EX no:01 DATE:	To identify a Software system that needs to be developed-Online Course Reservation System

Aim:

To identify a software system that needs to be developed – Online Course Registration System.

Introduction:

An Online Course Reservation System would be a great software system to develop. It could streamline the process of reserving courses, managing enrollments, handling payments, and providing users with a seamless experience.

Features of Online course Reservation System:

User Registration and Authentication:

- Allow users to create accounts and log in securely.

Course Catalog:

- Display a list of available courses with details such as course name, description, instructor information, schedule, and fees.

Search and Filter:

- Enable users to search for courses based on keywords, categories, dates, locations, or other criteria..

Course Details:

- Provide comprehensive information about each course, including syllabus, prerequisites, learning objectives, and materials required.

Enrollment Management:

- Allow users to enroll in courses they are interested in, view their enrolled courses, and manage their registrations.

Payment Integration:

- Integrate with payment gateways to facilitate secure online payments for course fees.

Waitlisting:

- Implement a waitlist feature for courses that are fully booked, allowing users to join a queue for potential openings.

Feedback and Ratings:

- Enable users to provide feedback and ratings for courses they have completed, helping others make informed decisions.

Admin Dashboard:

- Provide administrators with a centralized dashboard to manage courses, user accounts, payments, generate reports, and track analytics.

Instructor Portal:

- Offer instructors a dedicated portal to manage their course offerings, upload course materials, communicate with students, and track attendance.

Mobile Compatibility:

- Ensure the system is responsive and mobile-friendly to allow users to access and use it conveniently on various devices.

Integration with Learning Management Systems (LMS):

- If applicable, integrate with existing LMS platforms to sync course content, grades, and other data seamlessly.

Result:

Thus the software system that is need to be developed for online course reservation system was executed successfully.

Ex no:02 DATE:	**Document the Software Requirement Specification (SRS) for the identified model**

Aim:

To implement a software for Online Course Reservation System.

Problem statement:

Online Course Reservation System is meant for students who wish to apply for the course through online in a college. At first the system will list the available courses in the college. Then the student will check the available seats for the particular course in the college. If the interested course is available then the student will login and directed to fill the application. The system will check for the eligibility and if the student is eligible then they will directed for payment to reserve the course. If the course is reserved, then an acknowledgement will be send to the student and also to the registrar of the college.

Introduction

Purpose:

This SRS is developed for the entire system. Course Reservation by means if Direct communication would consume time and takes more man power. In order to resolve this one, an Automated Online Course Reservation System provides a smart way to reserve the course through online which helps the Student to apply in a efficient and convenient way.

Document Conventions:

In general, this document follows IEEE formatting requirements. This document contains Times template font size 14 for Sub headings, Size 18 for Headings and Arial template font size 11 for Descriptions throughout the document. Heading are followed by subheadings which is followed by paragraphs.

Content	Font face	Font size	Font style
Heading	Times new Roman	13	bold
Sub-heading	Times new Roman	13	Bold
Paragraph	Times new Roman	13	Normal

Intended Audience and Reading Suggestions:

This document can be used by,

- Developers
- Testers
- Users

Product Scope:

- The objective of the Software is to provide an easy way for students to reserve a course through online.
- The main goal of this system is the time efficiency.

References:

- IEEE Software Requirement Specification Template
- https://krazytech.com/projects/sample-software-requirements-specificationsrs

Overall Description:

Product Perspective:

The Online Course Reservation System will be useful for students so that they can reserve their interested course in a college. This Online Course reservation system is a interface between 'Students' and the 'College'. This system is developed to save energy and time of the student.

In the System, Students must register to reserve their interested course in a college. Then Students must enter the details in the application form and the system checks for the eligibility of student. If the student is eligible he/she will be allowed to pay for the course through E-Pay after payment both the student and registrar will be notified.

Product Functions:

- The System secures the information about the student's reservation
- The Students will receive SMS and Mail updates
- The Registrar will get the report about the registration made by each students

User Classes and Characteristics:

Students - They are the person who desires to obtain the course and submit the information to the database.

Administrator - He has privilege to allocate the seats for the course and check the eligibility of the student. He can approve or reject student's application.

Operating Environment:

This System Support following Operating Systems

- •Windows 10
- •Windows 8
- •Windows 7
- •Windows XP.

Design and Implementation Constraints

- The information of all users must be stored in a database that is accessible by the administrators.
- My SQL Server will be used to maintain a database.
- Users may access from any computer that has internet browsing capabilities.
- The system will work 24x7

User Documentation

The System provides User manual that defines the functions and option available in the system. The User manual will be available in a document format

Assumptions and Dependencies

- •The student may be required to upload the certificates.
- •The student and admin should have a internet connection to avoid disruption.

External Interface Requirements:

User Interfaces

- •The interface should be user friendly for the user. It can be implemented by using java script.
- •The user interface is easy to implement.

Hardware Interfaces

The System requires the Intel processor and minimum of 2 GB RAM for the client. And also have the dedicated links between the server and clients.

Software Interfaces

The client machines require Microsoft Windows XP or better. The server requires Oracle Database to hold all information, both the client and server computer must have internet browser to work online.

Communications Interfaces

The System will perform the following functions:

- Sophisticated and -friendly interface for all system.
- Individual account or profile for all related to the system.
- Sophisticated interfaces for all people who related to the system.
- Implement student, course and instructor database systems.
- Implement Account System for managing invoices.
- Keepsecretforallofstudentprofiles.Eachdivisioncansceonlynecessar- ydata of each student for analyzing.
- Internet connection to work on with the system.

System Feature:

System Featured

Description and Priority Reserve Seat:

- Sign In: The first needs to sign in to the system with the name and password he/she have provided with. The system needs to check for validation of that name and password and then only allow he/she to access the system.
- Check Availability: They must be allowed to see all available options for courses. And see if the seat is available or not.
- Reserve Position: Then if the position is available then it must be booked by the only and should not be granted to any other again till it gets free

Maintain History:

- The watch history for the course registered by the user must be maintained regularly when he/she signs out from the account.

Report Generation:

- Course Information: The System shall generate reports on course about the following

Information:

Course ID, Course Name.

- The System will generate reports on seats availability about the following information: Course ID, Seat Number, Reserved or Unreserved.

Database:

- Mandatory Information: First Name, Last Name, Phone Number, ID, Address, Postal Code, City, Country, name and Password.
- Course Type: Technical or Non-Technical, Instructor, Instructor Details.
- Update Information: The registrar will allow Administrator to update any of the information.
- Course Related Information: Course ID, No. of Seats.

Functional Requirements

The course registration system has the following requirements:

1. Login
2. View course details
3. Reserve for course
4. Pay fee
5. Check Status

Actors involved:

1. Student
2. Registrar

REQ-1: Login

The User enters the name and password and chooses whether the user is Student or Registrar. If entered details are valid, the user's account becomes available. If it is invalid, an appropriate message is displayed to the user.

REQ-2: View course details

In this use case, a student can search all the courses available to him and choose the best course he wants. The student can view the course duration, faculty and department of the courses he may choose.

REQ-3: Rest RVE for course

When a student has successfully chosen a course, he can register to that course.

Upon registration, the student's details are stored in the database.

REQ-4: Pay fee

After registration to any course, the student may see the details of his current course. He may wish to know details about fees and other information.

REQ-5: Check status

The system displays the status information to the student.

Other non-functional Requirements:

Performance Requirement

- Capacity: The System must support 1000 people at a time.
- Interface: The interface screen shall respond within seconds.
- Conformity: The systems must confirm to the Microsoft Accessibility guidelines.
- Network Connection: It should be connected to internet 24 X 7. And the Server must be on all time.

