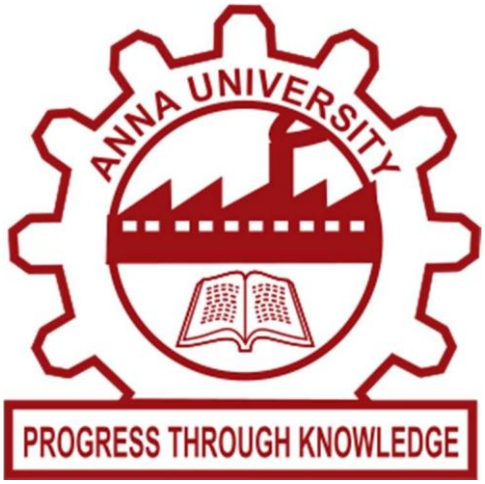
# UNIVERSITY COLLEGE OF ENGINEERING NAGERCOIL

**(ANNA UNIVERSITY CONSTITUENT COLLEGE)**

**KONAM, NAGERCOIL – 629 004**




## RECORD NOTE BOOK

**CCS356-OBJECT ORIENTED SOFTWARE ENGINEERING**

**REGISTER NO :**

**NAME :**

**YEAR/SEMESTER :**

**DEPARTMENT :**

# UNIVERSITY COLLEGE OF ENGINEERINGNAGERCOIL

**(ANNA UNIVERSITY CONSTITUENT COLLEGE)**

**KONAM, NAGERCOIL – 629 004**

**Register No:**

*Certified that, this is the bonafide record of work done by* ***Mr./Ms.*** *…… of VI Semester in Computer Science and Engineering of this college, in the* ***CCS356 – OBJECT ORIENTED SOFTWARE ENGINEERING*** *During the academic year 2023-2024 in partial fulfillment of the requirements of the B.E Degree course of the Anna University Chennai.*

**Staff-in-charge** **Head of the Department**

This record is submitted for the University Practical Examination held on .....

**Internal Examiner** **External Examiner**

# INDEX

| Exp No | Date | Title | Page | Sign |
|---|---|---|---|---|
| 1. | | Identify a software system that needs to be developed | | |
| 2. | | Document the Software Requirements Specification (SRS) for the Exam Registration System | | |
| 3. | | Identify the use cases and the develop the Use Case model | | |
| 4. | | Identify the conceptual classes and develop a Domain Model and also derive a Class diagram for the Exam Registration System | | |
| 5. | | Using the identified scenarios, find the interaction between objects and represent them using UML Sequence and Collaboration diagram | | |
| 6. | | Draw relevant State Chart and Activity Diagram for the Exam Registration System | | |
| 7. | | Implement the system as per the detailed design | | |
| 8. | | Test the software system for all the scenarios identified as per the use case diagram | | |
| 9. | | Improve the reusability and maintainability of the software system by applying appropriate design patterns | | |

| Ex.No:01 | Identify a Software system that needs to be developed |
|---|---|
| | |

**AIM:**

To identify a software system that needs to be developed Exam registration System.

**INTRODUCTION:**

Exam Registration System is an interface between the Student and the Exam Controller responsible for theIssue of Hall Ticket. It aims at improving the efficiency in the Issue of Hall ticket and reduces the complexitiesinvolved in it to the maximum possible extent.

**FEATURES OF EXAM REGISTRATION SYSTEM:**

**1.User Authentication and Authorization:**

Users should be able to create accounts, log in securely, and have appropriate access levels based on theirroles (e.g., student, administrator, instructor).

**2.Exam Management:**

This feature allows administrators to create, update, and manage exam details such as exam dates, times,locations, duration, eligibility criteria, and any prerequisites.

**3.Course and Class Management:**

Integration with course and class management systems to ensure that students can register for exams relevantto their enrolled courses and classes.

**4.Registration Process:**

A user-friendly registration process that allows students to browse available exams, select the ones they wantto register for, and pay any associated fees.

**5.Payment Integration:**

Integration with payment gateways to facilitate secure online payments for exam registration fees.

**6.Real-time Availability:**

The system should display real-time information about available seats for each exam, allowing students tomake informed decisions during the registration process.

**7.Waitlist Management:**

In case an exam reaches its capacity, a waitlist feature should allow students to join a queue and receivenotifications if seats become available.

**8.Email Notifications:**

Automated email notifications to confirm registration, provide exam details, remind students of upcomingexams, and notify them of any changes or cancellations.

**9.Exam Schedule Generation:**

Automatic generation of exam schedules to avoid conflicts and ensure that students do not have overlappingexams.

**10.Reporting and Analytics:**

Comprehensive reporting capabilities to track registration trends, exam attendance rates, revenue generated,and other relevant metrics. This data can help administrators make informed decisions and optimize the examregistration process.

**11.Security Measures:**

Implementation of security measures such as encryption of sensitive data, protection against unauthorizedaccess, and regular security audits to ensure compliance with data protection regulations**.**

**12.Scalability and Performance:**

The system should be designed to handle a large volume of concurrent users and transactions efficiently,ensuring optimal performance during peak registration periods.

**RESULT:**

Thus the software system that is need to be developed for Exam Registration System was executed successfully.

| Ex.No:02 | Document the Software Requirements Specification (SRS) For Exam Registration System |
|---|---|
| | |

**AIM:**

To implement a software for Exam Registration System.

**PROBLEM STATEMENT:**

Exam Registration system.is used in the effective dispatch of registration form to all of the students. This system adopts a comprehensive approach to minimize the manual work and schedule resources, time in a cogent manner. The core of the system is to get the online registration form (with details such as name, reg.no etc.,) filled by the student whose testament is verified for its genuineness by the Exam Registration System with respect to the already existing information in the database. This forms the first and foremost step in the processing of exam application. After the first round of verification done by the system, the information is in turn forwarded to the Exam Controller. The application is then processed manually based on the report given by the system. The system also provides the student the list of exam dates.The controller will be provided with fees details to display the current status of application to thestudent, which they can view in their online interface. After all the necessary criteria has been met, the originalinformation is added to the database and the hall ticket is sent to the student.

**SOFTWARE REQUIREMENT SPECIFICATION:**

**1.0 INTRODUCTION**

Exam Registration System is an interface between the Student and the Exam Controller responsible for the Issue of Hall Ticket. It aims at improving the efficiency in the Issue of Hall ticket and reduces the complexities involved in it to the maximum possible extent.

**1.1 PURPOSE**

If the entire process of 'Issue of Hall ticket' is done in a manual manner then it would takes several days for the hall ticket to reach the student. Considering the fact that the number of students for hallticket is increasing every year, an Automated System becomes essential to meet the demand. So this systemuses several programming and database techniques to elucidate the work involved in this process.

## 1.2 SCOPE

- The System provides an online interface to the user where they can fill in their personal details andsubmit the necessary documents (may be by scanning).
- The controller concerned with the issue of hall ticket can use this system to reduce his workload andprocess the application in a speedy manner.
- Students will come to know their status of application and the date in which they must subjectthemselves for manual document verification.

## 1.3 DEFINITIONS, ACRONYMS AND THE ABBREVIATIONS

- **Exam Controller -** Refers to the super user who is the Central Authority who has been vested withthe privilege to manage the entire system.
- **Student** - One who wishes to obtain the Hall Ticket.
- **ERS** - Refers to this Examination Registration System.
- **HTML -** Markup Language used for creating web pages.
- **J2EE** – Java 2 Enterprise Edition is a programming platform java platform for developing andrunning distributed java applications.
- **HTTP** - Hyper Text Transfer Protocol.
- **TCP/IP** – Transmission Control Protocol/Internet Protocol is the communication protocol used toconnect hosts on the Internet.

## 1.4 REFERENCES

IEEE Software Requirement Specification format.

## 1.5 TECHNOLOGIES TO BE USED

- HTML
- JSP
- JavaScript
- Java

## 1.6 TOOLS TO BE USED

- Eclipse IDE (Integrated Development Environment)
- Rational Rose tool (for developing UML Patterns)

## 1.7 OVERVIEW

SRS includes two sections overall description and specific requirements - Overall Description will describemajor role of the system components and inter-connections. Specific Requirements will describe roles & functions of the actors.

## 2.0.OVERALL DESCRIPTION

### 2.1 PRODUCT PERSPECTIVE

The ERS acts as an interface between the 'student' and the 'exam controller'. This system tries to make the interface as simple as possible and at the same time not risking the security of data stored in. This minimizes thetime duration in which the user receives the hall ticket.

### 2.2 SOFTWARE INTERFACE

- Front End Client - The exporter online interface is built using JSP and HTML.
- Web Server – Apache Tomcat Server (Oracle Corporation)
- Back End - Oracle 11g database

### 2.3 HARDWARE INTERFACE

The server is directly connected to the client systems. The client systems have access to the database in theserver.

