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 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 BPO Management 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 BPO management 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 BPO Management System		
7.		Implement the system as per the detailed design		
8.		Test the software system for all the scenarios identified as per the usecase 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 – BPO Management System

INTRODUCTION:

In today's business landscape, Business Process Outsourcing (BPO) plays a crucial role in streamlining operations and improving efficiency. A BPO Management System is essential for optimizing BPO processes, enhancing client communication, and ensuring quality service delivery. This introduction highlights the significance of such a system and outlines its key features.

Features of BPO Management System:

1. Client Management:

Allow administrators to manage client profiles, contracts, and requirements.

Implement authentication and authorization mechanisms for secure client access.

Define client categories and access levels (e.g., premium clients, standard clients).

2. Task Allocation and Monitoring:

Facilitate task assignment to agents based on client requirements and agent skills.

Provide real-time monitoring of task progress, deadlines, and agent performance.

3. Communication Tools:

Offer communication channels (e.g., chat, email) for seamless interaction between clients and BPO agents.

Implement notifications and alerts for task updates, deadlines, and client inquiries.

4. Quality Assurance:

Incorporate mechanisms for quality control and compliance with service-level agreements(SLAs).

Enable agents to submit reports, track errors, and implement corrective actions.

5. Reporting and Analytics:

Generate reports on task performance, client satisfaction, and agent productivity.

Provide analytics dashboards for insights into BPO operations and areas for improvement ,Architecture of BPO Management System.

6. Client-Server Architecture:

Adopt a client-server architecture where the web application serves as the client, and the server hosts the application logic and data.

Utilize RESTful APIs for communication between the client interface and server-side components.

7. Relational Database Schema:

Design a relational database schema to store client information, task details, agent profiles, and transactional data.

Ensure data integrity, normalization, and efficient querying for optimal system performance.

Overall, the BPO Management System aims to streamline BPO operations, enhance client satisfaction, and improve overall efficiency in delivering outsourced services.

RESULT:

Thus the software system that is need to be developed for BPO management system was executed successfully.

Ex.No:02	Document the Software Requirements Specification (SRS) for BPO Management System

AIM:

To implement a software for BPO management system

PROBLEM STATEMENT:

With the reduction in communication costs and improved bandwidths and associated infrastructure, BPO as a segment is witnessing a massive growth. One of the key challenges that BPO companies that provide data entry/data validation services is an efficient and effective way of getting the source documents from different customers and accurately route the same to different operators for processing.

SOFTWARE REQUIREMENT SPECIFICATION:

1.0 Introduction

Business process outsourcing (BPO) is a subset of outsourcing that involves the contracting of the operations and responsibilities of specific business functions (or processes) to a third-party service provider. Originally, this was associated with manufacturing firms, such as Coca Cola that outsourced large segments of its supply chain.[1] In the contemporary context, it is primarily used to refer to the outsourcing of business processing services to an outside firm, replacing in-house services with labor from an outside firm.

BPO is typically categorized into back office outsourcing - which includes internal business functions such as human resources or finance and accounting, and front office outsourcing -which includes customer-related services such as contact center services.

BPO that is contracted outside a company's country is called offshore outsourcing. BPO that is contracted to a company's neighboring (or nearby) country is called near shore outsourcing.

1.1 Purpose

- Huge reduction in printing, dispatch costs.
- Seamless process that is fully integrated ensuring better quality of service to customers.
- Cost reductions by automation of upload processes from clients; automatic routing of documents to operators using OCR.
- Documents as well as the status of process is accessible quickly and from anywhere to BPO management as well as customers.

- Security of documents as they are stored in digital form
- Enriched experience for users as they can search for documents and process them online.

1.2 Scope

- As part of BPO, documents need to be managed between the outsourcing company and the offshore company.
- Multiple clients need to be managed by the BPO Company.
- Security of the documents has to be ensured so that there is no unauthorized access of the documents to other organizations.
- Quick turnaround times have to be managed.
- Appropriate process flow of the documents has to be present in the system to check the status of the documents at any point of time.

1.3 Definitions, Acronyms and The Abbreviations

- Client- One who wishes to outsource their job
- BPOS- Refers to this Business Process Outsourcing System
- BPO Organization- Refers to this Business Process Outsourcing Organization
- HTML - Markup Language used for creating web pages.
- J2EE – Java 2 Enterprise Edition is a programming platform java platform for developing and running distributed java applications.
- HTTP - Hyper Text Transfer Protocol.
- TCP/IP – Transmission Control Protocol/Internet Protocol is the communication protocol used to connect 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 –

1. Overall Description will describe major role of the system components and interconnections.
2. Specific Requirements will describe roles & functions of the actors.

2.0 Overall description

2.1 Product perspective

The BPOS acts as an interface between the 'client' and the 'BPO organization'. 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 the time duration in which the user receives the documents.

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 BPO system's server is directly connected to the client systems via ftp. The client systems have access to the database in the server.

2.4 System function

- BPO has been designed with the base product to suit the BPO vertical requirements.
- This is a complete web based solution, which enables the BPO Organization's clients to store documents automatically on the server.
- Clients can scan documents from multiple locations and an automatic uploader module that can be scheduled automatically uploads the documents.
- Documents uploaded are automatically routed to specific users based on the rules definable by the admin user.
- Documents uploaded are automatically routed to specific users based on the rules definable by the admin user.
- Data Entry and Quality check users can automatically download the documents in a web- based interface to do their respective operations on the documents uploaded.
- Comprehensive reports have been developed to track the status of the documents at any given point of time.

2.5 User Characteristics

• BPO Organization - They are the people who desire to obtain the outsourcing job from various clients and submit the information to the database.

• Client - He has the certain privileges to outsource their jobs and to approve the issue of document. He may contain a group of persons under him to verify the documents and give suggestion whether or not to approve the dispatch of job.

2.6 Constraints

• The BPO require a computer (FTP) to submit their information.

• Although the security is given high importance, there is always a chance of intrusion in the web world which requires constant monitoring.

• The user has to be careful while submitting the information. Much care is required.

2.7 Assumptions and Dependencies

• The organization and client must have basic knowledge of computers and English Language.

• Provide privacy and security for the documents and client information

RESULT:

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

Ex.No:03	Identify Use cases and develop the use case control for BPO Management System

AIM:

To identify use cases and develop the use case control for BPO management system.

USECASE DIAGRAM:

A use case is a list of actions or event steps typically defining the interactions between a role of an actor and a system to achieve a goal. A use case is a useful technique for identifying, clarifying, and organizing system requirements. A use case is made up of a set of possible sequences of interactions between systems and users that defines the features to be implemented and the resolution of any errors that may be encountered.

While a use case itself might drill into a lot of detail (such as, flow of events and scenarios) about every possibility, a use-case diagram can help provide a higher-level view of the system, providing the simplified and graphical representation of what the system must actually do.

