



*We Make You Shine*  
**St. JOSEPH'S INSTITUTE OF TECHNOLOGY**  
(An Autonomous Institution)  
**St. JOSEPH'S GROUP OF INSTITUTIONS**  
OMR, CHENNAI - 119



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**  
**ACADEMIC YEAR (2024-2025) ODD SEMESTER**  
**INNOVATIVE TEACHING METHODS**

Name of Pedagogy Used:	<b>SOFTWARE ENGINEERING VIRTUAL LAB</b>
Branch/Year/Sem/Sec:	<b>CSE/III/V/B&amp;C</b>
Subject Code/Subject Name:	<b>CS4508/SOFTWARE ENGINEERING &amp; DESIGN LAB</b>
Topic:	<b>TECHNIQUES AND METHODOLOGIES IN SOFTWARE ENGINEERING LAB</b>
Date/Period/Timing	<b>20.09.2024/7/1.00 PM TO 1.40 PM</b>
Link for Virtual Lab	<b><a href="http://vlabs.iitkgp.ernet.in/se/">HTTP://VLABS.IITKGP.ERNET.IN/SE/</a></b>
<b>Description</b>	<p>The Software Engineering Virtual Lab has been developed by keeping in mind the following objectives:</p> <ul style="list-style-type: none"><li>• To impart state-of-the-art knowledge on Software Engineering and UML in an interactive manner through the Web</li><li>• Present case studies to demonstrate the practical applications of different concepts</li><li>• Provide a scope to the students where they can solve small, real life problems</li></ul>





<b>Students Feedback</b>	<p><b>312422104174 :</b>We have learned the importance of various associations in UML diagrams.</p> <p><b>312422104179:</b> We have realized the importance of UML diagrams in the Analysis and design phase in Software Engineering.</p>
Total No. of Students	22
No. of Students Present	21
No: of Students Absent	01
Action Plan for Absentees	<ul style="list-style-type: none"> <li>Planned to conduct another virtual lab session for the absentees.</li> </ul>

### **DOCUMENT PROOF**

## **A REPORT ON INNOVATIVE TEACHING MODEL**

### **SOFTWARE ENGINEERING VIRTUAL LAB**

#### **1. MODELING UML USE CASE DIAGRAMS AND CAPTURING USE CASE SCENARIOS**

**Draw a use case diagram for the following problem**

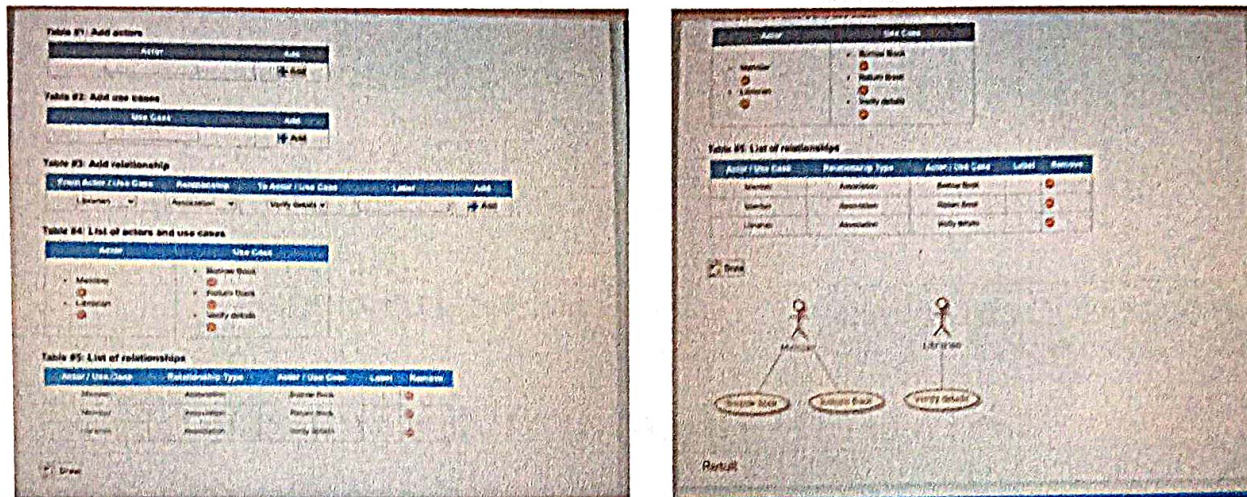
Consider a library, where a member can perform two operations: issue book and return it. A book is issued to a member only after verifying his credentials. Draw a use case diagram for the problem.

**Learning Objectives:**



- Identify the actors and use cases
- Associate the use cases with the actors by drawing a simple use case diagram

**Limitations:** While extending a use case, extension points could not be defined through this interface.



## 2. IDENTIFYING DOMAIN CLASSES FROM THE PROBLEM STATEMENTS

**Identify the domain classes from the following problem statement**

The latest cab services agency in the city has approached you to develop a Cab Management System for them. Following is the information they have given to implement the system.