### 2.4 SYSTEM FUNCTIONS

- Secure Registration of information by the Students.
- SMS and Mail updates to the students by the controller.
- Controller can generate reports from the information and is the only authorized personnel to addthe eligible application information to the database.

### 2.5 USER CHARACTERISTICS

- **Student** - They are the people who desire to obtain the hall ticket and submit the information tothe database.
- **Exam controller** - He has the certain privileges to add the registration status and to approve theissue of hall ticket. He may contain a group of persons under him to verify the documents and give suggestionwhether or not to approve the dispatch of hall ticket.

### 2.6 CONSTRAINTS

- The applicants require a computer to submit their information.
- Although the security is given high importance, there is always a chance of intrusion in the web worldwhich requires constant monitoring.
- The user has to be careful while submitting the information. Much care is require

### 2.7ASSUMPTIONS AND DEPENDENCIES

- The Students and Exam Controller must have basic knowledge of computers and EnglishLanguage.
- The student may be required to scan the documents and send.

## 2.8 FUNCTIONAL REQUREMENTS

1. **User Registration:**
    - A new student creates an account on the system, providing their personal details.
    - The student logs in to their account using a secure authentication process.
2. **Exam Registration:**
    - The student browses the list of available exams.
    - The student selects an exam and completes the registration process, including payment of the registration fee.
    - The system sends a confirmation email/SMS to the student.
3. **Exam Management:**
    - An administrator creates a new exam, specifying all required details.
    - The administrator updates the exam details if any changes are needed.
    - The administrator views reports on the number of students registered for each exam.
4. **Payment Processing:**
    - The system calculates the fee for an exam registration.
    - The student completes the payment using an integrated payment gateway.
    - The system confirms the payment and updates the student's registration status.
5. **Notifications:**
    - The system sends an exam reminder to the student one week before the exam date.
    - The administrator sends a custom notification to all students registered for a particular exam.

## 2.9 NON FUNCTIONAL REQUREMENTS

**1.Scalability and Performance**

- Use of Load Balancers: Distribute incoming requests across multiple servers.
- Caching: Implement caching mechanisms (e.g., Redis, Memcached) to reduce database load and improve response times.

**2.Security**

- OAuth2 for Authentication: Use OAuth2 for secure user authentication and authorization
- TLS for Encryption: Use TLS to encrypt data in transit.

**3. Availability and Reliability**

- Cloud Infrastructure: Deploy the system on a cloud platform (e.g., AWS, Azure) that offers high availability and redundancy.
- Database Replication: Use database replication to ensure data availability and reliability.

**RESULT**:

Thus the Software Requirement Specification (SRS) document was implemented successfully

| Ex.No:03 | IDENTIFY THE USE CASES AND DEVELOP THE USE CASE MODEL |
|---|---|
| | |

**Aim:**

To identify the use cases and develop the use case model for Exam Registration system.

**Introduction of Use case diagram:**

A Use Case Diagram is a vital tool in system design, it provides a visual representation of how users interact with a system. It serves as a blueprint for understanding the functional requirements of a system from a user's perspective, aiding in the communication between stakeholders and guiding the development process.

**Purpose:**

The main purpose of a use case diagram is to portray the dynamic aspect of a system. It accumulates thesystem's requirement, which includes both internal as well as external influences. It invokes persons, use cases, and several things that invoke the actors and elements accountable for the implementation of use case diagrams.It represents how an entity from the external environment can interact with a part of the system.

Following are the purposes of a use case diagram given below:

- It gathers the system's needs.
- It depicts the external view of the system.
- It recognizes the internal as well as external factors that influence the system.
- It represents the interaction between the actors.

**Components:**

→**Actors:** An actor represents a role of a user that interacts with the system that you are modeling. The usercan be a human user, an organization, a machine, or another external system.

→**Use Cases:** A use case describes a function that a system performs to achieve the user's goal. A use case must yield an observable result that is of value to the user of the system. Actors. An actor represents a role of auser that interacts with the system that you are modeling.

→**Relationship:** A relationship is a connection between model elements. A UML relationship is a type ofmodel element that adds semantics to a model by defining the structure and behavior between the model elements.

**Benefits:**

**Use case diagrams offer several benefits in software development:**

**1.Communication :** They provide a clear and visual representation of system functionality, making it easierfor stakeholders to understand and communicate requirements.

**2..Analysis :** Use case diagrams help in understanding how users will interact with the system, identifying keyfunctionalities, and defining system boundaries.

**3.Validation :** They assist in validating requirements and ensuring that all necessary functionalities arecovered.

**4.Planning :** Use case diagrams can be used as a basis for defining test cases and planning systemimplementation.

**5.Design** : They aid in designing the system architecture and can be a starting point for more detailed designactivities.

**6.Documentation** : Use case diagrams serve as a valuable document that can be referenced throughout thesoftware development lifecycle

**Use Case diagram for Exam Registration System:**

In a Exam Registration System ,the use case diagram could include actors like “Student” and “SystemDB”. Some potential use cases might be ”Login”, “View exam details”, “Register”, ”Acknowledgement”, and“Fee Processing”.

**Actors:**

- Student
- System DB

**Use cases:**

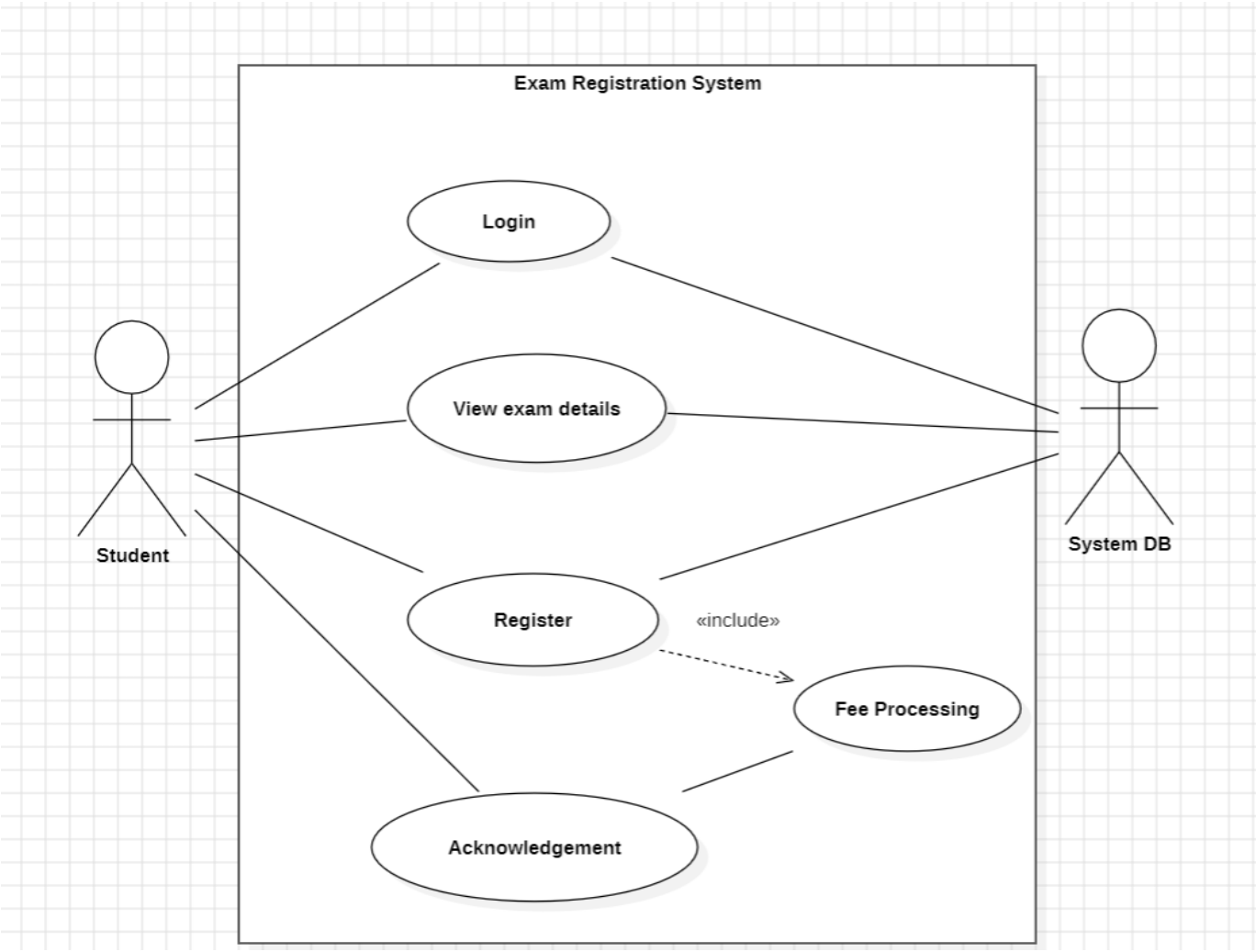
- Login
- View exam details
- Register
- Acknowledgement
- Fee Processing

**Student:**

The "Student" actor initiates actions within the system by triggering specific use cases, such as logging in,registering for exams, viewing schedules, and updating personal profiles.