A use case (or set of use cases) has these characteristics:

1. Organizes functional requirements
2. Models the goals of system/actor (user) interactions
3. Describes one main flow of events (main scenarios) and possibly other exceptional flows (alternatives), also called paths or user scenarios

The BPO management system use cases are:

- Search for client/job
- Negotiate the project
- Upload input data
- Perform required conversion
- Quality Check
- Shipment
- Payment

Actors:

Actors are as follows:

1. BPO Organization
2. Client

Actors Documentation:

1. **BPO Organization:**

   i) Searching the client

   ii) Downloads the input document

   iii) Views images and enters data in accounting package

   iv)Checks the quality or images, output of operators

   v) Uploads the output to the clients

2. **Client:**

Clients can scan documents from multiple locations and an automatic uploader module that can be scheduled automatically uploads the documents.

Use-case name:

1.Search for client/job

BPO organization searches the outsourcing job.

2.Negotiate the project

Once job has been found then negotiate with the client for doing that project.

3.Upload input data

After finalizing the negotiation client uploads the input to the BPO organization through FTP

4. Perform required conversion

BPO organization starts the required conversion process.

5. Quality Check

This usecase is used to ensure that the quality of the product. Randomly audits the outcome of the project to ensure the quality. This process is continued until we achieve the required quality.

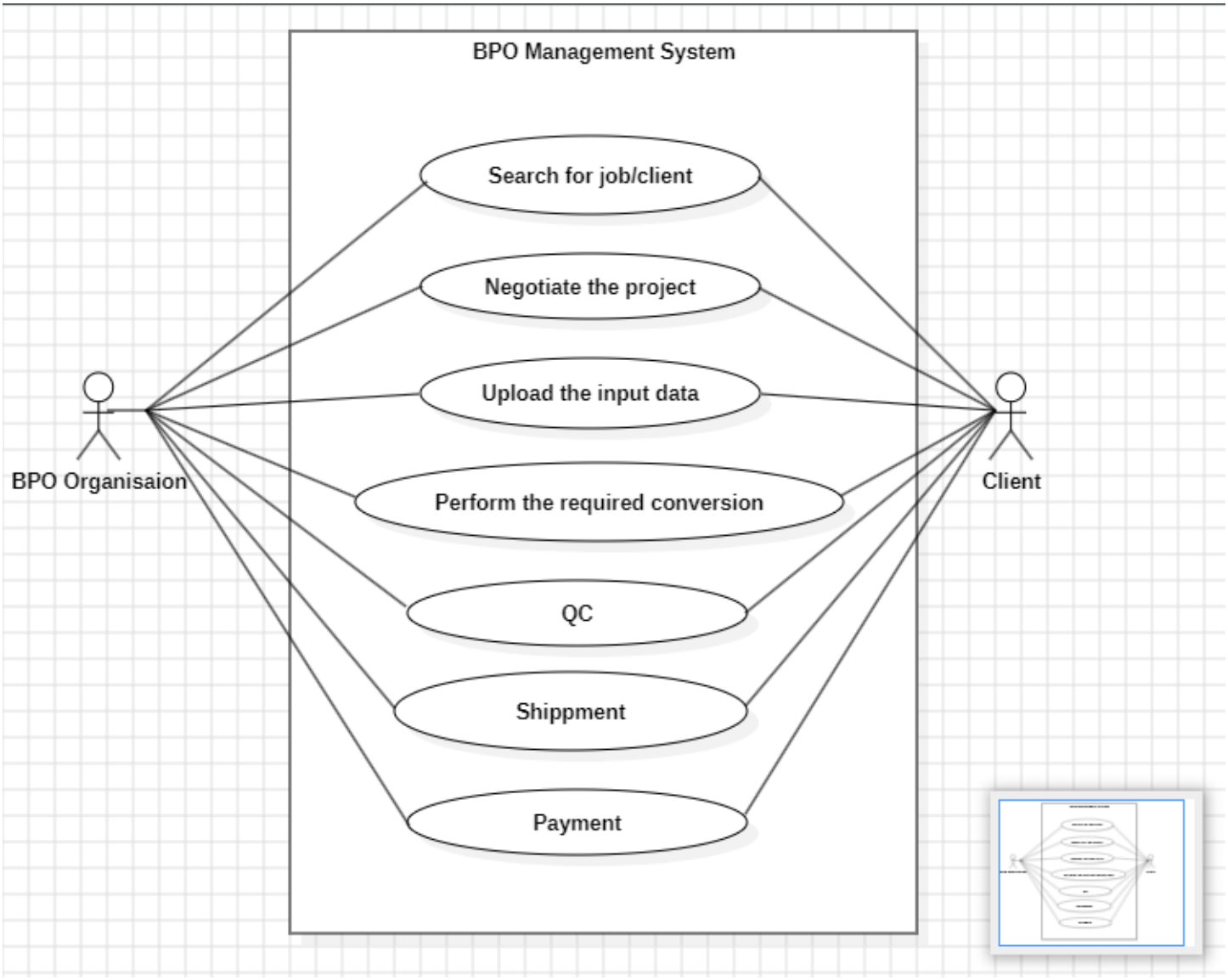
6. Shipment

After QC, upload the output to the client.

7. Payment

Get the payment for the project from client

UML USE-CASE DIAGRAM:



RESULT:

Thus the use case diagram for BPO management system was developed successfully.

Ex.No:04	Identify the conceptual classes and develop a Domain Model and also derive a class diagram for BPO Management System

AIM:

To identify the conceptual classes and develop a domain model and also derive a class diagram for BPO Management System

CLASS DIAGRAM:

- The UML class diagram is to illustrate class interfaces and their actions. They are used for static object modeling, 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 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.
- The class diagram describes the attributes and operations of a class and also the constraints imposed on the system.
- The class diagram is 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.
- 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:

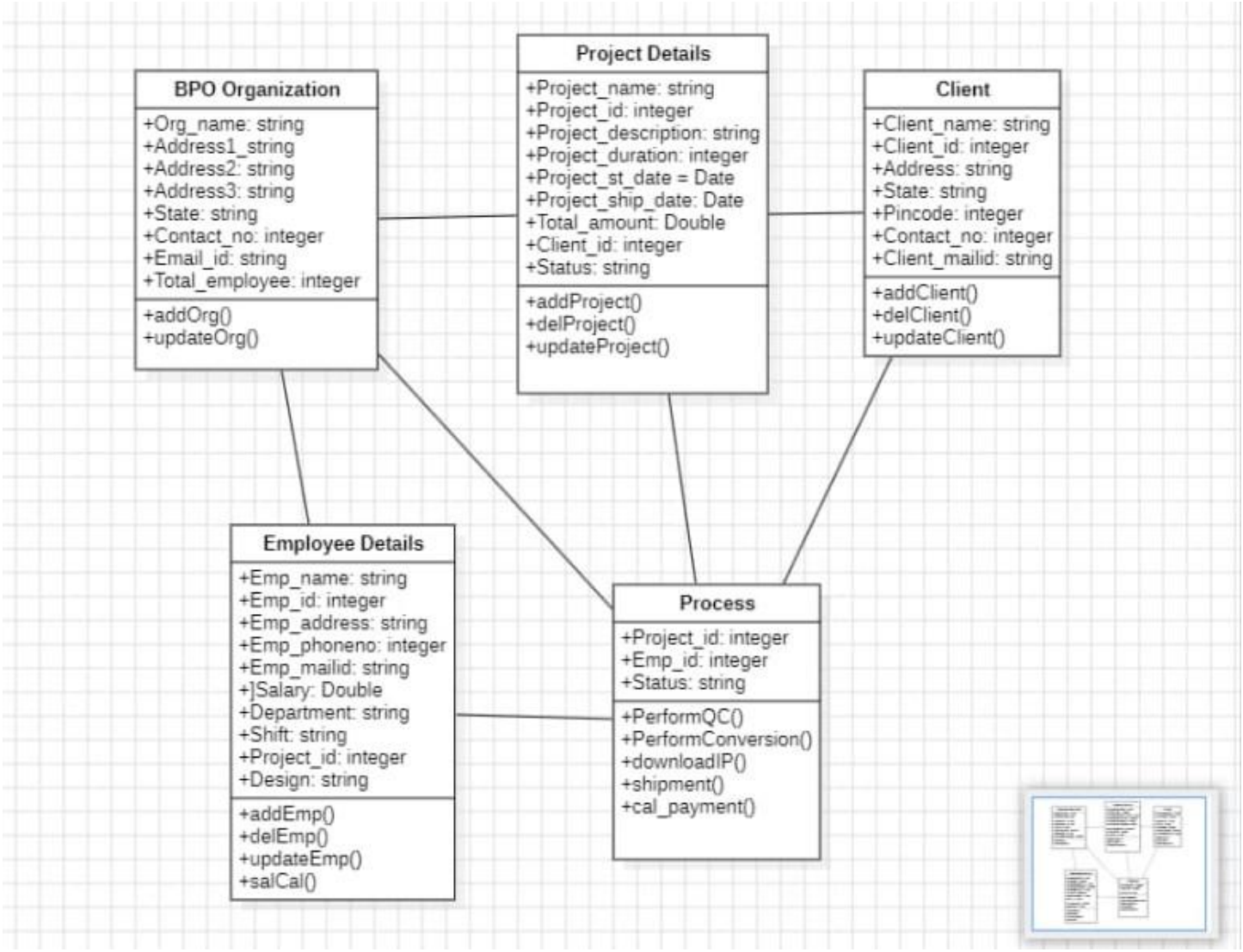
A class is drawn as a rectangle with three compartment.

1.Top hold class name

2.Middle hold class name

3.Bottome hold list of operation.

CLASS DIAGRAM FOR BPO:



RESULT:

Thus the class diagram for BPO Management 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 BPO Management System

AIM:

To represent the interaction between objects using UML Sequence diagram and Collaboration diagram for BPO Management 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:

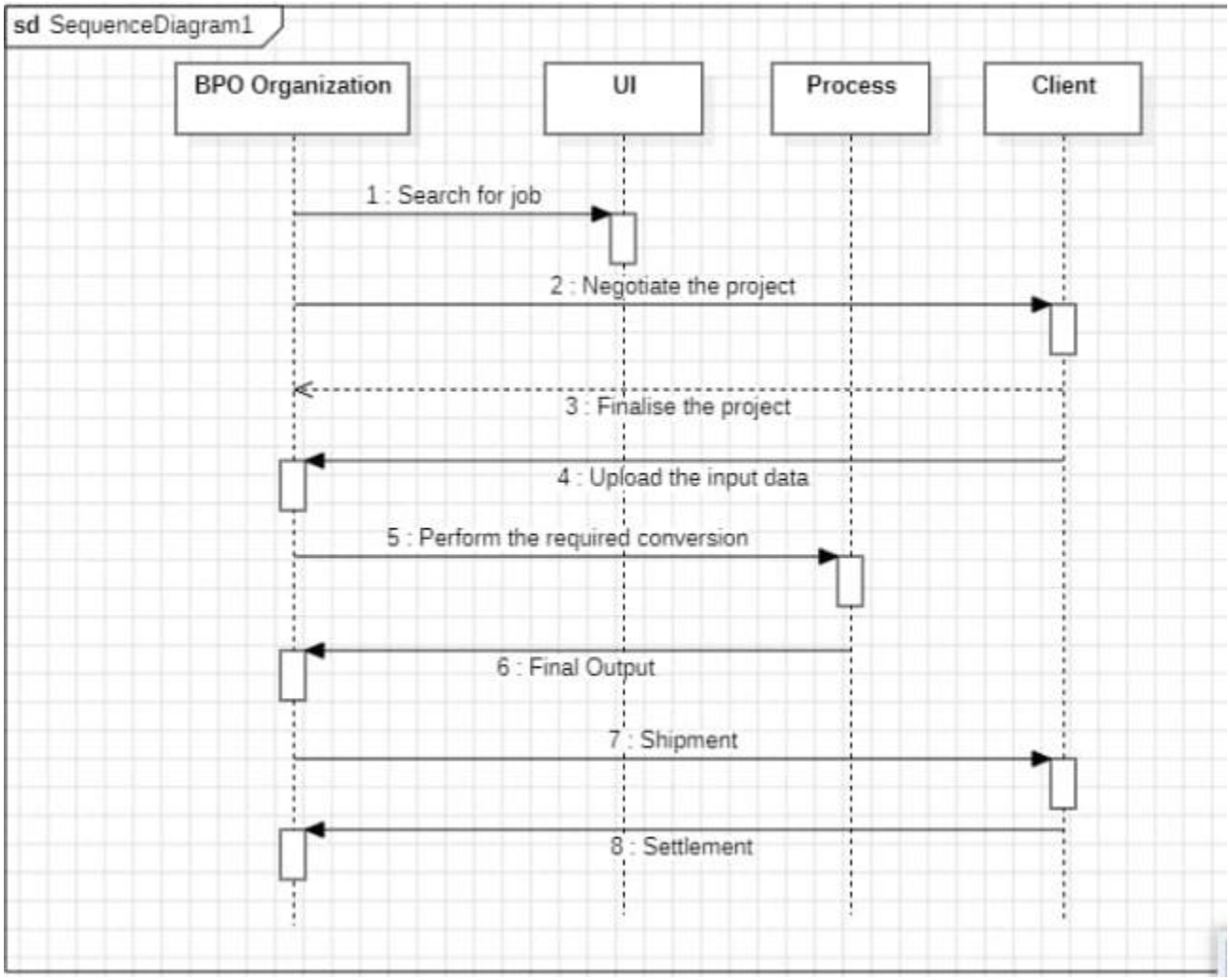
i) To capture dynamic behaviour of a system.