### Learning Objectives:

1. Identifying potential classes (and their attributes) from a given problem statement
2. Use expert knowledge on the subject matter to identify other relevant classes

**Limitations:** The workspace provided is constrained to solve the current problem only. This is not a general user interface to solve *any* problem.



**Table #1: Add nouns / noun phrases identified from the problem statement**

Noun / Noun phrase	Add
<input type="text"/>	<input type="button" value="+ Add"/>

**Table #2: Categorization of the above identified nouns (Rose's method)**

Potential objects	Category	Add
<input type="checkbox"/> Passenger <input type="checkbox"/> Driver <input type="checkbox"/> Cab	Place	<input type="button" value="+ Add"/>

**Table #3: List of categorized nouns (potential objects)**

Place	People	Things	Organization	Concept	Event	Redundant

**Table #4: Identify attribute(s) for objects**

Object	Attributes	Add
<input type="checkbox"/> Address <input type="checkbox"/> Color <input type="checkbox"/> Cost <input type="checkbox"/> Weight <input type="checkbox"/> Name <input type="checkbox"/> Serial number <input type="checkbox"/> Time <input type="checkbox"/> Weight	<input type="button" value="+ Add"/>	

**Table #5: Add other possible attribute(s) [Optional]**

Attribute	Add
<input type="text"/>	<input type="button" value="+ Add"/>

**Table #6: List of objects and it's attribute(s)**

Object	Attributes

**Table #7: Define classes from list of attribute(s)**

Attributes	Class	Add
<input type="checkbox"/> Address <input type="checkbox"/> Color <input type="checkbox"/> Cost <input type="checkbox"/> Weight <input type="checkbox"/> Name <input type="checkbox"/> Serial number <input type="checkbox"/> Time <input type="checkbox"/> Weight	<input type="text"/>	<input type="button" value="+ Add"/>

**Table #8: List of classes and their attribute(s)**

Classes	Attributes

**Table #9: Add any other relevant class (based on domain knowledge) [Optional]**

Additional class	Add
<input type="text"/>	<input type="button" value="+ Add"/>

☐ Top Level Classes

**Table #10: List of top level classes**

Class	Object

### 3. STATECHART AND ACTIVITY MODELING

- Capturing the dynamic view of a system is very important for a developer to develop the logic for a system. State chart diagrams and activity diagrams are two popular UML diagram to visualize the dynamic behaviour of an information system.
- In this experiment, we will learn about the different components of activity diagram and state chart diagram and how these can be used to represent the dynamic nature of an information system.

**Draw a State chart diagram to graphically represent the following system**

Consider a bulb with a push down switch. The bulb initially remains off. When the switch is pushed down, the bulb is on. Again when the switch is pushed up, the bulb turns off. The lifecycle of the bulb continues in this way until it gets damaged.

**Think about these points:**

- What are the different states of the bulb?
- What activities are performed in each state?
- What action does make the bulb move from one state to another?

**Learning Objectives:**

1. Identifying different states of a system



## 2. Identifying activities performed in each state

**Limitations:** A complex system often has sub-states, which is not covered as a part of this lab. The following interface only let you represent simple states. Please check out the references section to know more about them.

The screenshot displays a software interface for creating a statechart diagram. It includes four configuration tables and a visual diagram.

**Table #1: Add states of the system**

State Name	Add
	+ Add

**Table #2: Internal activities of a state**

State Name	Action Label	Action Expression	Add
Damaged	Do		+ Add

**Table #3: Add a note [optional]**

State Name	Note	Position	Add
State		Select position	+ Add

**Table #4: List of system states**

State	Activities	Note Position	Remove
ON	entry / Turned On		
OFF	exit / Switched OFF		
Damaged	do / Repeat ON and OFF		

**Table #5: List of state transitions**

Current State	Event	Guard Condition	Action	Next State	Remove
Initial	Switched ON		ON	ON	
ON	Switched ON		Switched ON	ON	
OFF	Turned OFF		OFF	Damaged	
Initial	Switched off			ON	
ON	Switched off			OFF	
OFF	Repeat			Damaged	
Damaged	Stop			Final	

The diagram shows a statechart with three states: ON, OFF, and Damaged. Transitions are labeled with events and actions: 'Switched ON / ON' from Initial to ON, 'Switched ON / Switched ON' from ON to ON, 'Switched off' from ON to OFF, 'Turned OFF / OFF Repeat' from OFF to OFF, 'Repeat ON and OFF' from OFF to Damaged, and 'Stop' from Damaged to Final.