**System DB:**

The system's database is not typically represented as an actor. Instead, the database is considered an internalcomponent of the system and is not directly interacted with by external actors like students.




**Login:**

For retrieving information the student must have to enter the username and password to login and retrievethe information.

**View Exam Details:**

The student view the details about the exam schedule which contains Date,time,etc… The student can seethe specifics of the test schedule containing the date, time, etc.

**Register:**

The student should notify the fee details that only the student can pay the correct amount. The studentshould tell the fee information that the required amount can only be charged by the student.

**Acknowledgement:**

The exam fees should be paid by the student to get the hall ticket from the exam controller. The exam feesshould be collected by the student to get the roll number from the exam controller.

**Fee Processing**:

All the details should be viewed by both the student and the controller to verify whether all the entered details are correct. Both the Student and controller should review all the data to check if all the informationentered correct.

**Result:**

Thus the Use Case Model was identified and developed successfully.

| Ex.No:04 | Identify the conceptual classes and develop a DomainModel and also derive a class diagram for Exam Registration System |
|---|---|
| | |

**AIM:**

To identify the conceptual classes and develop a domain model and also derive a class diagram for Exam Registration System.

**CLASS DIAGRAM**:

- The UML class diagram is to illustrate class interfaces and their actions. They are used for static objectmodeling, we have already introduced and used their UML diagram while domain modeling.
- A UML class diagram is refered to as object modeling is the main static analysis diagram. The classdiagram 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 systembut also for constructing executable code of the software application.
- The class diagram describes the attributes and operations of a class and also the constraints imposed onthe system.
- The class diagram shows a collection of classes, interfaces, associations, collaborations and constraints.It is also known as a structural diagram.
- The purpose of the class diagram can be summarized as:
    i. Analysis and design of the static view of an application.
    ii. Describe responsibilities of a system.
    iii. Base for component and deployment diagrams.
    iv. Forward and reverse engineering

**Notations:**

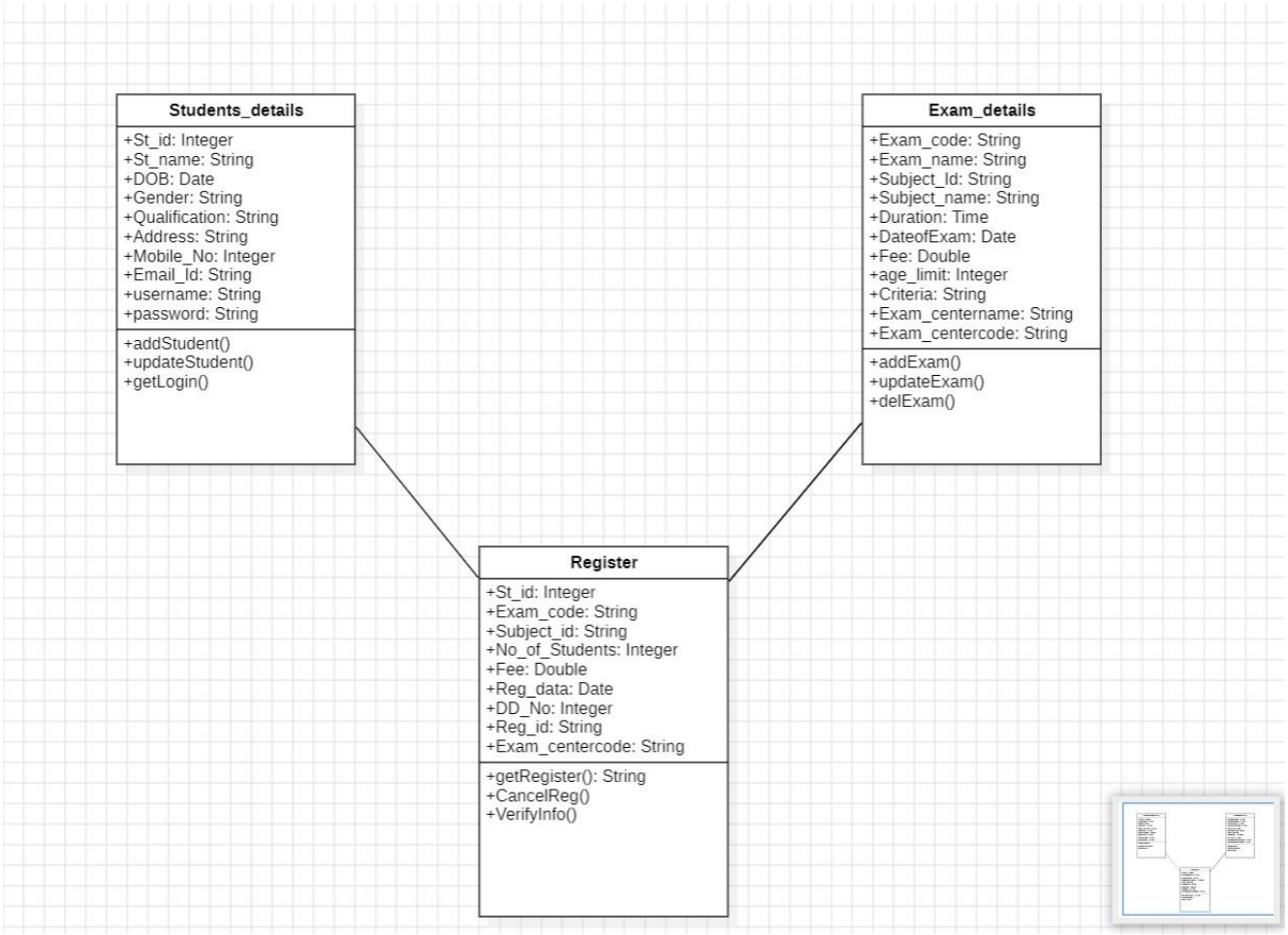
A class is drawn as a rectangle with three component:

1.Top hold class name

2.Middle hold class name

3.Bottome hold list of operation.

## CLASS DIAGRAM FOR ERS:




## RESULT:

Thus the class diagram for Exam Registration System was developed and implemented successfully.

| Ex.No:05 | Using the identified scenarios, find the interaction between using objects and represent them using UML Sequence and Collaboration Diagram for Exam Registration System |
|---|---|
| | |

**AIM:**

To represent the interaction between objects using UML Sequence diagram and Collaboration diagram for Exam Registration 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. 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.