Safety and Security Requirements

- Identification: The system requires to identify himself or herself using E-mail.
- Login ID: Any who uses the system shall have a Login ID and Password.
- Modification: Any modification(Insert, Delete, Update) for the Databases shall be authorized and done only by the administrator.
- Administrator's Rights: Administrator shall be able to view and modify all information in System.
- Back Up: The system shall provide the capability to back-up the Data
- Errors: The system shall keep a log of all the errors.
- Reliability: The system keeps the data securely students who enrolled the courses.

Result:

Thus the implementation of Software Requirement Specification for Online course reservation system was implemented successfully.

EX no:03 DATE:	To identify the Use cases and develop the use case model

Aim:

To identify the use cases and develop use case model.

Use case diagram:

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different usecases in which the user is involved.

A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

Purpose of use case diagram:

The purpose of use case diagram is to capture the dynamic aspect of a system. However, this definition is too generic to describe the purpose, as other four diagrams (activity, sequence, collaboration, and Statechart) also have the same purpose.

Basic Use Case Diagram Symbols and Notations:

System:

Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.

Usecase:

Draw use cases using ovals. Label the ovals with verbs that represent the system's functions.

Actors:

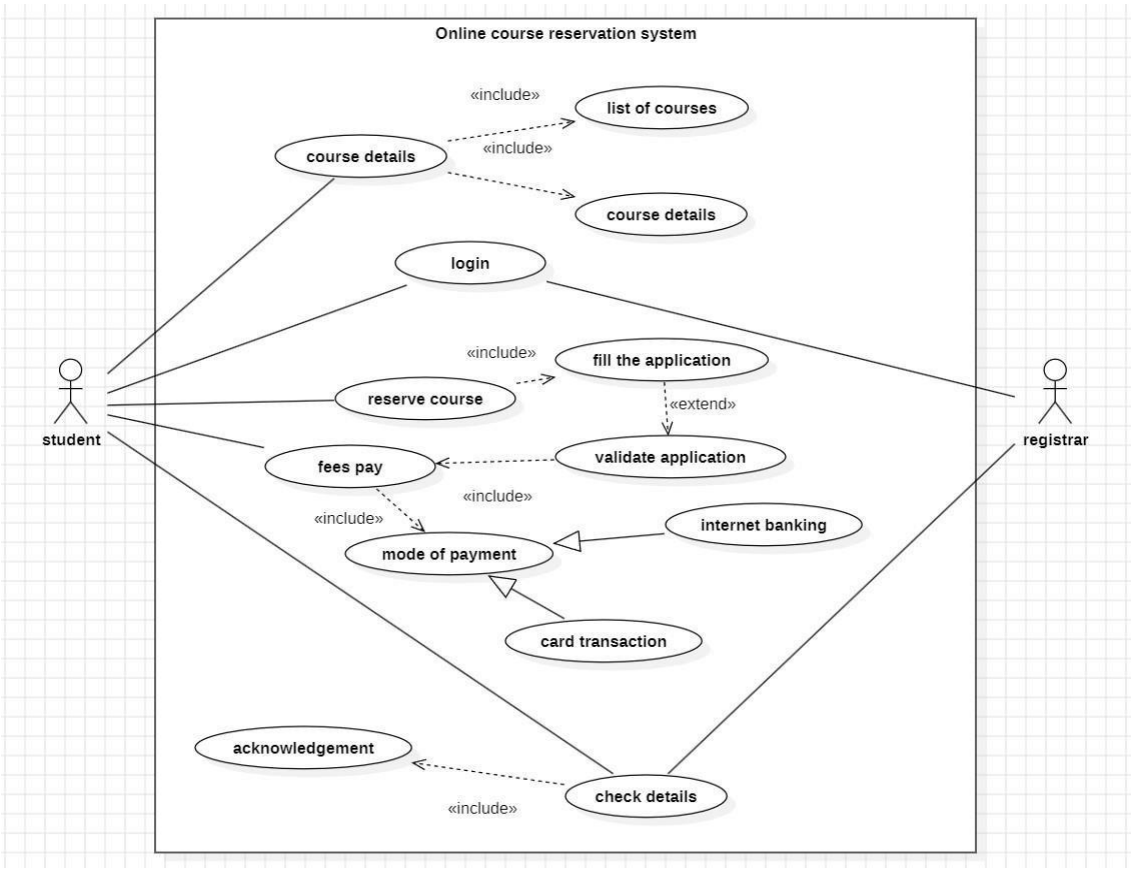
Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype.

Relationships:

Illustrate relationships between an actor and a use case with a simple line. For relationships among use cases, use arrows labeled either "uses" or "extends." A "uses" relationship indicates that one use case is needed by another in order to perform a task. An "extends" relationship indicates alternative options under a certain use case.

Use case diagram for online course reservation system:

Use case modelling:



Usecase name:

Login Brief:

The Student enters into the system's software and looks into the control panel, then he enters the login ID and password. If the password is valid then the system will display the next step or if the password is wrong then it shows a message.

Casual:

Success scenario:

The Student enters into the system's software and looks into the control panel, then he enters the login ID and password.. If the password is valid then the system will display the next step.

Alternate scenario:

If the password entered by the Student is invalid, the system cannot be opened and can't perform any actions. So he must reenter the valid password. The system allows to enter the valid password upto three attempts.

Usecase name: Check Status

Brief:

The Student can enter the system and select the course to check the status of seat availability for the course. If the seat is available for that course then the student will select the course else the student will select the other available course.

Casual:

Success scenario:

The student can enter the system and select the course to check the status of seat availability for the course. If the seat is available for that course then the student will select the course.

Alternate scenario:

But at some situations, if the seat is not available then the student will select the other available course.

RESULT:

Thus the usecase diagram for Online Course registration System is implemented and executed successfully.

EX no:04 DATE:	**identify the conceptual classes and develop a domain model and derive class diagram for Online Course Reservation System**

Aim:

To identify the conceptual classes and develop a domain model and derive class diagram for Online Course Reservation System.

Domain Model:

Steps to create a domain model:

1. Find conceptual classes.
2. Draw them as classes in a UML class diagram.
3. Add Associations and attributes.

Finding Conceptual classes:

1. Reuse or modify existing conceptual models
2. Use category list.
3. Identify noun phrases

Identify the noun phrase:

Main success scenario:

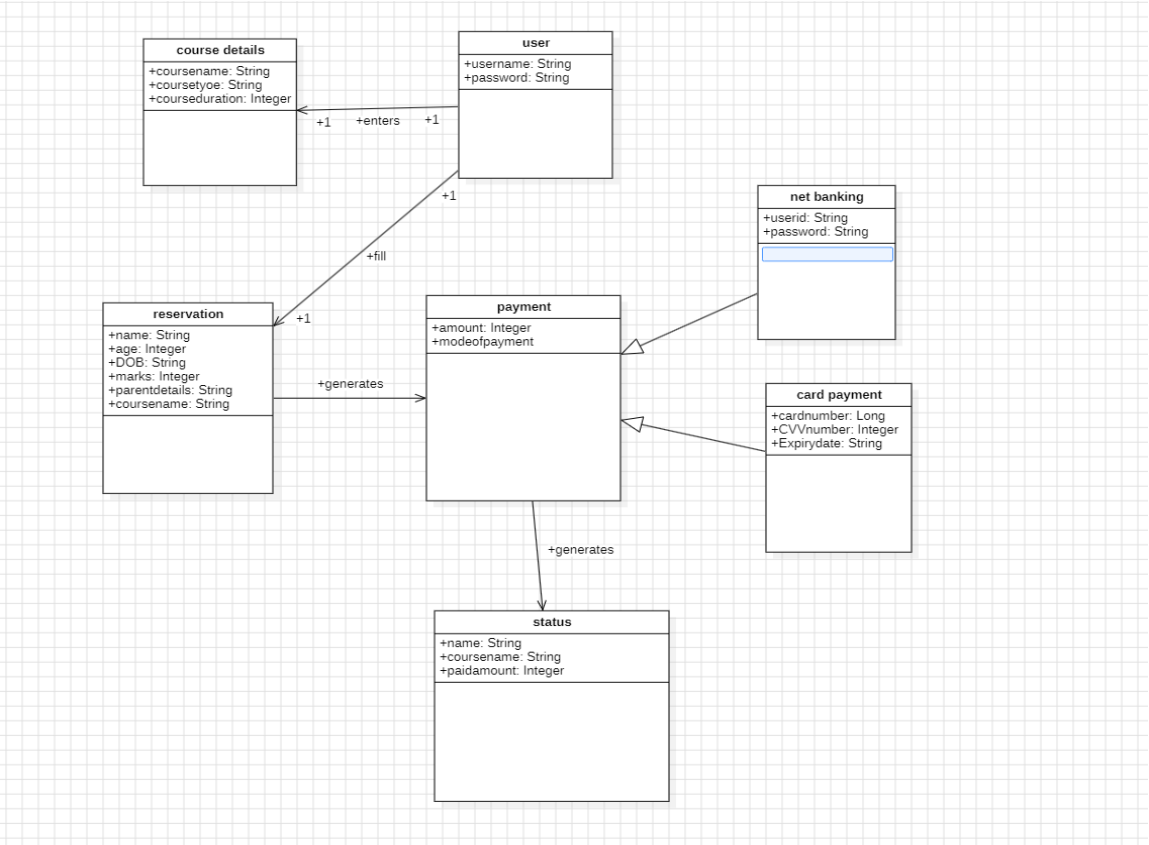
- Here student can view the course details should login with their username and password.
- Next to the select course the student should reserve.
- For the reservation process we should give the appropriate details of the students.
- Then the system validate the user details.
- After the validation step, the student should pay the fees for the reserved course.
- After the end of payment the acknowledgement sent via the email of the student and so only the student can easily check their details..

Nouns:

- Course
- Student
- Registrar
- Detail
- Username
- Password

- Acknowledgement
- Email
- Fees
- System
- Payment
- Details

Domain Model:



Class Diagram for Online course reservation system:

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

Purpose of Class Diagrams:

- Analysis and design of the static view of an application.
- Describe responsibilities of a system.
- Base for component and deployment diagrams.
- Forward and reverse engineering.

Members:

UML provides mechanisms to represent class members, such as attributes and methods, and additional information about them.

Visibility:

To specify the visibility of a class member (i.e. any attribute or method), these notations must be placed before the member's name.

+ Public
- Private
Protected
- Package

Derived property:

Is a property which value (or values) is produced or computed from other information, for example, by using value of other properties.

Derived property is shown with its name preceded by a forward slash /'.

Scope:

The UML specifies two types of scope for members: instance and classifier, and the latter is represented by underlined names.

- Classifier members are commonly recognized as "static" in many programming languages.

The scope is the class itself.

- Attribute values are equal for all instances
- Attribute values maya tete inane sale i change instance attribute)

To indicate a classifier scope for a member, its name must be underlined. Otherwise, instance scope is assumed by default.

Class Notation:

UML class is represented by the following figure. The diagram is divided into four parts.

- The top section is used to name the calss.
- The third section is used to describe the operations performed by the class.

Classes are used to represent objects. Objects can be anything having properties and responsibility.

Object Notation:

It is represented in the same way as the class. The only difference is that name which is underlined as shown in the following figure

Classes:

1.User

- Username: String
- Password: String

2.CourseDetails

- Coursename: String
- Coursetype: String
- Duration: Integer

3.Reservation

- Name: String
- Age: Integer
- DOB: String
- Marks:Integer
- Parentdetail:String

4.Payment

- Amount: Integer
- ModeofPayment: String

5.NetBanking

- userid: String
- password: String

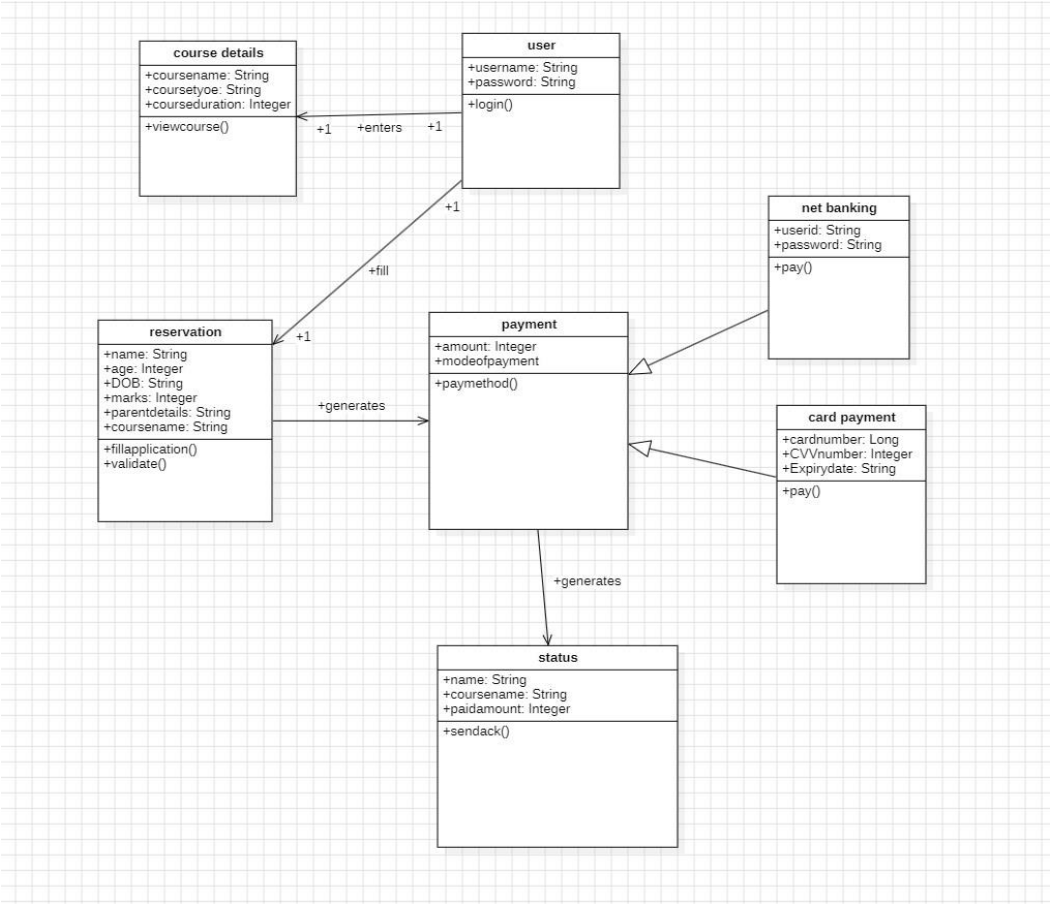
6.CardPayment

- Cardnumber: Long
- ExpiryDate: String
- CVVnumber: Integer

7.Status

- Name: String
- Coursename: String
- Paidamount: Integer

Class diagram for online course reservation system:



Result:

Thus the class diagram for Online Course Registration System is implemented and executed Successfully.