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 BPO:



COLLABORATION DIAGRAM:

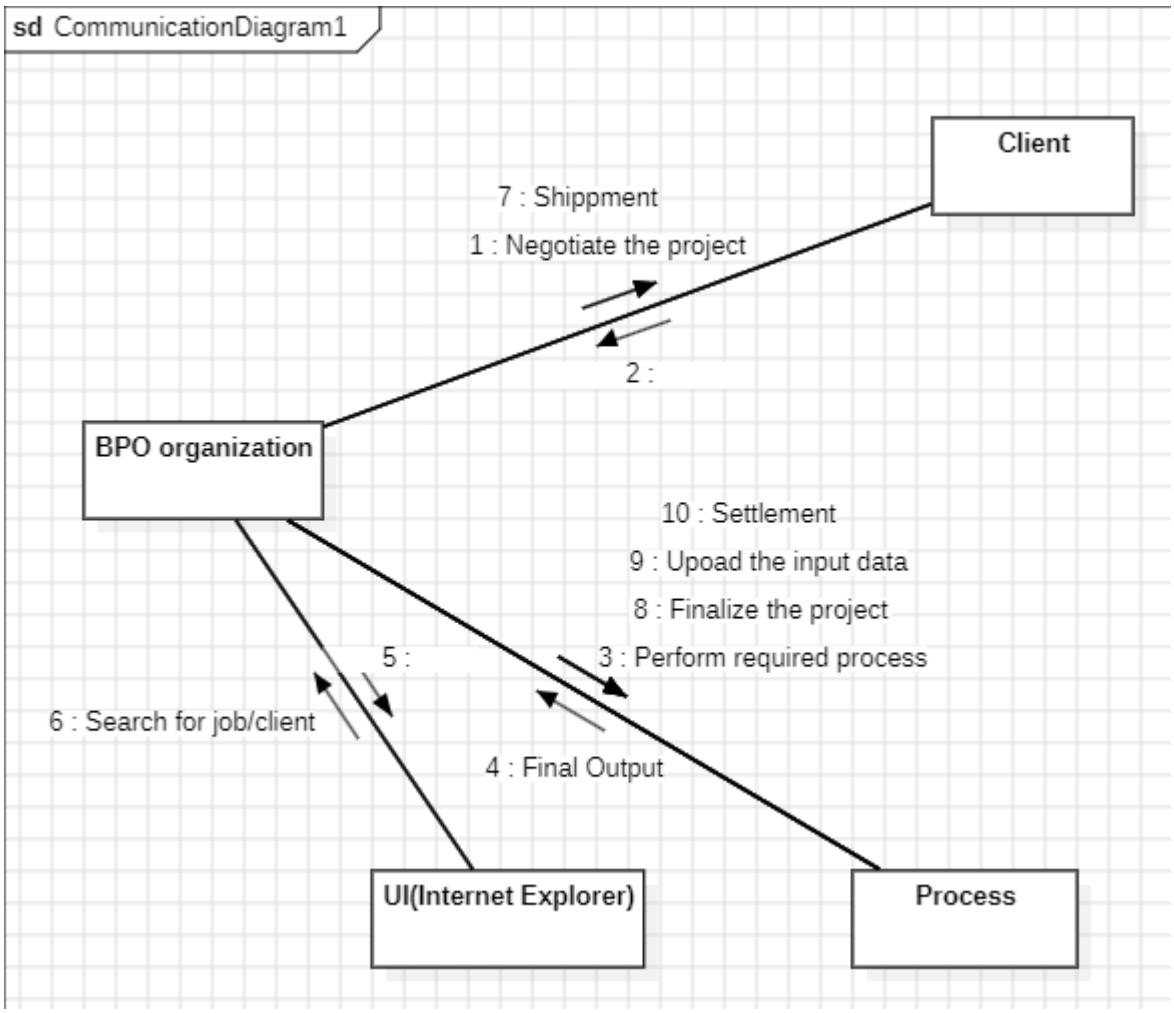
Communication diagram illustrate that object interact on a graph or network format in which object can be placed where on the diagram.

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 BPO:



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

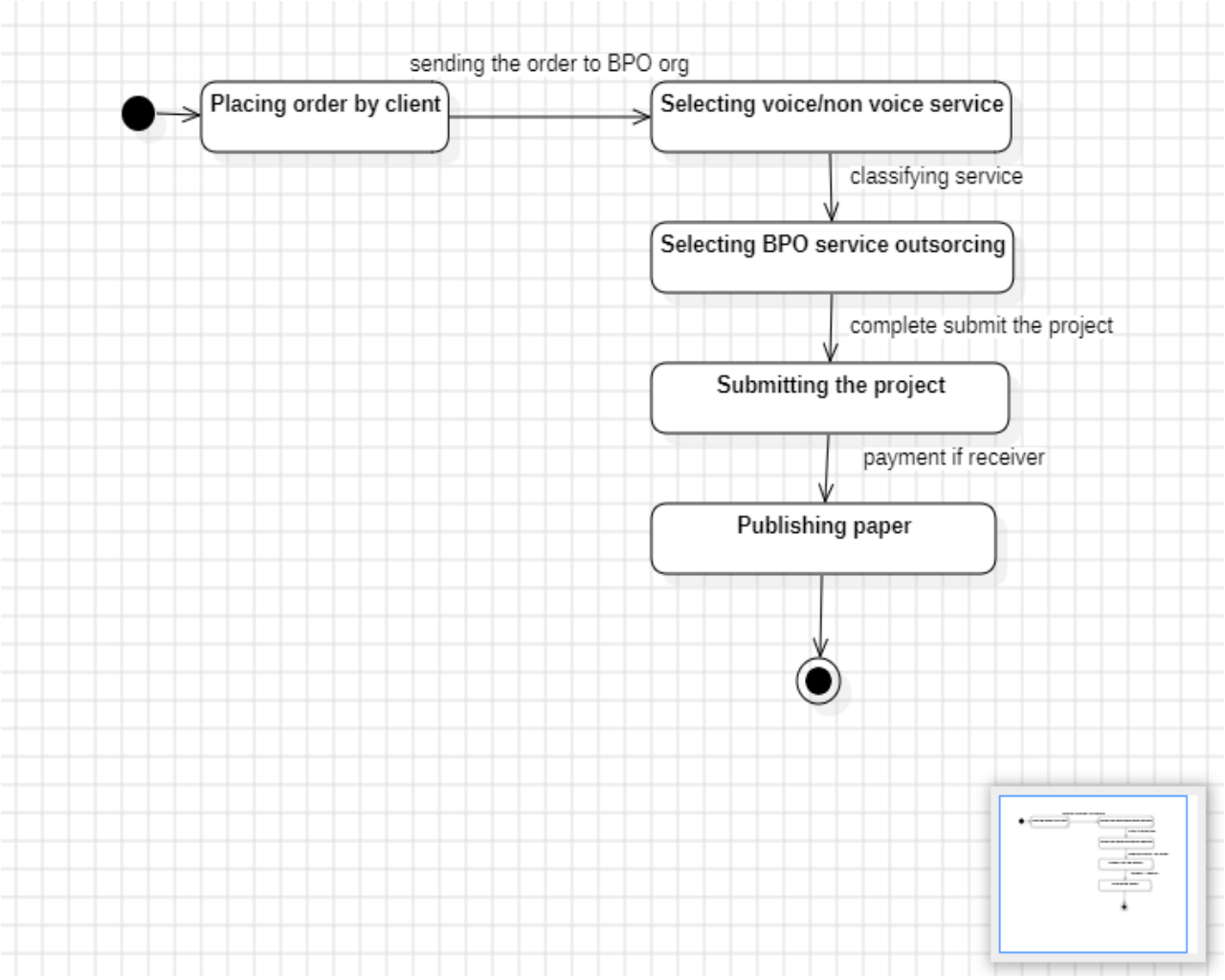
AIM:

To draw Activity diagram for BPO Management System

STATE CHART DIAGRAM:

State diagram describes all the possible states that particular objects can get into and how the object state changes as the result of events that reach the object. It describes the behavior of the system. At start point on initial transition call as get fast item leads to the checking state. Checking state has an activity, associated with check item.