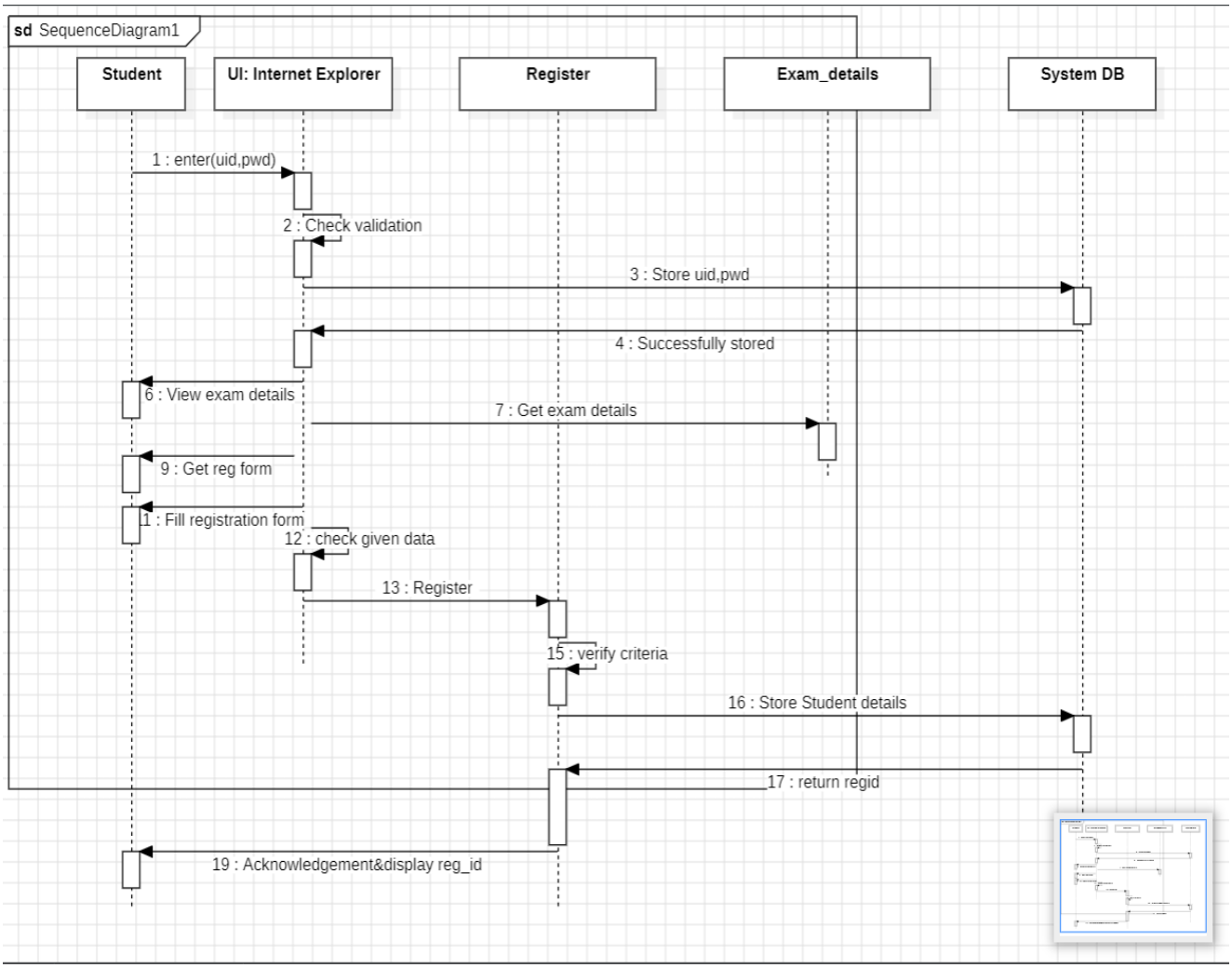
Self call is an message that an object sends to itself by sending messages arrow back to the same lifeline. A sequence diagram illustrates a kind of format in which each object interacts via message. It is generalize between two or more specialized diagram.

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.



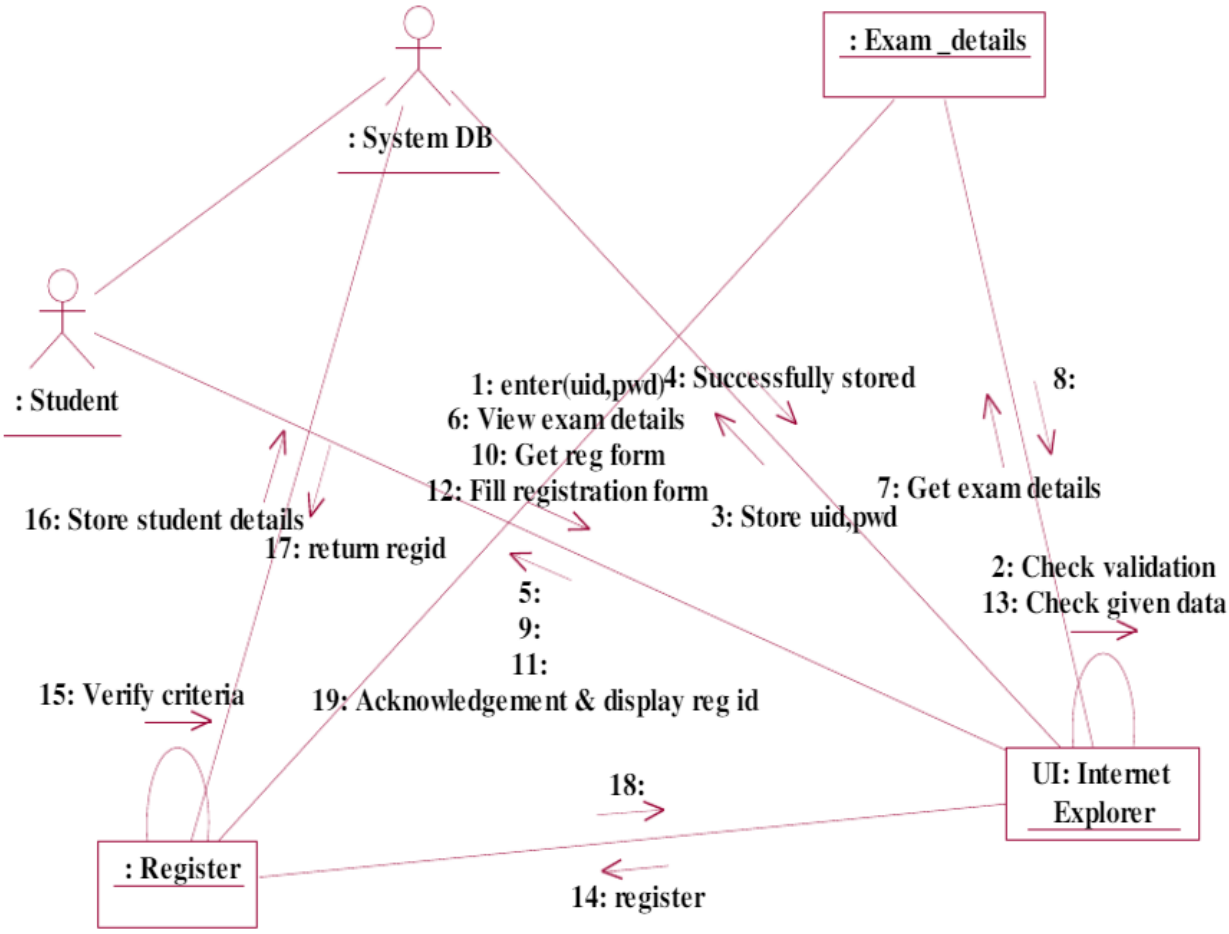

**COLLABORATION DIAGRAM:**

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




**RESULT:**

Thus the UML Sequence diagram and Collaboration diagram for BPO Management System was developed successfully.

| Ex.No:06 | Draw relevant State chart and Activity diagram for the same system |
|---|---|
| | |

**AIM:**

To draw Activity diagram for Exam Registration 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-bystep 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.

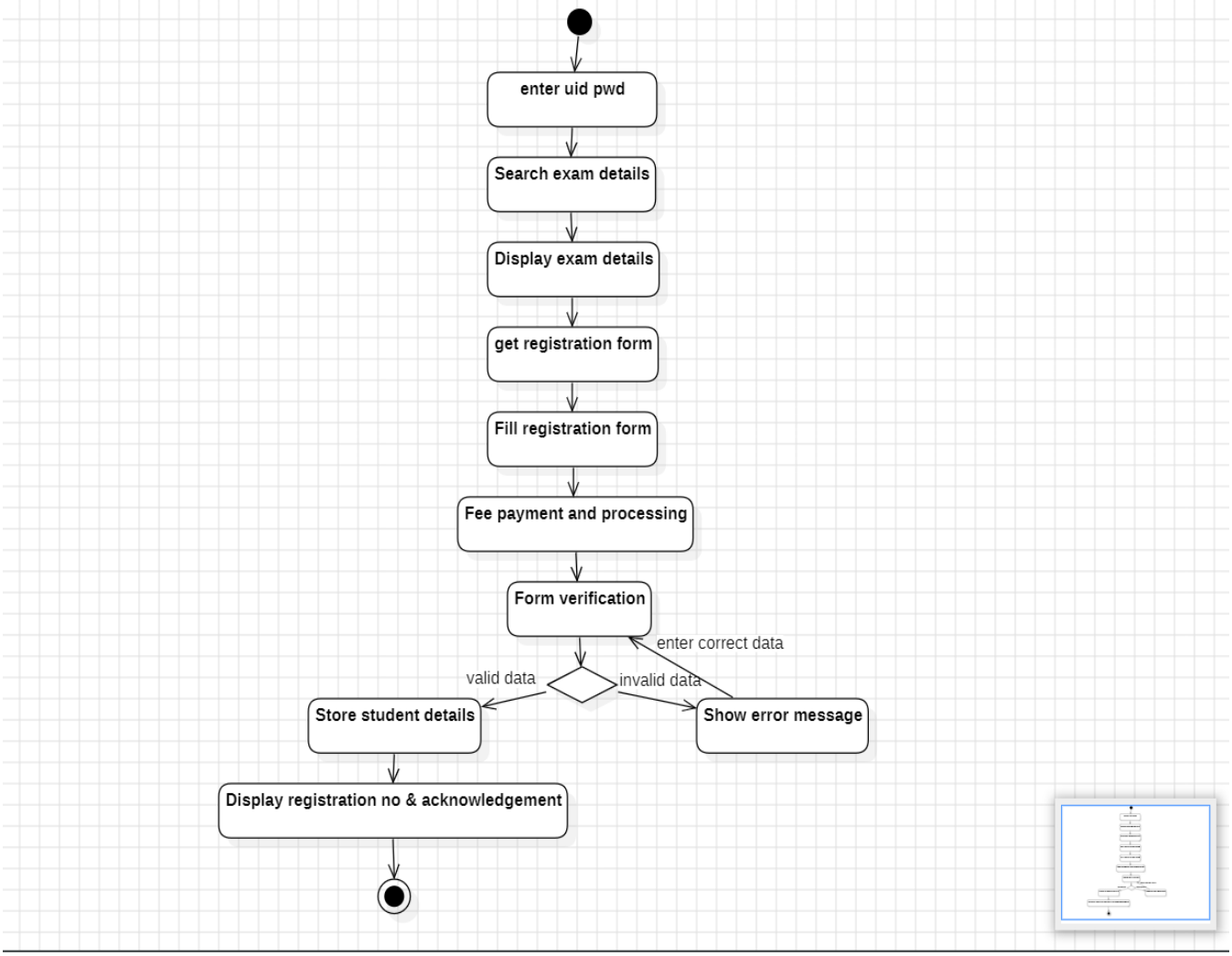
Activity diagrams deals with all type of flow control by using different elements like fork, join etc. The basic purposes of activity diagrams are similar to other four diagrams. It captures the dynamic behaviour of the system. Other four diagrams are used to show the message flow from one object to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