EX no:05 DATE:	**Using the identified scenarios, find the interaction between using objects and represent them using UML Sequence and Collaboration Diagram for Online Course Reservation System**

Aim:

To represent the interaction between objects using UML Sequence diagram and Collaboration diagram for Online Course Reservation System

Sequence diagram:

An object is shows as an box and the top of dash vertical line. This vertical line is called object life line. The life line represents objects life during interaction this was give by “Jackupson”. Each message is represent by an arrow between the lifeline of Two object.

The order of message is occurred from top to bottom of a page. Message contain Messages name, argument and some control information.

Self call is an message that an object sends to itself by sending messages arrow back to the same lifeline.

A sequence diagram illustrates a kind of format in which each object interacts via message. It is generalize between two or more specialized diagram.

This interactive behaviour is represented in UML by two diagrams known as Sequence diagram and Collaboration diagram. Sequence diagram emphasizes on time sequence of messages and collaboration diagram emphasizes on the structural organization of the objects that send and receive messages.

The purposes of interaction diagrams are to visualize the interactive behaviour of the system. Now visualizing interaction is a difficult task. So the solution is to use different types of models to capture the different aspects of the interaction, that is why sequence and collaboration diagrams are used to capture dynamic nature but from a different angle.

The purposes of interaction diagram can be describes as:

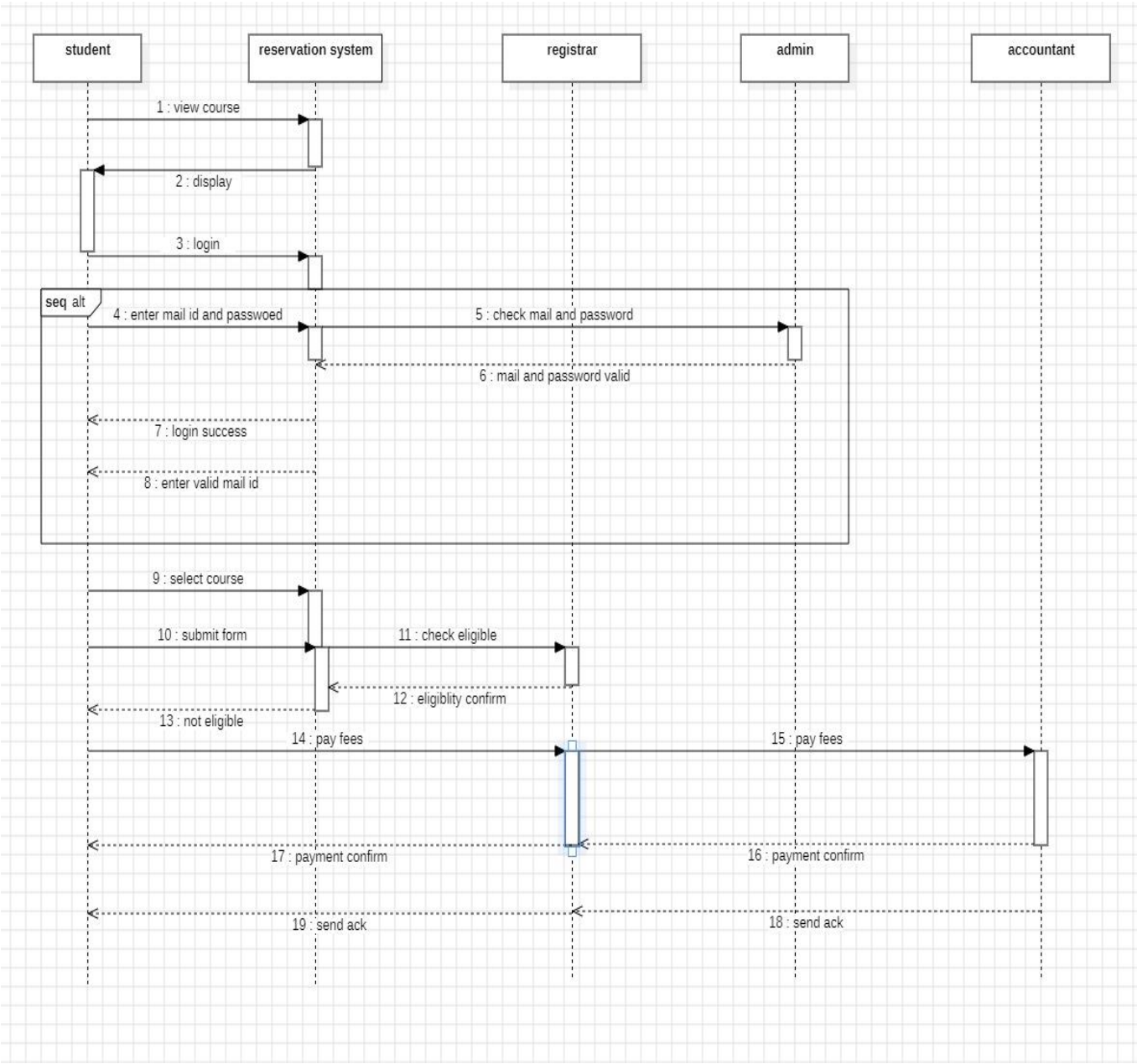
i) To capture dynamic behaviour of a system.

ii)To describe the message flow in the system.

iii)To describe structural organization of the objects.

iv)To describe interaction among objects.

Sequence diagram for online course reservation system:



Collaboration Diagram:

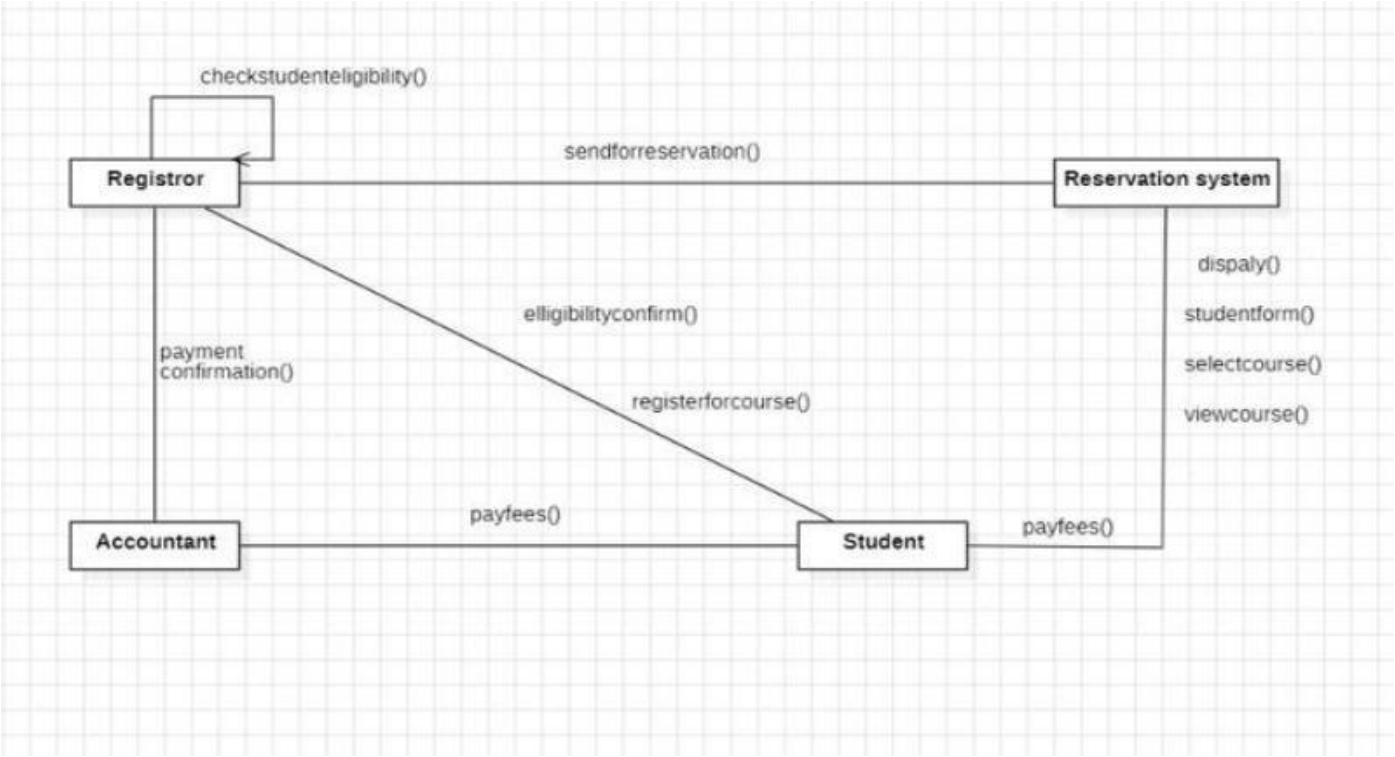
Communication diagram illustrate that object interact on a graph or network format in which object can be placed where on the diagram.