## 4. DESIGNING TEST SUITES

### Learning Objectives:

1. Get familiarized with unit testing
2. Verify implementation of functional requirements by writing test cases
3. Analyze results of testing to ascertain the current state of a project

**Limitations:** This workspace attempts to provide a very simple version of a testing framework. Real life testing frameworks are much more extensive and provide a lot of options like creating test cases from user requirements, automatic reporting of bug when a test case fails, and so on. Nevertheless, this workspace is expected to make a student familiar to testing and some of its templates and reports.

### Design a test suite for the following problem

The Absolute Beginners Inc. seems to have been fascinated by your work. Recently they have entrusted you with a task of writing a web-based mathematical software (using JavaScript). As part of this software, your team mate has written a small module, which computes area of simple geometric shapes. A portion of the module is shown below.



Virtual Labs  
All MHRD Govt. of India Initiative

Home Credits Feedback Advanced Network Technologies Virtual Lab Virtual Labs

## Designing Test Suites

Introduction Theory Simulation Case Study Self-evaluation Procedure Exercises References

Select 1

**Design a test suite for the following problem**

The Absolute Beginners Inc. seems to have been fascinated by your work. Recently they have entrusted you with a task of writing a web-based mathematical software (using JavaScript). As part of this software, your team mate has written a small module, which computes area of simple geometric shapes. A portion of the module is shown below.

```
function square(side) { return side * side ; }
function rectangle(side1, side2) { return side1 * side2 ; }
function circle(radius) { return Math.PI * radius * radius ; }
function right_triangle(base, height) { return 1 / 2 * base * height ; }
```

Prepare a test suite that will

- Verify each of the above mentioned individual function is working correctly

Your task essentially is to verify whether each of the above function is returning correct values for given inputs. For example, a rectangle with length 10 unit and breadth 5 unit will have an area of 50 sq. unit. This can be verified from the output of the function call

```
rectangle(10, 5);
```

However, testing also attempts to point out possible bugs in the software. How would the above code behave for a call

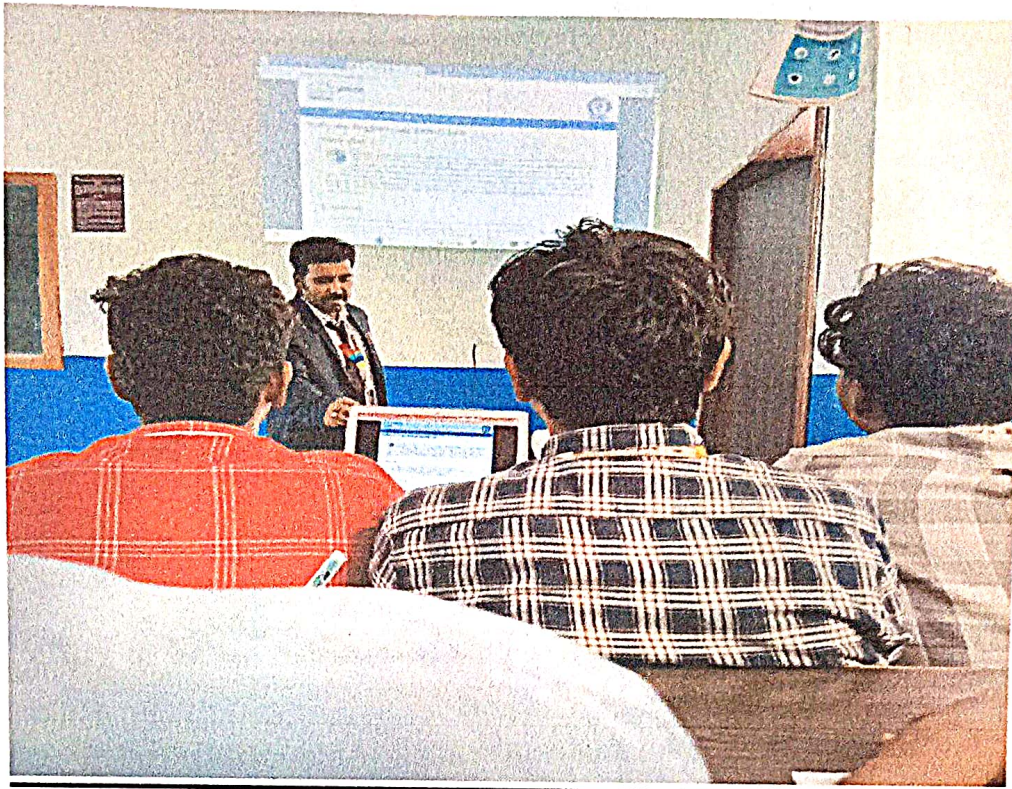
```
rectangle(10, -5);
```

### (i) Session on Discussing various Problem Statements





**(ii) Session on Various Associations in UML Class Diagrams**



**FACULTY INCHARGE**

**Dr.J. DAFNI ROSE, Ph.D.**  
Professor & Head  
Department of CSE  
St. Joseph's Institute of Technology  
hodcse@stjosephstechnology.ac.in