**ACTIVITY DIAGRAM FOR ERS:**




**RESULT:**

Thus the activity diagram for Exam Registration System was developed successfully.

| Ex.No:07 | Implement the System as per the detailed design |
|---|---|

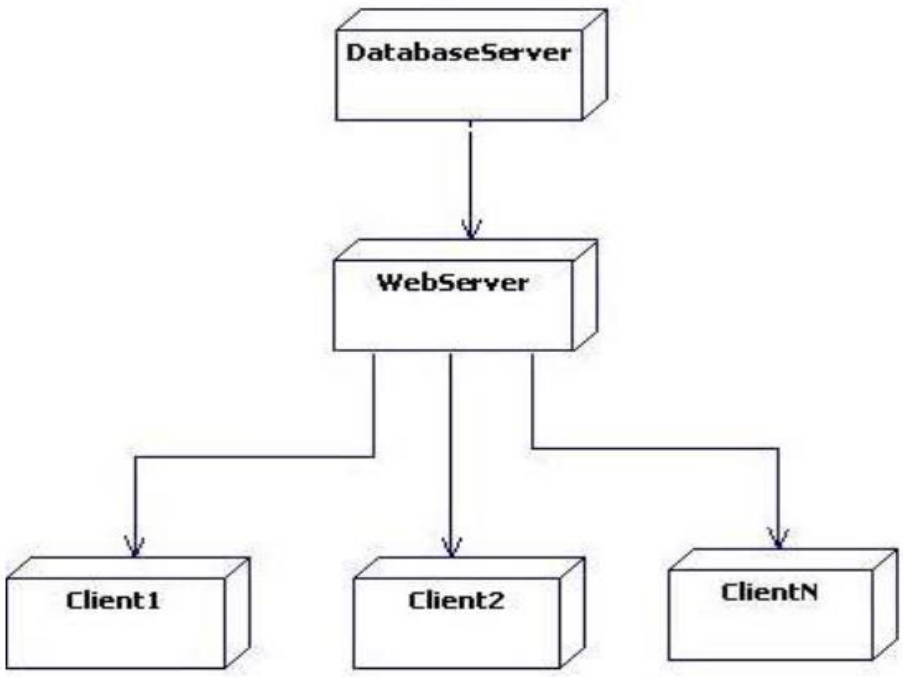
**AIM:**

To implement the Exam Registration System as per the detailed design.

**IMPLEMENTATION DIAGRAM:**

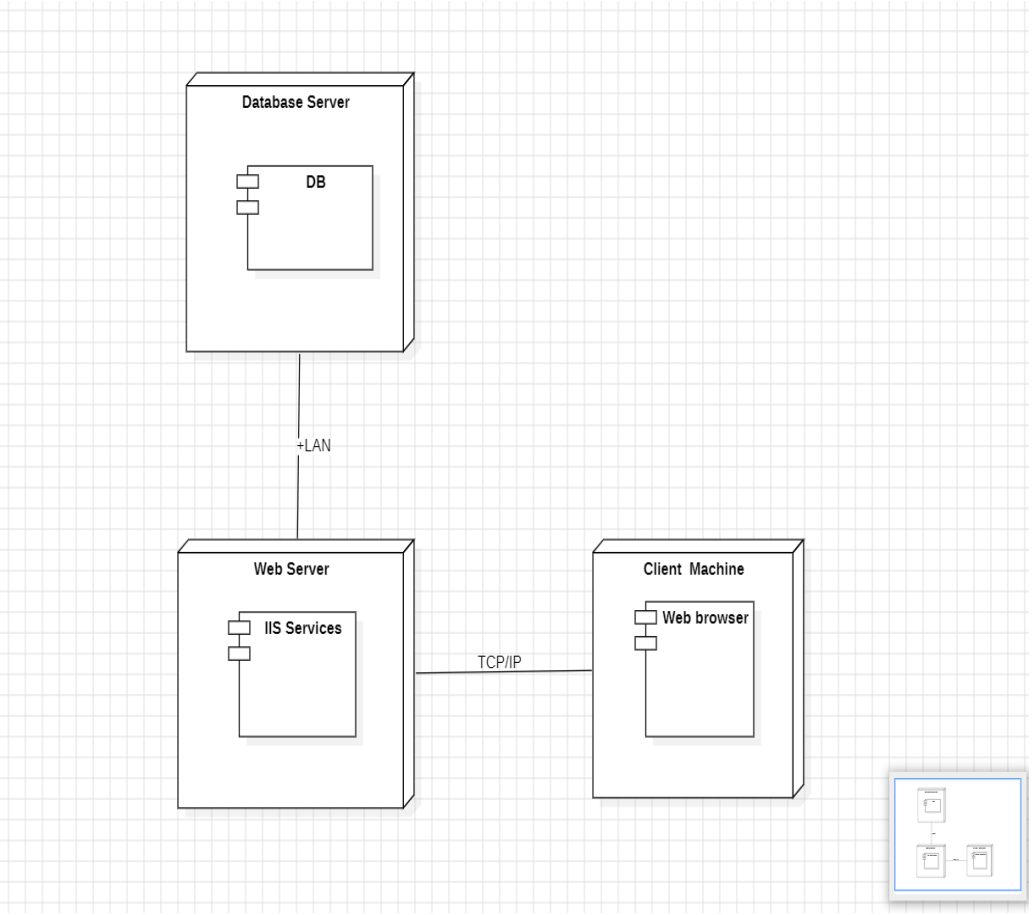
**Deployment Diagram:**

Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed.




**Component Diagram:**

Component diagrams are used to visualize the organization and relationships among components in a system.




**IMPLEMENTATION FOR EXAM REGISTRATION SYSTEM:**

```
public class examDetails
{
  private string examCode;
  private string examName;
  private string subjectId;
  private string subjectName;
  private time duration;
  private date dateOfExam;
  private double fee;
  private integer ageLimit;
  private string criteria;
  private string examCentreName;
  private string examCentreCode;
public studentDetails theStudentDetails;
public register theRegister;
```

```
 public void addExam()
 {
 }
 public void updateExam()
 {
 }
public void delExam()
 {
 }
public class Register
{
  private int studid;
  private String ExamCode;
  private String subid;
  private int no.ofSubject;
  private double fees;
  private string regid;
  private String ExamCenterCode;
  public Register()
 {
 }
  public void getRegister()
 {
 }
  public void cancelRegister()
 {
 }
  public void verifyIngormation()
 {
 }
 public class StudentDetails
{
```

```
  private string Studname;
  private integer Studid;
  private Date DOB;
  private String gender;
  private String qualification;
  private string Address;
  private integer mobileno;
  private string emailid;
  private string username;
  private string password;
  public StudentDetails()
  {
   }
  public void addStudent()
  {
  }
  public void updateStudent()
  {
  }
  public void getLogic()
  {
  }
void studentdetails.getlogin()
 {
 }
studentdetails.studentdetails()
void studentdetails.updatestudent()
 {
 }
void studentdetails.addstudent()
 {
 }
```