In collaboration diagram the object can be placed in anywhere on the diagram.

The collaboration comes from sequence diagram. The collaboration diagram represents the collaboration which is a set of object related to achieve and decide outcome.

In collaboration the sequence is indicated by numbering the messages several numbering schemes are available.

Collaboration diagram for online course reservation system:



Result:

Thus the UML Sequence diagram and Collaboration diagram for Online Course Reservation System was developed successfully.

EX no:06 DATE:	**Draw relevant state chart and activity diagram for the same state**

Aim:

To draw Activity diagram for Online Course Reservation System

Activity diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system.

An activity diagram shows the overall flow of control. An activity is shown as an rounded box containing the name of the operation. This activity diagram describes the behaviour of the system.

An activity diagram is variation or special case or a state machine, in which the states are activities representing a performance of operation and the transition or triggered by the completion of operation.

Activity diagram is similar to state chart diagram where the token represented as an operation. An activity shows as a round box containing name of operation. The concurrent control is indicated by multiple arrows, leaving a synchronization bar represented by short or thick bar with incoming and outgoing arrows. Activity diagram is another important diagram in UML to describe dynamic aspects of the system.

This diagram is basically a flow chart to represent the flow form one activity to another activity.

The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent.

Activity diagrams deals with all type of flow control by using different elements like fork, join etc. The basic purposes of activity diagrams are similar to other four diagrams. It captures the dynamic behaviour of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward

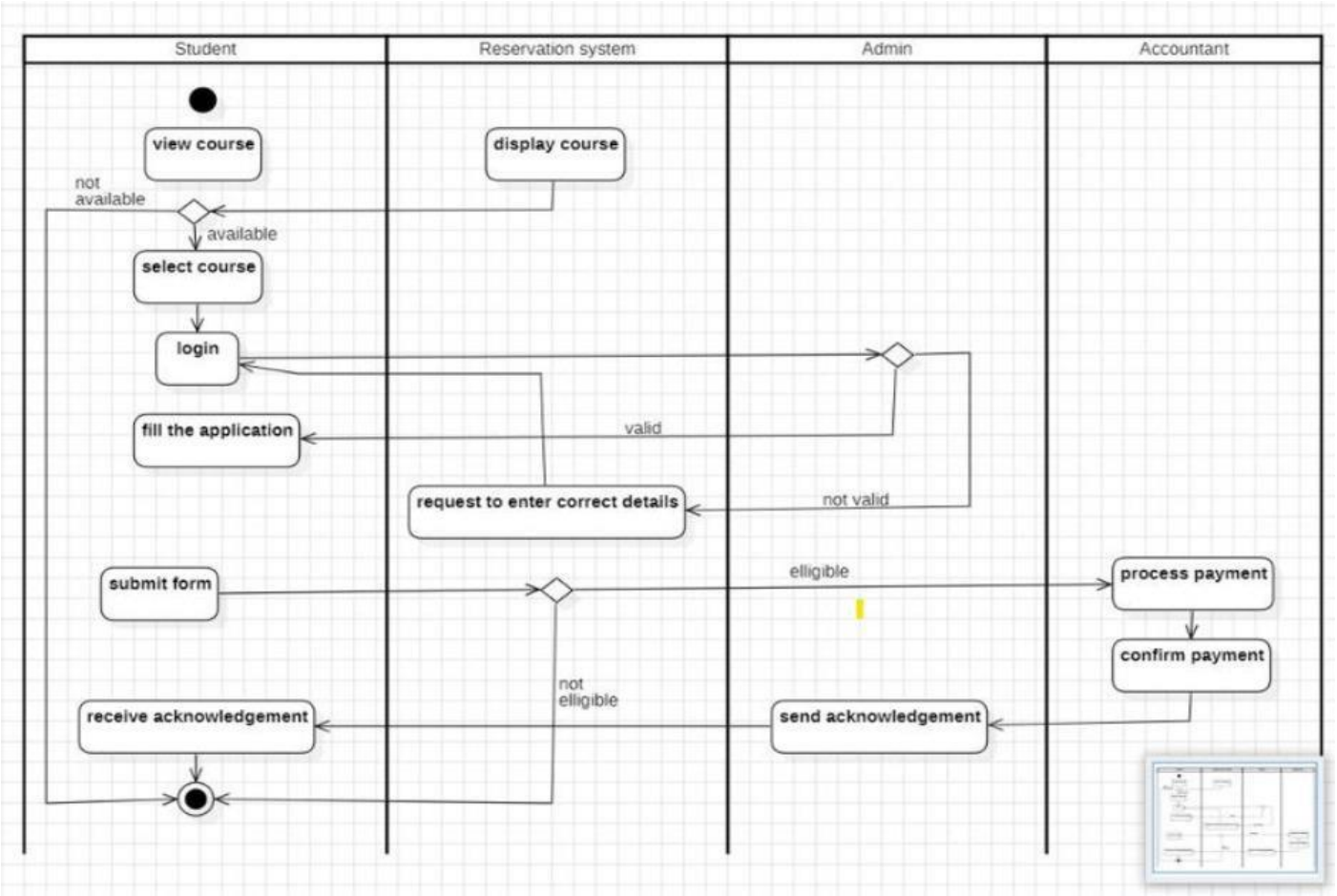
and reverse engineering techniques. The only missing thing in activity diagram is the message part. It does not show any message flow from one activity to another.

Activity diagram is some time considered as the flow chart. Although the diagrams looks like a flow chart but it is not. It shows different flow like parallel, branched, concurrent and single.

The purposes can be described as:

- Draw the activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and concurrent flow of the system.

State chart diagram for online course reservation system:



Result:

Thus the activity diagram for online course reservation system was developed successfully.

EX no:07 DATE:	Implement the system as per the detailed design

Aim:

To implement the Online Course Reservation System as per the detailed design.

