

Exercise 2: System Design with CASE Tools (18 Problems)

20. CASE Tool Setup – Install and Configure StarUML or Visual Paradigm

CASE (Computer Aided Software Engineering) tools help in creating UML diagrams and managing software design visually.

Popular tools:

StarUML

Visual Paradigm

Steps:

Download the tool from official website

Install following setup wizard

Configure:

Default UML notation

Project folder location

Diagram templates

Benefits:

- Faster diagram creation
- Error checking
- Professional documentation
- Team collaboration support

21. Team Workspace Setup – Set Up Collaborative Environment

A team workspace allows multiple members to work on the same project.

Typical Setup:

Create shared project repository (cloud/local server)

Assign roles:

Analyst

Designer

Developer

Enable:

Version control

Diagram locking

Change tracking

Advantages:

- Prevents overwriting work
- Improves collaboration
- Maintains design history

22. Project Structure Creation in CASE Tool

Create organized folders inside the tool:

- Requirements
- Use Case Diagrams
- Class Diagrams
- Sequence Diagram
- Activity Diagrams
- Component Diagrams
- Deployment Diagrams

Purpose:

1. Easy navigation
2. Professional documentation
3. Scalable design structure

23. Actors Identification

Actors represent users or external systems interacting with software.

Common Actors (Example: Student Management System)

Student

Admin

Faculty

Payment Gateway

System Database

Why important:

- Defines system boundaries
- Clarifies responsibilities

24. Use Case Definition (8–12 Core Use Cases)

Examples:

Login

Register Student

Enroll Course

View Grades

Pay Fees

Generate Report

Manage Users

Update Profile

Logout

Purpose:

Shows system functionality

User interaction flow

25. Use Case Relationships

Include (mandatory reuse):

Example:

Enroll Course → includes → Login

Extend (optional behavior):

Example:

Login → extends → Forgot Password

Generalization:

Admin and Student inherit from User actor

Benefits:

- Reduces repetition
- Improves clarity

26. Use Case Documentation Format

Each use case should contain:

Example – “Enroll Course”

Preconditions:

Student must be logged in

Main Flow:

Student selects course

System verifies availability

System enrolls student

Alternative Flow:

If course is full → show error message

Postconditions:

Enrollment saved successfully

27. Class Identification

Typical classes:

Student

Course

Enrollment

Payment

User

Report

Admin

Purpose:

Represents real-world objects in system

28. Class Attributes & Methods

Example: Student Class

Attributes:

studentID

name

email

password

Methods:

login()

enrollCourse()

viewGrades()

29. Class Relationships

Association

Student — enrolls — Course

Aggregation

Course contains multiple Students

Inheritance

Admin → User

Student → User

Benefits:

- Shows object interaction
- Defines system structure

30. Sequence Diagrams (3–4 Critical Ones)

Typical diagrams:

Login process

Course enrollment

Fee payment

Report generation

What they show:

Message flow between objects

Time order of actions

31. Object Interactions in Sequence Diagrams

Example: Login Sequence

Student → Login UI → Auth System → Database

Messages:

enterCredentials()

verifyUser()

returnStatus()

32. Alternative Flows in Sequence Diagrams

Example:

- Valid login → dashboard opens
- Invalid login → error message displayed

Shows exception handling clearly.

33. Activity Diagrams (2 Required)

Typical ones:

Student enrollment workflow

Admin report generation

Elements:

Start node

Activities

Decision points

End node

34. Workflow Documentation

Include:

- Loops (retry login)
- Decisions (payment success/failure)
- Parallel actions (email confirmation + database update)

Purpose:

Visualizes business process clearly.

35. Component Diagram

Shows system architecture:

Components such as:

User Interface

Authentication Module

Database Module

Payment Module

Reporting Module

Benefits:

- Shows software organization
- Helps developers understand system structure

36. Deployment Diagram

Shows physical setup:

- Client Device
- Web Server
- Database Server

Connections show data flow.

Purpose:

Real-world system installation view

Performance planning

37. Design Document Creation

A formal document explaining:

Includes:

System overview

UML diagrams

Design choices

Technology used

Security decisions

Performance considerations

Why important:

- Helps future developers
- Supports maintenance
- Professional project record