**RESULT:**

Thus the implementation of Exam Registration System was executed and the codes were generated successfully

| EX NO:08<br><br>DATE: | **Test the software system for all scenarios identified as per the use case diagram.** |
|---|---|

**Aim:**

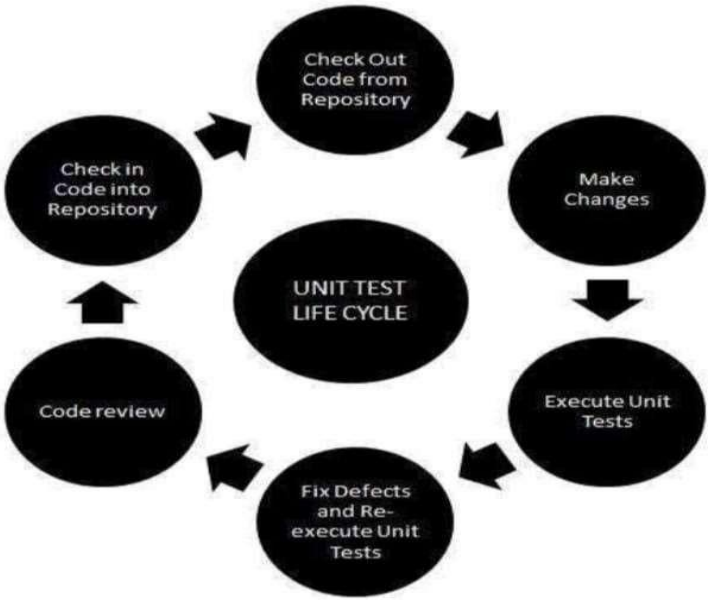
To test the software system for all scenarios identified in the use case diagram.

**Student Information System:**

The Student Information System (SIS) currently in use struggles with manual processes and resource-intensive tasks, hindering the efficient management of student data across educational institutions.

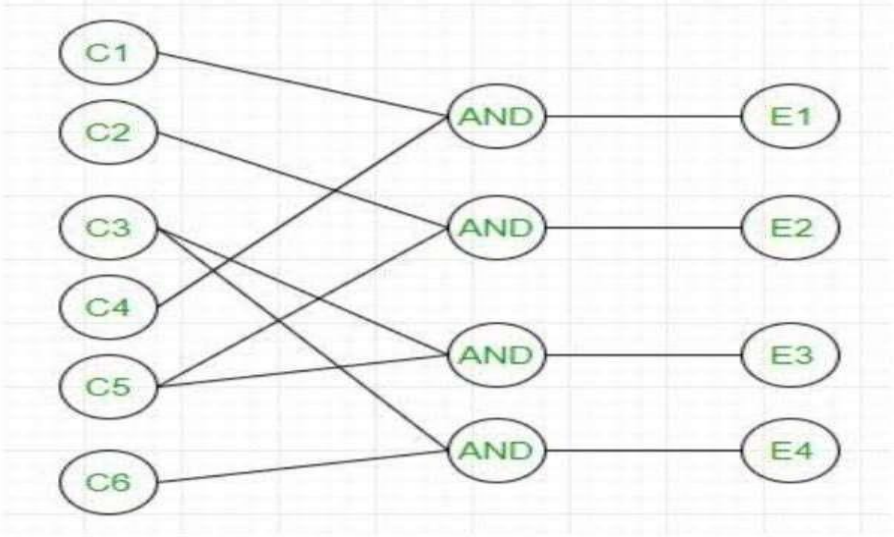
**Unit Testing:**

- 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:**

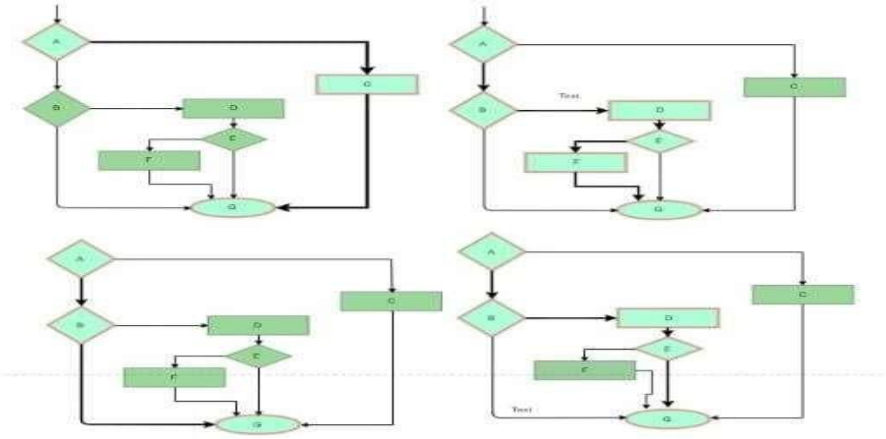
- Black box testing is a technique where the internal workings of the system are not known to the tester. The tester only tests the system's functionality based on its specifications.
- Testers would input various sets of data into the system and verify that the expected output is produced.
- Example: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

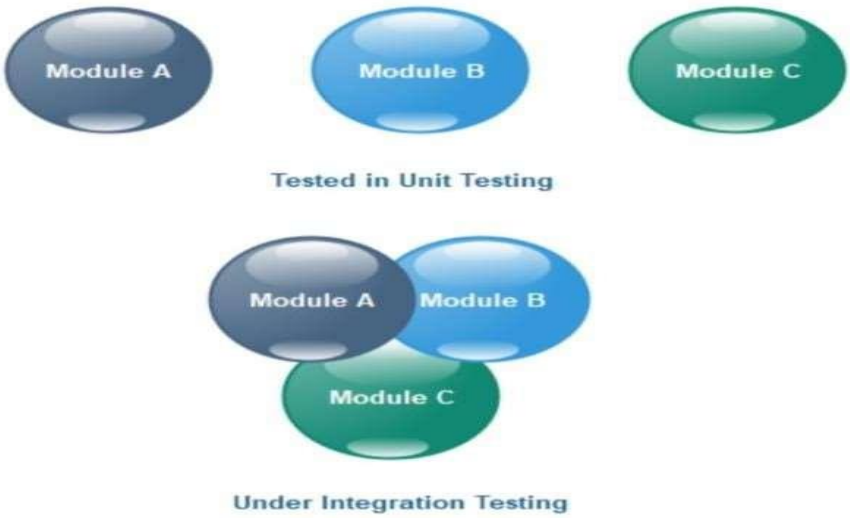
**White Box Testing:**

- 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.



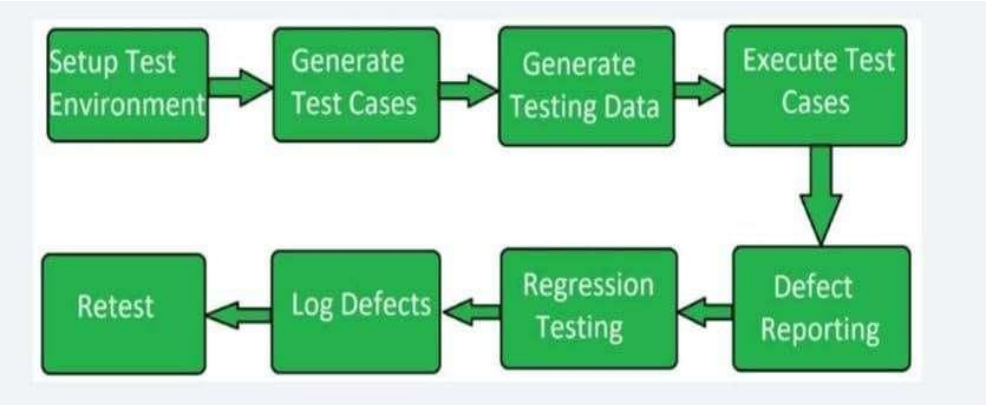
**Integration Testing:**

- 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.