Implementation for online course reservation system:

User.java

```
public class User
 {
 public String Username;
public String Password;
public CourseDetails enters;
public Reservation fill;
public void loginD{
}
}
```

Status.java

```
public class Status
 {
 public String Name;
public String CourseName;
 public Integer Paidamount;
 public void sendack {
}
}
```

CourseDetails.java

```
public class CourseDetails
{
public String coursename;
public String Coursetype;
 public Integer Courseduration;
```

```
 public void viewCourse {
}
}
```

Reservation.java

```
import java.util.Vector;
 public class Reservation
{
 public String Name;
public Integer Age;
public String DOB; public Integer Marks;
public String ParentDetails;
public String Coursename;
 public Vector generates;
 public void fillApplication {
}
public void validate
}
}
```

Payment.java

```
import java.util.Vector;
 public class Payment
 {
 public Integer Amount;
public String ModeofPayment;
public Vector generates;
 public void payMethod{
}
}
```

NetBanking.java

```
public class NetBanking extends Payment
{
 public String Userid;
```

```
public String Password;
 public void pay(){}
}
}
```

Result:

Thus the implementation of Online Course Reservation System was executed and the codes were generated successfully.

EX no:08 DATE:	**Test the software system for all scenarios identified as per the use case diagram**

Aim:

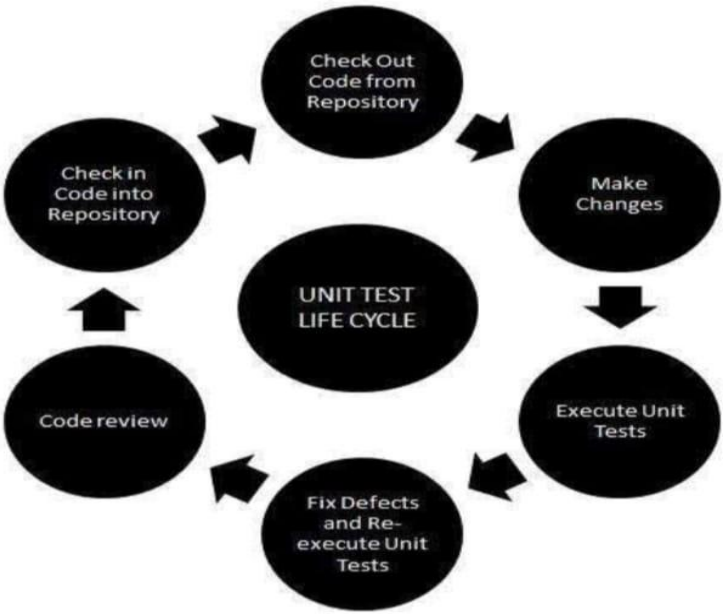
To test the software system for all scenarios identified in the use case diagram.

Online Course Reservation System:

In the domain of education, efficient management of course reservations is essential for ensuring smooth operations and student satisfaction. An Online Course Reservation System simplifies the process of browsing available courses, making reservations, managing schedules, handling payments, and communicating with students**.**

Unit Testing:

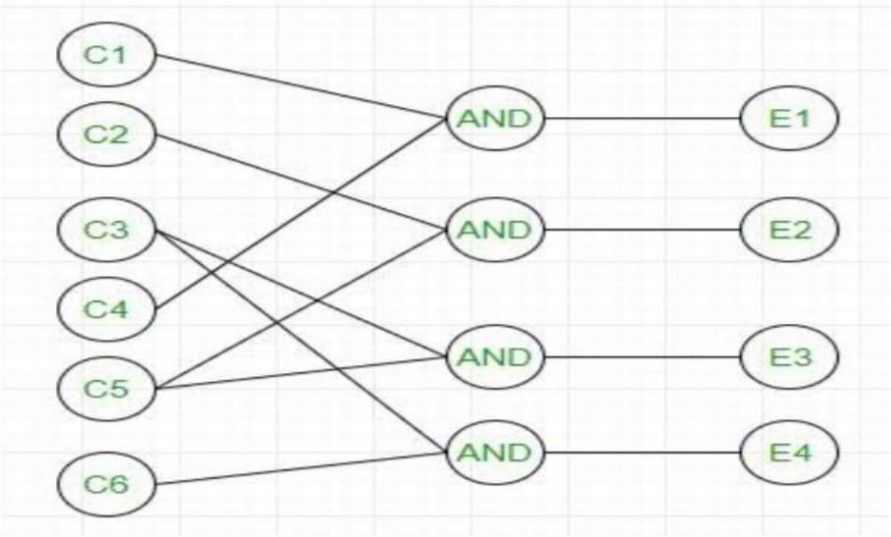
- Unit testing involves testing individual components or modules of the software to ensure they function correctly in isolation.
- Each module, such as client management, project management, employee management, etc., would undergo unit testing.
- Example: Testing the "addClient()" function to ensure that a new client is successfully added to the system.



Black Box Testing:

- Black box testing is a technique where the internal workings of the system are not known to the tester. The tester only tests the system's functionality based on its specifications.
- Testers would input various sets of data into the system and verify that the expected output is produced.

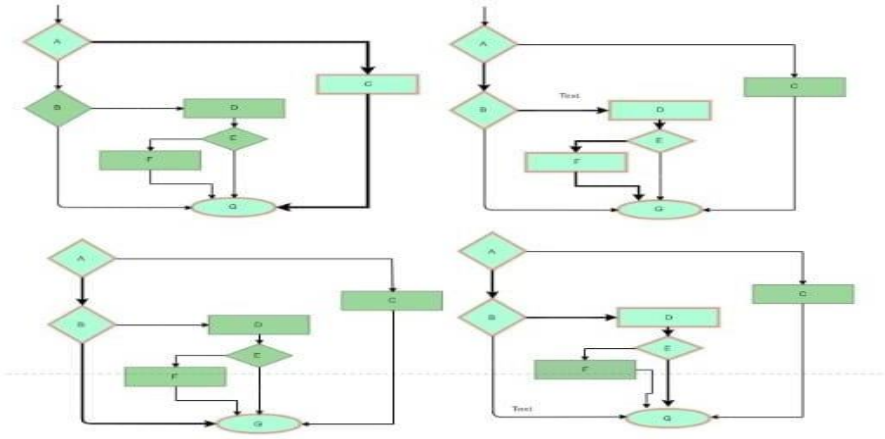
Example: Testing the "searchForJob()" functionality to ensure that it returns the expected results based on different search criteria



		1	2	3	4
CAUSES	C1	1	0	0	0
	C2	0	1	0	0
	C3	0	0	1	1
	C4	1	0	0	0
	C5	0	1	1	0
	C6	0	0	0	1
EFFECTS	E1	X	-	-	-
	E2	-	X	-	-
	E3	-	-	X	-
	E4	-	-	-	X

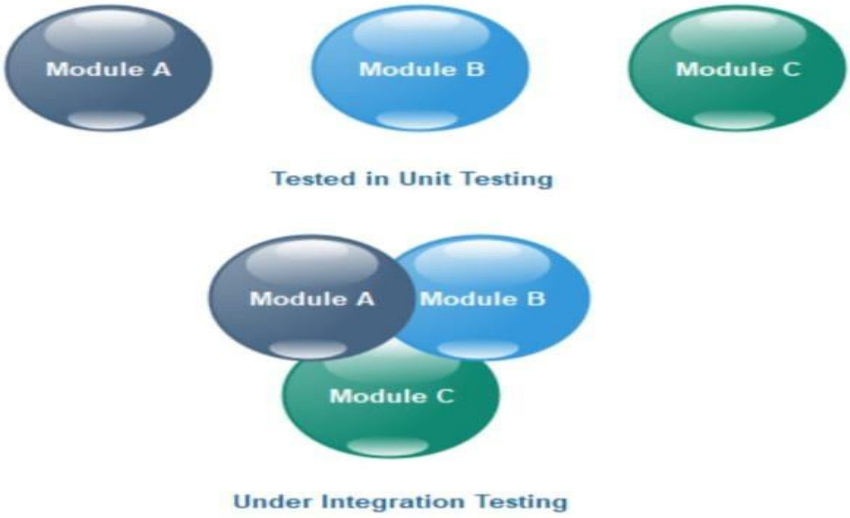
White Box Testing:

- White box testing involves testing the internal logic and structure of the software code.
- Testers would examine the code of individual modules to ensure that all code paths are tested.
- Example: Testing the "performQC()" function to ensure that it adequately checks the quality of the processed data.