The name of the diagram itself clarifies the purpose of the diagram and other details. It describes different states of a component in a system. The states are specific to a component/object of a system.



STATE CHART DIAGRAM FOR BPO

A Statechart diagram describes a state machine. Now to clarify it state machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events Statechart diagram is one of the five UML diagrams used to model dynamic nature of a system. They define different states of an object during its lifetime. And these states are changed by events. So Statechart diagrams are useful to model reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

State chart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. So the most important purpose of Statechart diagram is to model life time of an object from creation to termination. State chart diagrams are also used for forward and reverse engineering of a system. But the main purpose is to model reactive 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.

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.

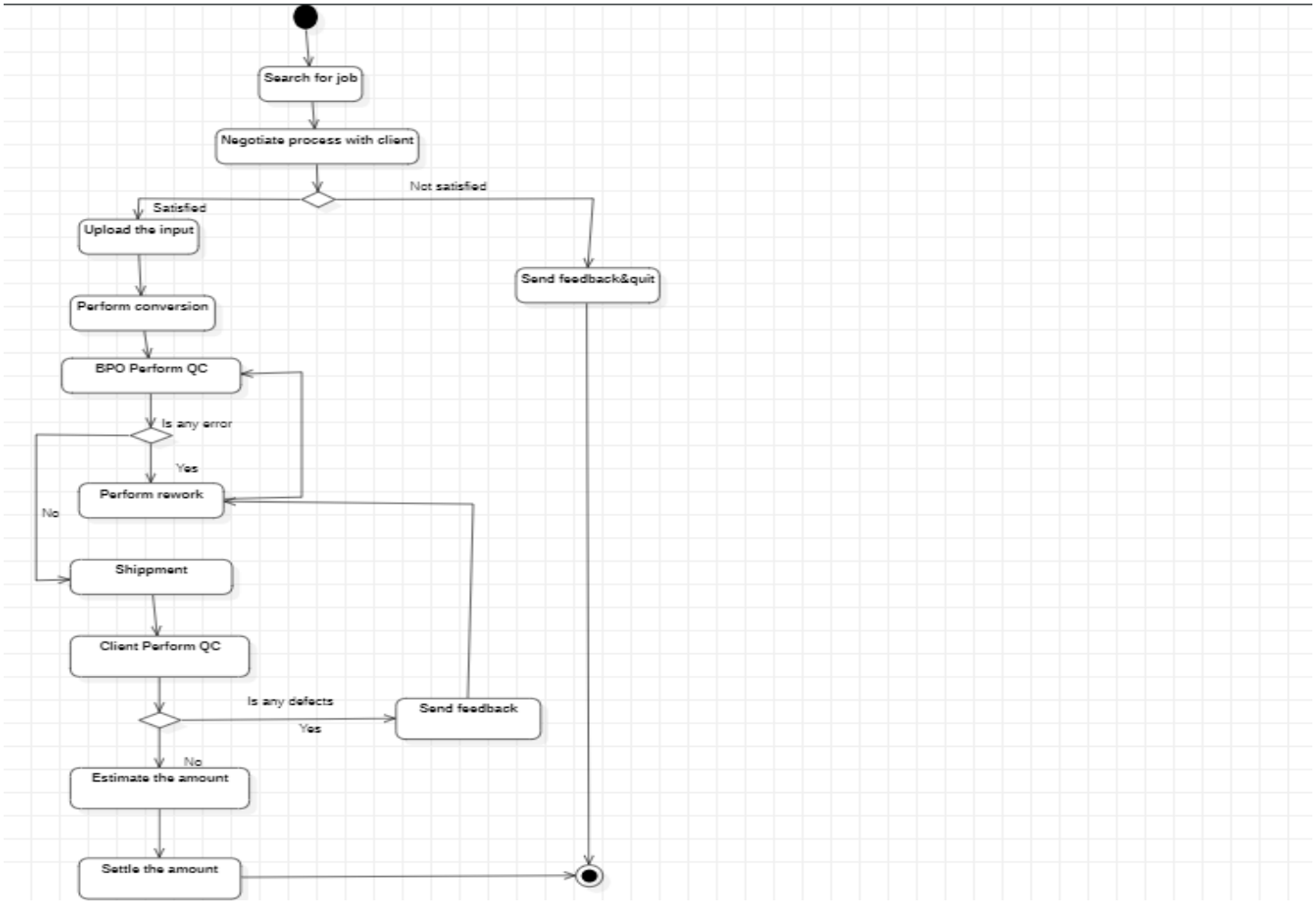
Activity is a particular operation of the system. Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is the message part. 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.

ACTIVITY DIAGRAM FOR BPO:



RESULT:

Thus the activity diagram for BPO Management System was developed successfully.

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

AIM:

To implement the BPO Management System as per the detailed design.

IMPLEMENTATION FOR BPO MANAGEMENT SYSTEM:

BPO Organisation.java

```
public class BPOOrganization
{
 private String orgname;
 private int clientId;
 private int projetId;
 private string addr1;
 private string addr2;
 private string addr3;
 private string state;
 private integer contactNo;
 private string emailid;
 private int TotalnoofEmp;
 public EmployeeDetails theEmployeeDetails;
 /**
 @roseuid 512B098E008C
 */
 public BPOOrganization()
 { }
 /**
 @roseuid 512B0638033C
 */
 public void addOrg()
 { }
 /**
 @roseuid 512B063F0186
```

```
 */
 public void updateOrg()
 { }
}
```

Client.java

```
public class client
{
 private string clientName;
 private integer clientId;
 private string address;
 private string state;
 private integer pincode;
 private integer contactNo;
 private string clientEmailId;
 public ProjectDetails theProjectDetails;
 public process theProcess;
 /**
 @roseuid 512B031701F4
 */
 public client()
 { }
 /**
 @param 0
 @roseuid 512B0269005D
 */
 public void addClient(int 0)
 { }
 /**
 @roseuid 512B026E02BF
 */
 public void delClient()
 { }
```

```
 /**
 @roseuid 512B0276001F
 */
 public void updateClient()
 {
 }
}
//void client.addClient(){
//
// }
```

EmployeeDetails.java

```
public class EmployeeDetails
{
 private String EmpName;
 private integer empId;
 private string EmpAddr;
 private integer empPhoneno;
 private string empEmailId;
 private double salary;
 private string Shift;
 private string dept;
 private integer projectId;
 private string design;
 public BPOOrganization theBPOOrganization;
 /**
 @roseuid 512B098E0177
 */
 public EmployeeDetails()
 { }
 /**
 @roseuid 512B08A50109
 */
```

```
 public void addEmp()
 {
 }
 /**
 @roseuid 512B08A901D4
 */
 public void delEmp()
 { }
 /**
 @roseuid 512B08AD009C
 */
 public void updateEmp()
 { }
 /**
 @roseuid 512B08B102DE
 */
 public void SalCal()
 {
 }
}
```