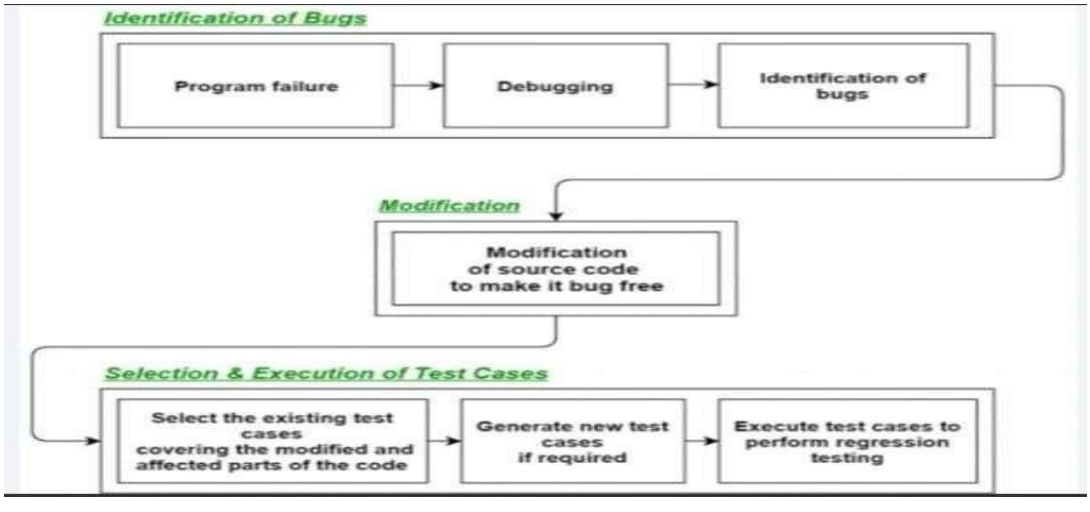
**System Testing:**

- 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 | Improve the Reusability and Maintainability of the Software System by Applying Appropriate Design Patterns |
|---|---|
| | |

**Aim:**

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

**Exam Registration System:**

Exam Registration system.is used in the effective dispatch of registration form to all of the students. This system adopts a comprehensive approach to minimize the manual work and schedule resources, time in a cogent manner. The core of the system is to get the online registration form (with details such as name, reg.no etc.,) filled by the student whose testament is verified for its genuineness by the Exam Registration System with respect to the already existing information in the database.

**1.Model-View-Controller Pattern:**

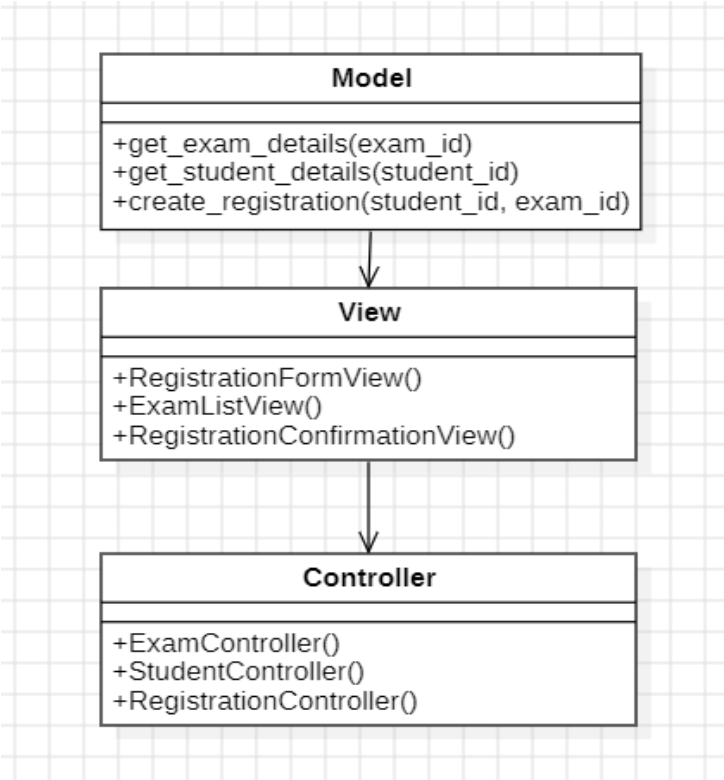
The MVC pattern separates the representation of information from the user's interaction with it. In the Exam Registration System, it divides the system into three interconnected components: Model (data and business logic), View (user interface), and Controller (handles user input and updates the model and view accordingly).

i. **Model:**
   - The Model represents the data and business logic of the application.
   - In the context of an exam registration system, the Model includes classes and components responsible for managing data related to exams, users, registrations, and other entities.

ii. **View:**
   - The View represents the presentation layer of the application.
   - Views are responsible for displaying information to the user and capturing user input.
   - In an exam registration system, Views could include user interfaces for browsing available exams, filling out registration forms, viewing registration status, and displaying exam schedule

### iii. Controller:

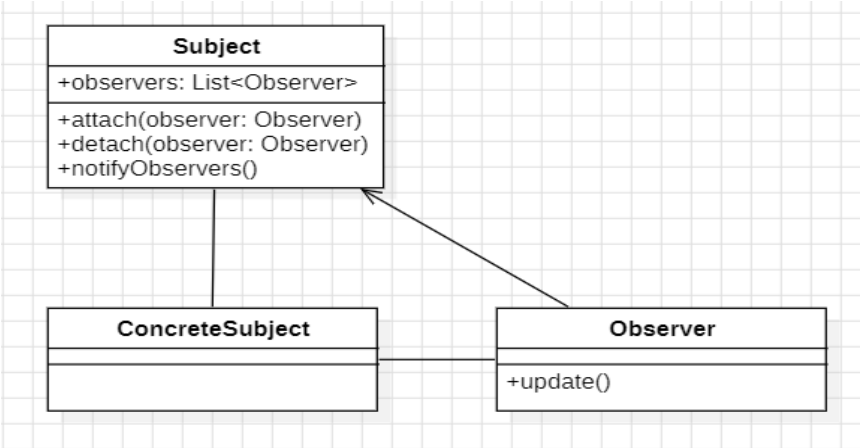
- The Controller acts as an intermediary between the Model and the View.
- It handles user input, processes requests, and updates the Model accordingly.
- In the context of an exam registration system, Controllers interpret user actions from the View (e.g., submitting a registration form, selecting an exam), invoke the appropriate methods in the Model to perform the necessary actions (e.g., saving registration data, retrieving exam information), and update the View to reflect changes (e.g., displaying confirmation messages, refreshing exam listings).




### 2.Observer Pattern:

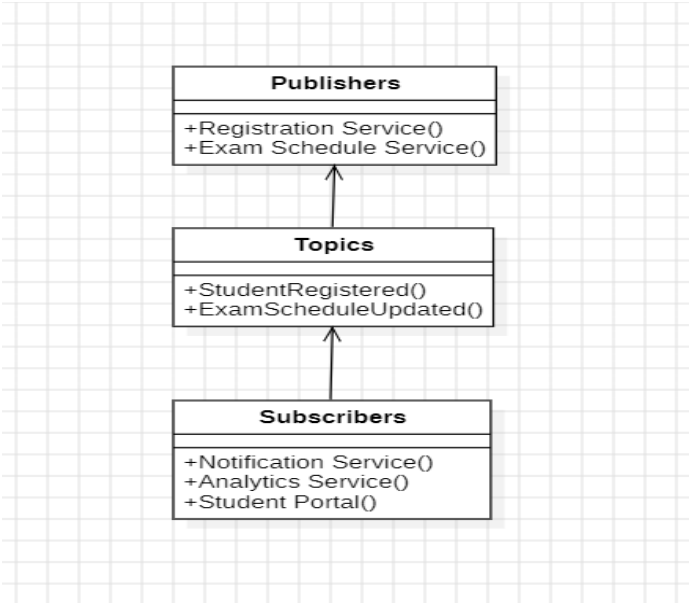
The Observer Pattern notifies components about system changes, enhancing flexibility and scalability. It enables easy addition of new notifications, improving system maintainability.

- ❖ 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.