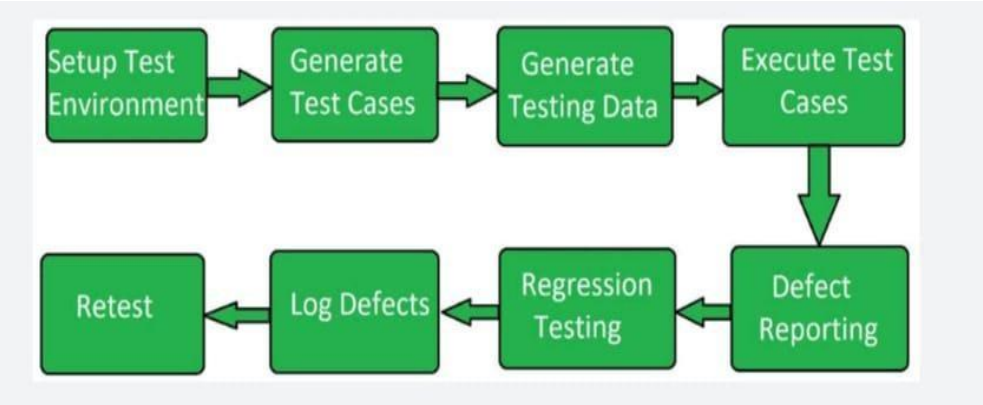
Integration Testing:

- Integration testing verifies that different modules of the software work together as expected.
- Testers would test the interaction between different modules, such as client management, project management, and employee management.
- Example: Testing the interaction between the "addClient()" and "addProject()" functions to ensure that a project can be associated with a client successfully.



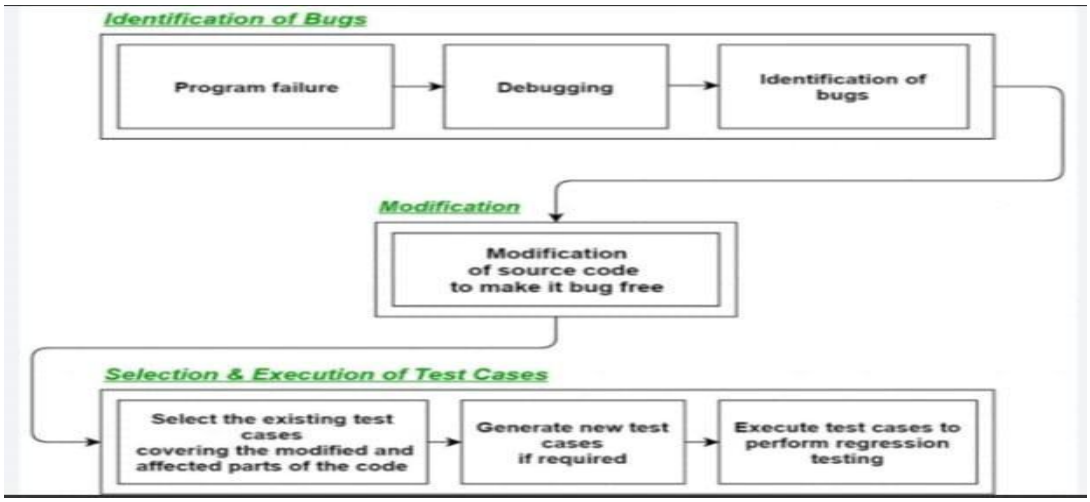
System Testing:

- System testing involves testing the entire system as a whole to verify that it meets the specified requirements.
- Testers would perform end-to-end testing of the entire system, including all modules and their interactions.
- Example: Testing the entire process from client negotiation to project delivery and payment to ensure that it functions as expected.



Regression Testing:

- Regression testing ensures that new changes or additions to the system do not adversely affect existing functionalities.
- Testers would re-run previously conducted tests after new changes or additions are made to the system to ensure that existing functionalities are not affected.
- Example: After adding a new feature to the system, testers would re-run all existing tests to ensure that the new feature did not introduce any bugs or errors.



Result:

Thus the software system for all scenarios identified in the use case diagram was tested successfully.

EX no:09 DATE:	**Improve the reusability and maintainability of the software by applying appropriate design patterns**

Aim:

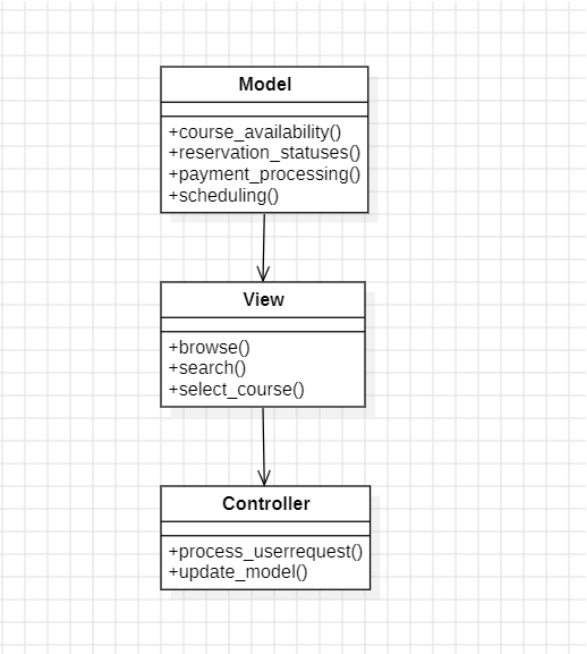
To improve the reusability and maintainability of the software system by applying appropriate design patterns.

Online course reservation system:

In an Online Course Reservation System, employing design patterns is vital for enhancing its flexibility and maintainability. The Factory Method Pattern facilitates the creation of diverse course instances uniformly, accommodating various types of courses. The Observer Pattern ensures real-time updates across modules, keeping users informed of changes in course availability or details. Leveraging the Decorator Pattern allows for the augmentation of course reservations with additional features, tailoring the user experience. The Composite Pattern simplifies the representation of hierarchical course structures, aiding in efficient management. Lastly, the Singleton Pattern ensures critical system components maintain singular instances, optimizing resource utilization and consistency throughout the system.

Model-View-Controller(MVC):

- In an Online Course Reservation System, the Model represents course data and reservation-related business logic. It manages operations such as course availability, reservation statuses, payment processing, and scheduling.
- The View component displays course information and provides interfaces for users to browse, search, and select courses for reservation. It presents screens or pages where users can view course details, availability, and pricing, facilitating the reservation process.
- The Controller processes user requests and coordinates actions between the View and Model in the system. It interprets user input, initiates corresponding actions such as course selection or reservation confirmation, and updates the Model based on user interactions, ensuring a seamless reservation experience for users.