Process.java

```
public class process
{
 private int proj.id;
 private int emp.id;
 private string status;
 public client theClient;
 /**
 @roseuid 51274911032C
 */
 public process()
 { }
```

```
/**
@roseuid 51273C0F036B
*/
public void performQc()
{ }
/**
@roseuid 51273C120222
*/
public void performConversion()
{
}
/**
@roseuid 51273C12030D
*/
public void download()
{
}
/**
@roseuid 51273C13000F
*/
public void shipment()
{
}
/**
@roseuid 51273C1300FA
*/
public void calRenumeration()
{
}
}
```

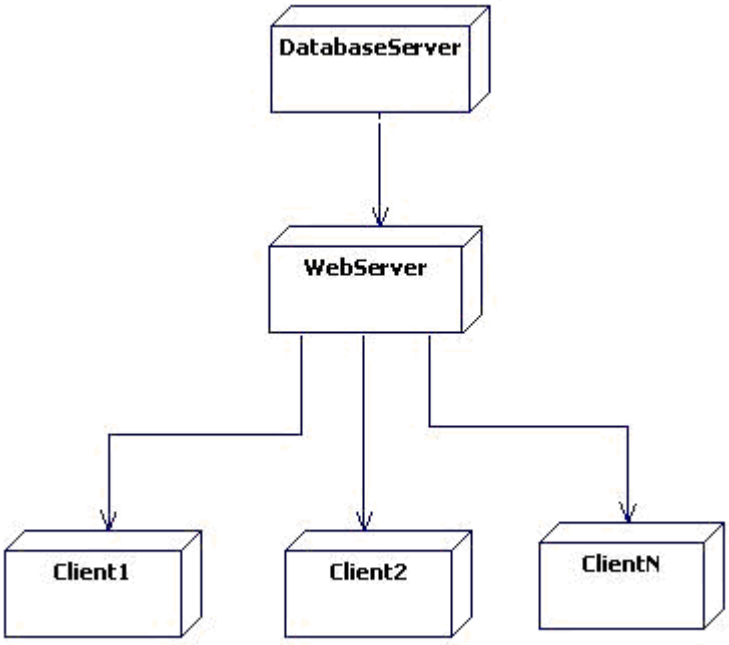
ProjectDetails.java

```
public class projectDetails
```

```
{
 private STRING projname;
 private int proj.id;
 private string proj.description;
 private string proj.duration;
 private date date;
 private date shipdate;
 private double amt;
 private int clientId;
 private string status;
 public process theProcess;
 public client theClient;
 /**
 @roseuid 512749110280
 */
 public projectDetails()
 {
 }
 /**
 @roseuid 51273A2203A9
 */
 public void addProject()
 {
 }
 /**
 @roseuid 51273A240186
 */
 public void delProject()
 {
 }
 /**
 @roseuid 51273A240242
```

```
 */
 public void status()
 {
 }
}
```

DEPLOYMENT DIAGRAM:



RESULT:

Thus the implementation of BPO Management System was executed and the codes were generated successfully.

Ex.No:08	Test the software system for all scenarios identified as per the use case diagram

Aim:

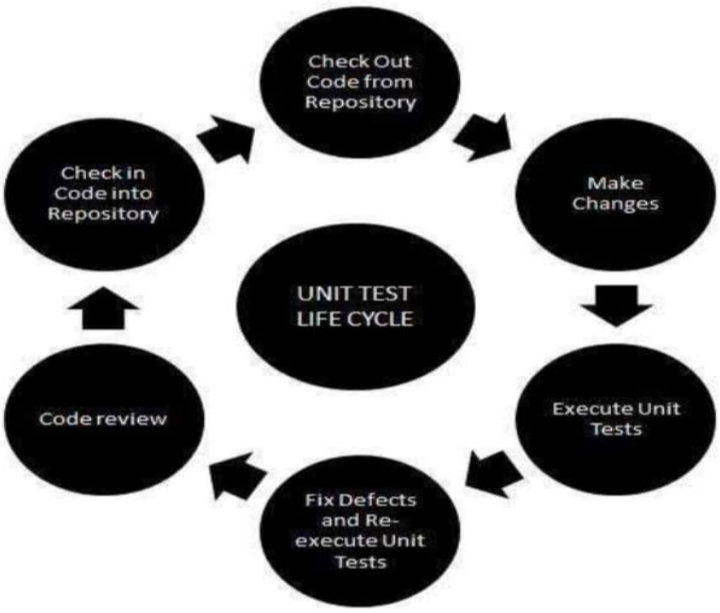
To test the software system for all scenarios identified in the use case diagram.

BPO Management System:

In the realm of outsourcing, effective management of business processes is critical for operational efficiency and client satisfaction. A BPO management system streamlines the process of handling client requests, assigning tasks to employees, ensuring quality assurance, and facilitating client communication.

. **i) Unit Testing:**

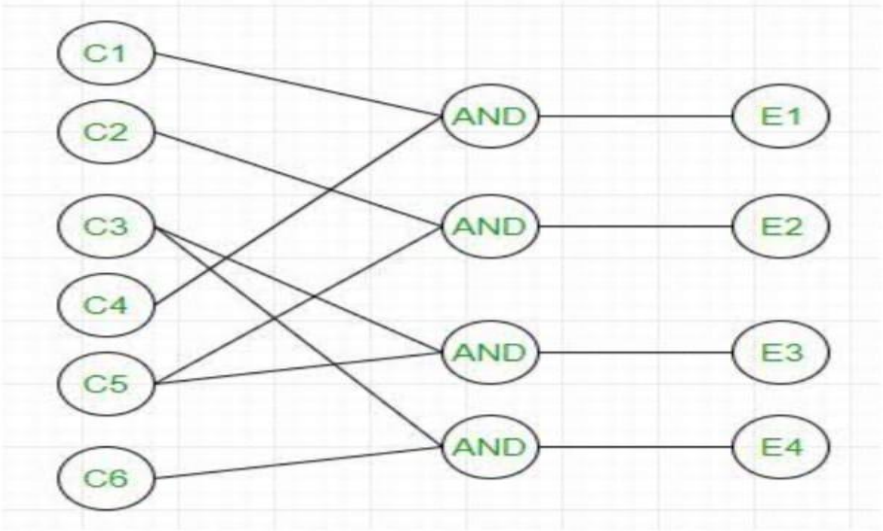
- Unit testing involves testing individual components or modules of the software to ensure they function correctly in isolation.
- Each module, such as client management, project management, employee management, etc., would undergo unit testing.
- Example: Testing the "addClient()" function to ensure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



ii) 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.