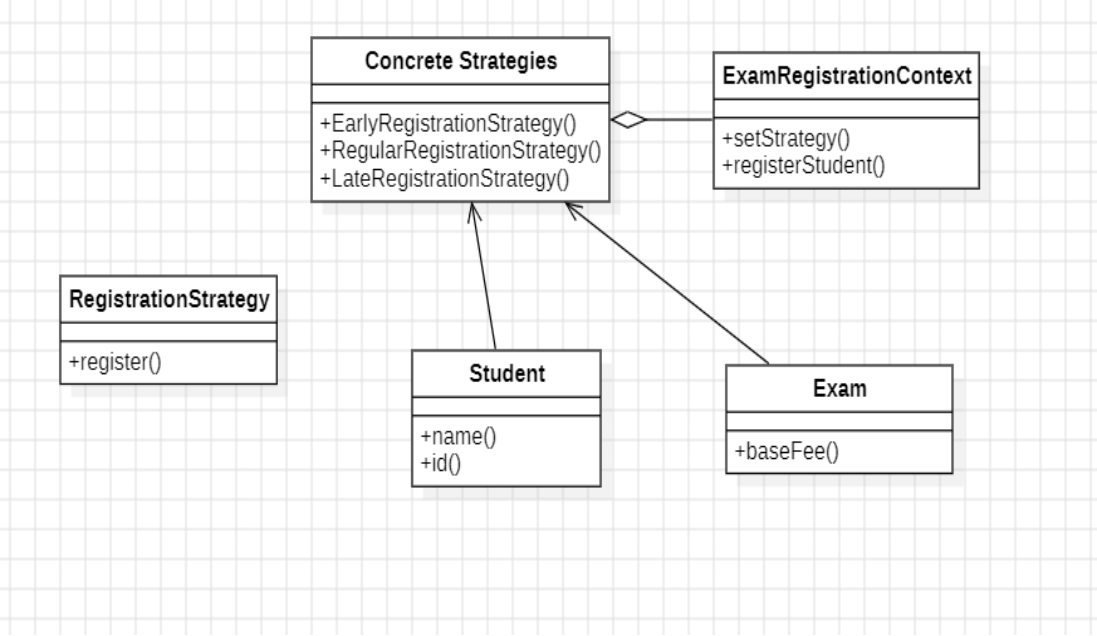

**3.Publish-Subscribe Pattern:**

- The Publish-Subscribe Pattern enables real-time updates and notifications across multiple components in trading systems.
- This pattern enhances scalability, flexibility, and responsiveness within the trading system architecture.
    - ❖ Context: Real-time updates are crucial in trading operations.
    - ❖ Problem: Distributing real-time market data without the Publish-Subscribe Pattern leads to data latency and inefficiencies.
    - ❖ Solution: The pattern enables efficient data distribution, letting providers publish to specific topics, ensuring relevant updates reach subscribing modules, enhancing system scalability and responsiveness.



**4.Strategy Pattern:**

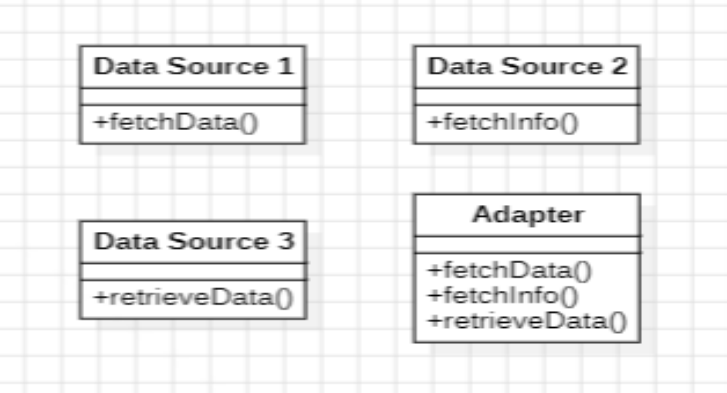
- The Strategy Pattern can be employed to handle different algorithms for international currency conversion.
- By implementing the Strategy Pattern, the system allows for interchangeable algorithms, providing flexibility and enabling users to select the most suitable currency conversion strategy.
    - ❖ Context: Used to manage a family of algorithms interchangeably.
    - ❖ Problem: Facilitates dynamic algorithm selection and switching, promoting code flexibility and reusability.
    - ❖ Solution: Define a strategy interface with algorithm methods. Implement concrete strategy classes for each algorithm and allow runtime switching between them.




**5.Adapter Pattern:**

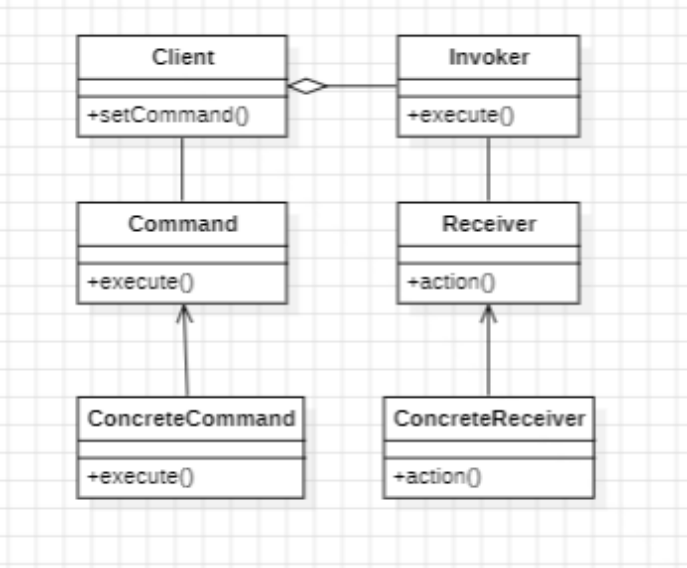
- The Adapter Pattern is used to integrate different trading data sources seamlessly.
- It solves the problem of incompatible interfaces between data sources and the trading system by providing a wrapper that translates data formats and protocols.
- This pattern ensures smooth communication and data exchange, enhancing interoperability and flexibility in the trading ecosystem.

- ❖ Context: Integrating diverse data sources with different formats and protocols is common.
- ❖ Problem: Incompatibility between these data sources and the trading system leads to communication issues and data parsing errors.
- ❖ Solution: Adapter Pattern provides a wrapper translating data formats and protocols for seamless integration without compromising data integrity or communication.



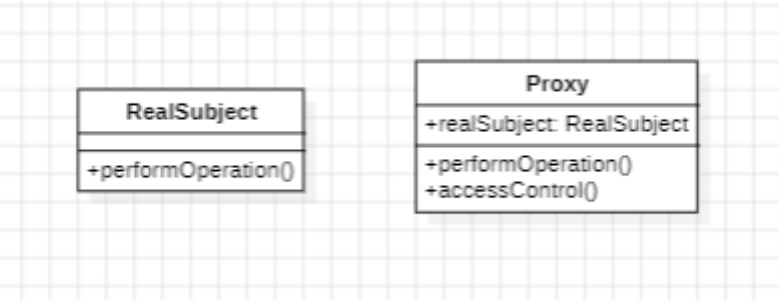

**6.Command Pattern:**

- The Command Pattern encapsulates requests as objects, allowing parameterization of clients with different requests, queuing, logging, and supporting undoable operations.
- This pattern enables the system to support undoable operations, transactional behavior, and logging, enhancing maintainability, scalability, and reliability within the system.
    - ❖ Context: Used to encapsulate requests as objects, enabling parameterized and queued requests.
    - ❖ Problem: Decouples sender and receiver, supports undoable operations, and provides structured request handling.
    - ❖ Solution: Define command objects that encapsulate requests and parameters. Clients create and queue commands, and invokers execute them when needed



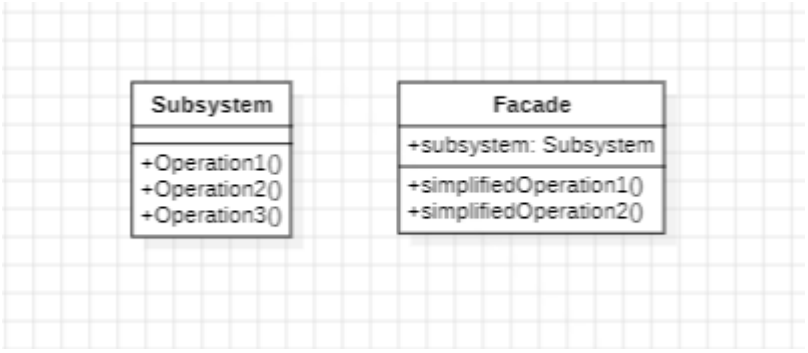

**7.Proxy pattern:**

- The Proxy Pattern is used to control access to sensitive trading data .
- It solves the problem of directly accessing such resources by providing a surrogate object that acts as a representative.
  - ❖ Context: Need to control access to sensitive trading data or manage expensive operations.
  - ❖ Problem: Direct access to such resources can lead to security risks or performance issues.
  - ❖ Solution: Proxy Pattern provides a surrogate object managing access, adding functionalities like caching or logging



**8.Facade pattern:**

- In a Exam Registration System, the Facade Pattern simplifies interactions with complex subsystems by providing a unified interface.
- It solves the problem of managing multiple subsystems with varying complexities by abstracting their functionality behind a single facade class.
  - ❖ Context: Direct interactions with subsystems cause code complexity and increased dependencies.
  - ❖ Problem: Managing multiple subsystem interactions leads to complex code and dependencies.\
  - ❖ Solution: Facade Pattern simplifies interactions, hiding subsystem complexities for easier system management and maintenance.




By incorporating these design patterns into the Exam Registration System, we can improve the reusability and maintainability by promoting code reuse, encapsulating complexity, and decoupling components. This leads to a more modular, flexible, and maintainable architecture, making it easier to extend, modify, and maintain the system over time.

**Result:**

Thus, the reusability and maintainability of the software system by applying appropriate design pattern was improved successfully,