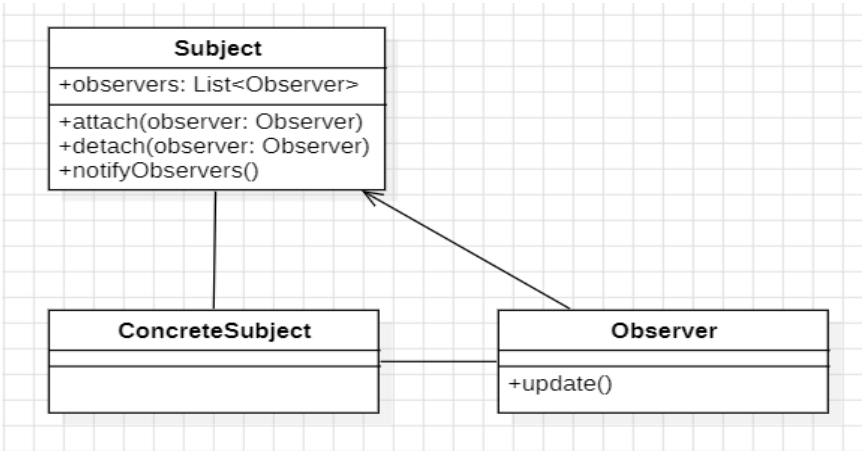
Observer Pattern:

- The Observer Pattern notifies components about system changes, enhancing flexibility and scalability. It enables easy addition of new notifications, improving system maintainability.

❖ Context: Used when objects need to be notified of changes in another object's state.

❖ Problem: Establishes a one-to-many relationship without tightly coupling objects, ensuring automatic updates.

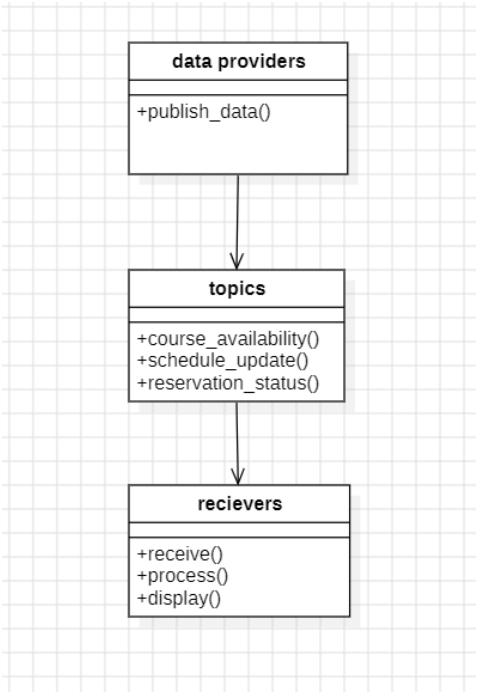
❖ Solution: Define a subject interface with methods for attaching, detaching, and notifying observers. Observers register with subjects to receive updates on state changes.



Publish-Subscribe Pattern:

The Publish-Subscribe Pattern plays a crucial role in facilitating real-time updates and notifications across various modules within an Online Course Reservation System.

- **Context:** Real-time updates are imperative for maintaining the integrity and functionality of an Online Course Reservation System, especially concerning dynamic changes in course availability, schedule updates, and reservation statuses.
- **Problem:** Without implementing the Publish-Subscribe Pattern, the system may suffer from data latency and operational inefficiencies when distributing real-time course information to users.
- **Solution:** By incorporating the Publish-Subscribe Pattern, the system can efficiently disseminate course-related updates to subscribing users and modules. This approach enables timely broadcasting of relevant information, bolstering the system's scalability, adaptability, and responsiveness.



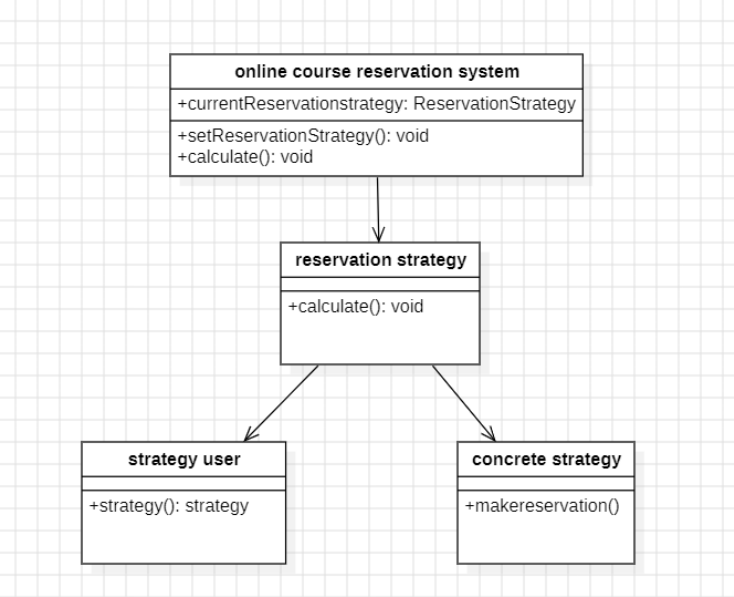
Strategy Pattern:

The Strategy Pattern proves instrumental in managing various algorithms for course reservation strategies within an Online Course Reservation System. By implementing this pattern, the system facilitates interchangeable reservation algorithms, offering users the flexibility to choose the most suitable booking strategy.

Context: Utilized to manage a range of algorithms interchangeably for course reservation purposes, accommodating diverse user preferences and system requirements.

Problem: The need arises for dynamic selection and switching of reservation algorithms, promoting code flexibility, and ensuring reusability across different reservation scenarios.

Solution: Define an interface for reservation strategies with algorithmic methods tailored to various booking approaches. Implement concrete strategy classes for each reservation method, allowing for seamless switching between them at runtime. This approach enables adaptability and customization in the course reservation process, catering to the evolving needs and preferences of users.



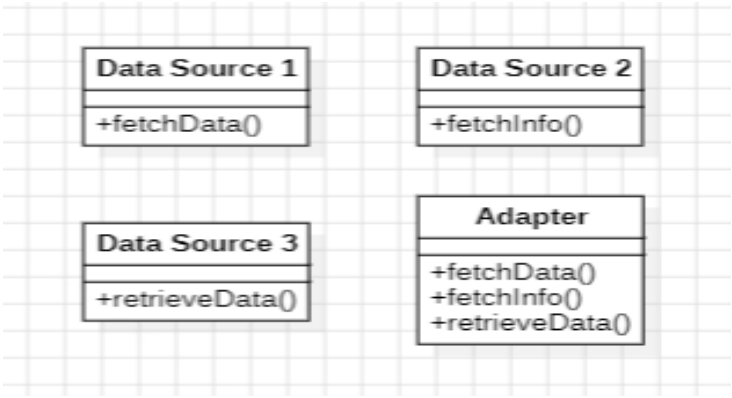
Adapter Pattern:

Adapter Pattern in the context of an Online Course Reservation System is instrumental in integrating diverse external systems or APIs with varying interfaces seamlessly. This pattern resolves the challenge of incompatible interfaces between the reservation system and external services by providing a wrapper that translates different data formats and protocols. By adopting this pattern, the system ensures smooth communication and data exchange, thereby enhancing interoperability and flexibility within the reservation ecosystem.

Context: Integrating external systems or APIs with varying interfaces is a common requirement within online course reservation systems.

Problem: Incompatible interfaces between these external systems or APIs and the reservation system result in communication obstacles and data parsing discrepancies.

Solution: The Adapter Pattern addresses this challenge by providing a wrapper that translates data formats and protocols, enabling seamless integration without compromising data integrity or communication efficiency.



.

Result:

Thus the reusability and maintainability of the software system by applying appropriate design patterns was executed successfully