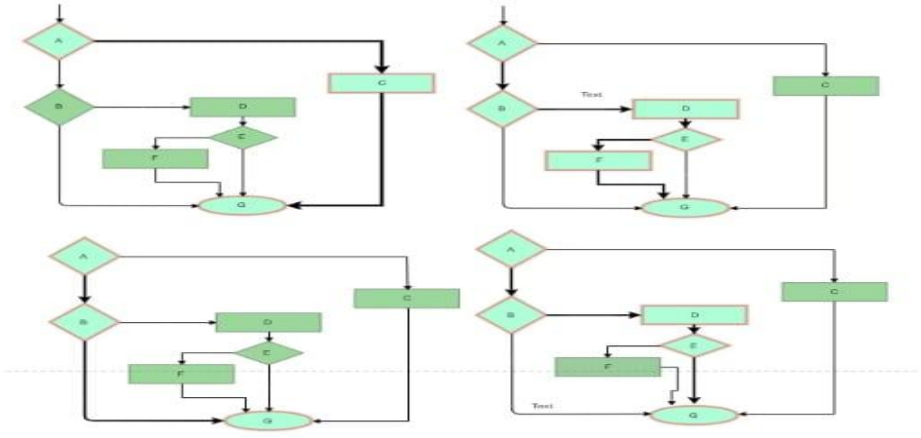
- Testers would input various sets of data into the system and verify that the expected output is produced.
- Example: Testing the "searchForJob()" functionality to ensure that it returns the expected results based on different search criteria.



		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

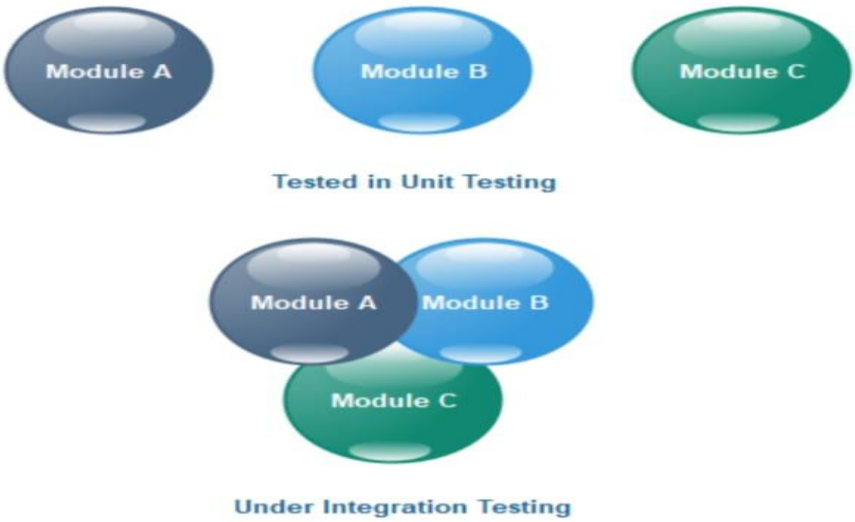
iii) White Box Testing:

- 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.



iv) Integration Testing:

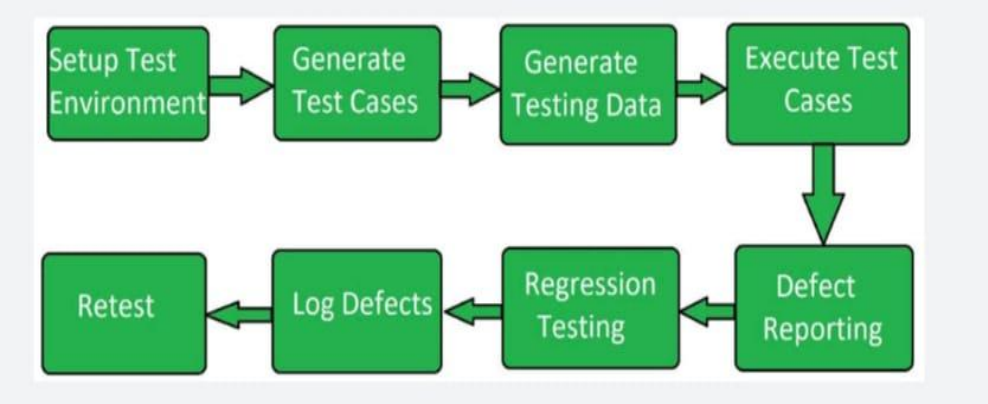
- Integration testing verifies that different modules of the software work together as expected.
- Testers would test the interaction between different modules, such as client management, project management, and employee management.
- Example: Testing the interaction between the "addClient()" and "addProject()"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



v) System Testing:

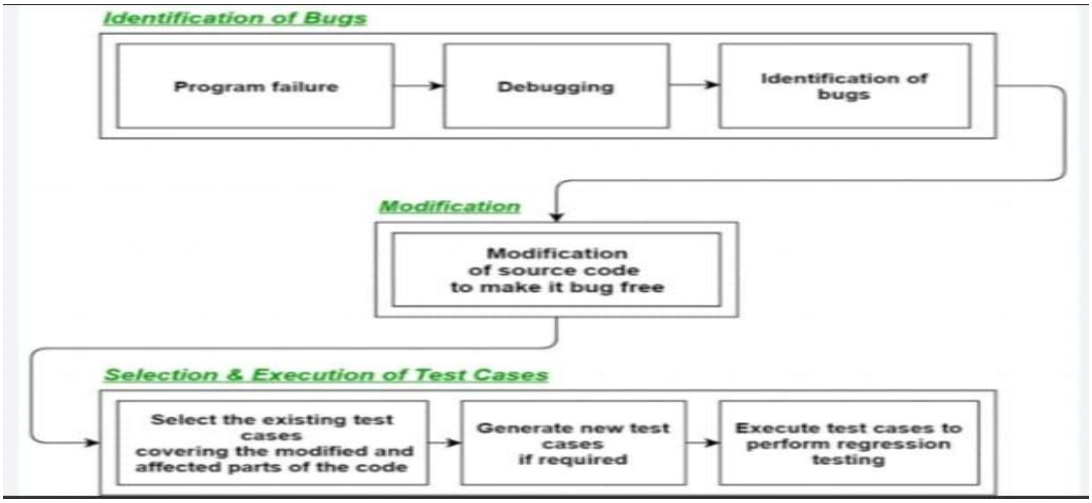
- System testing involves testing the entire system as a whole to verify that it meets the specified requirements.
- Testers would perform end-to-end testing of the entire system, including all modules and their interactions.

