41 What is data modeling?.What is a data object?
- 42 What is cardinality in data modeling?
- 43 What does modality in data modeling indicates?
- 44 What is ERD?
- 45 What is DFD?
- 46 What does Level0 DFD represent?
- 47 What is a state transition diagram?
- 48 Explain in detail about Functional Modeling.
- 49 Explain in detail about Structural Modeling.
- 50 Explain in detail about data modeling.
- 51 Explain about rapid prototyping techniques.
- 52 Explain the prototyping approaches in software process.
- 53 What are the elements of Analysis model?
- 54 What are the elements of design model?
- 55 How the Architecture Design can be represented?
- 56 Define design process. List the principles of a software design.
- 57 What is the benefit of modular design?
- 58 What is a cohesive module?
- 59 What are the different types of Cohesion?
- 60 What is coupling?
- 61 What are the various types of coupling?
- 62 What are the common activities in design process?
- 63 What are the benefits of horizontal partitioning?
- 64 What is vertical partitioning?
- 65 What are the advantages of vertical partitioning?
- 66 What are the various elements of data design?
- 67 List the guidelines for data design.
- 68 Name the commonly used architectural styles.
- 69 Explain in detail the design concepts.
- 70 Explain the design principles.

- 71 Explain the design steps of the transform mapping.
- 72 What are the testing principles the software engineer must apply while performing the software testing?
- 73 Define White Box Testing?
- 74 What are the two levels of testing?
- 75 What are the various testing activities?
- 76 Write short note on black box testing.
- 77 What is equivalence partitioning?
- 78 What is Regression Testing?
- 79 What is a boundary value analysis?
- 80 What are the reasons behind to perform white box testing?
- 81 What is cyclomatic complexity?
- 82 How to compute the cyclomatic complexity?
- 83 Distinguish between verification and validation.
- 84 What are the various testing strategies for conventional software?
- 85 Write about drivers and stubs.
- 86 What are the approaches of integration testing?
- 87 What are the advantages and disadvantages of big-bang?
- 88 What are the benefits of smoke testing?
- 89 What are the conditions exists after performing validation testing?
- 90 Distinguish between alpha and beta testing.
- 91 What are the various types of system testing?
- 92 Explain the types of software testing.
- 93 Explain in detail about Black box testing.
- 94 Explain about the software testing strategies.
- 95 What are the advantages and disadvantages of size measure?
- 96 Write short note on the various estimation techniques.
- 97 What is the Objective of Formal Technical Reviews?
- 98 What is COCOMO model?
- 99 Give the procedure of the Delphi method.
- 100 What is the purpose of timeline chart?
- 101 What is EVA?
- 102 What are the metrics computed during error tracking activity?
- 103 Why software change occurs?
- 104 Write about software change strategies.
- 105 Define CASE Tools.

- 106 What is software maintenance?
- 107 Define maintenance.
- 108 What are the types of software maintenance?
- 109 What is architectural evolution?
- 110 How the CASE tools are classified.
- 111 Explain about software cost estimation.