- Example: Testing the entire process from client negotiation to project delivery and payment to ensure that it functions as expected.



vi) 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 by applying appropriate design patterns

Aim:

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

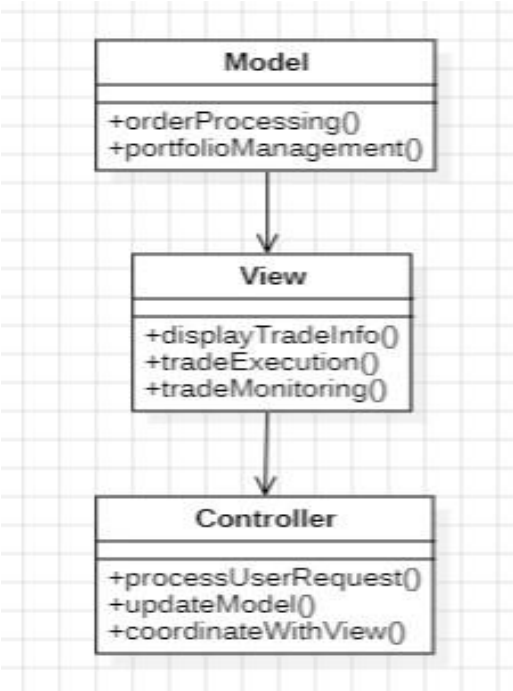
BPO Management System:

A BPO (Business Process Outsourcing) management system is a software application designed to streamline and optimize various business processes outsourced to a BPO service provider. It facilitates the organization, monitoring, and management of tasks, workflows, and resources involved in BPO operations. The system typically includes functionalities for task allocation, tracking, reporting, client communication, and quality assurance.

To enhance the reusability and maintainability of the BPO management system, we can incorporate several design patterns. Here are some design patterns that can be applied:

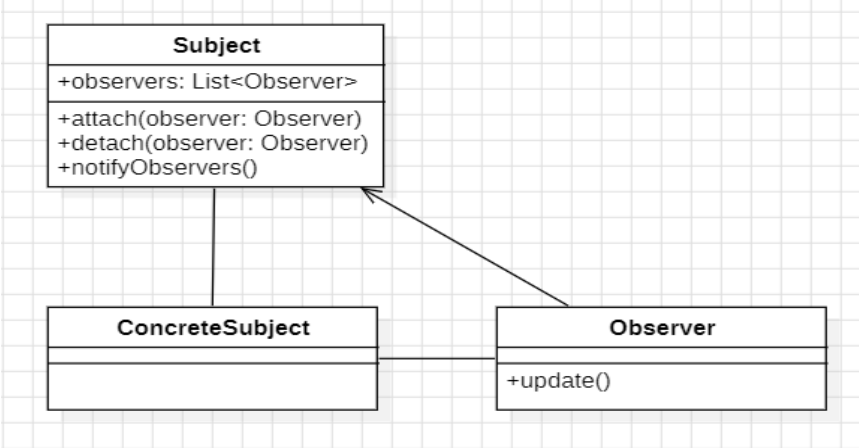
i)Model-View-Controller (MVC):

The MVC pattern separates the representation of information from the user's interaction with it. In the BPO Management 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).



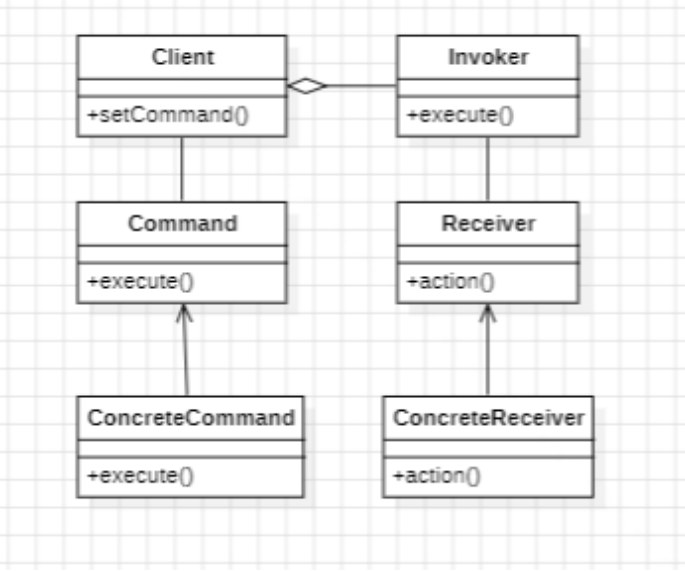
ii) Observer Pattern:

The Observer pattern establishes a one-to-many dependency between objects, where multiple observers are notified of changes in a subject. In the BPO Management System, it is used to notify clients or users about the status of their projects, such as when a project is completed, when data is uploaded, when payments are processed, etc.



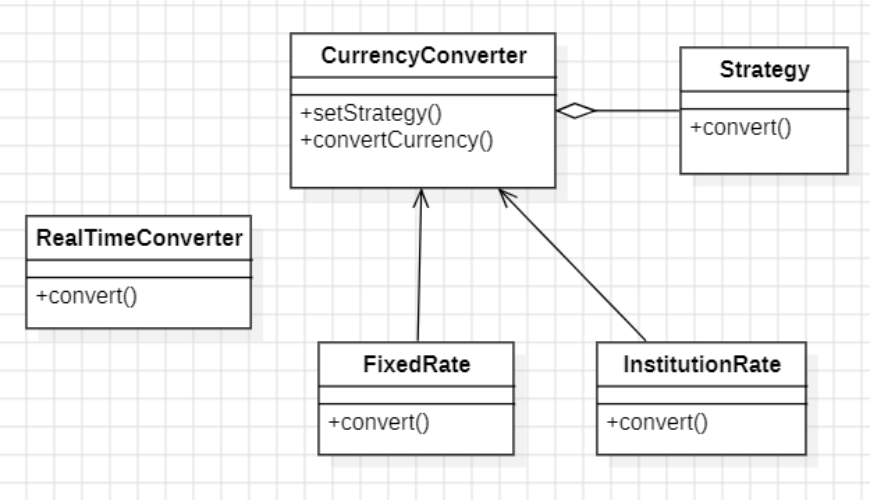
iii) Command Pattern:

The Command pattern encapsulates requests as objects, allowing for parameterization of clients with queued or logged requests. In the BPO Management System, it is used to implement undo/redo functionality, batch processing, or asynchronous operations.



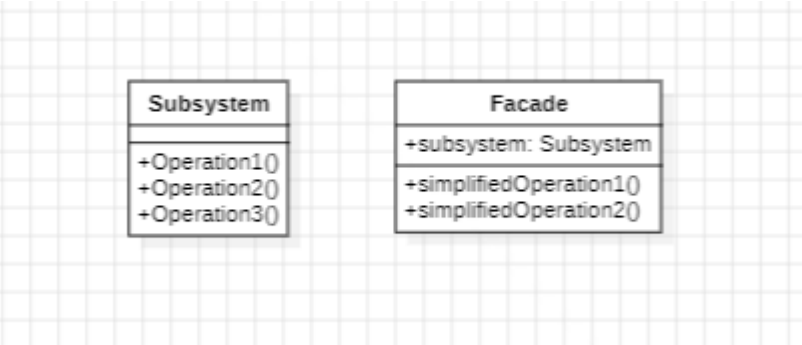
iv) Strategy Pattern:

The Strategy pattern defines a family of algorithms, encapsulates each one, and makes them interchangeable. In the BPO Management System, it is used to implement different task prioritization algorithms or task allocation strategies based on workload.



v) Facade Pattern:

The Facade pattern provides a simplified interface to a complex subsystem, hiding its complexities from clients. In the BPO Management System, it is used to abstract away the complexities of task routing, assignment, and tracking.



Result:

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